République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Ferhat Abbas de Sétif- Sétif 1
Faculté des Sciences
Département d'Informatique

Thèse présentée par:

Said Gadri

Pour l'obtention du grade de:

Doctorat de Sciences
En Informatique

Thème

Catégorisation Automatique Contextuelle de Documents Semi-structurés Multilingues

Soutenue publiquement le ……

Devant un jury composé de:

Mr. TOUAHRIA Mohamed	Prof, Univ. Setif 1	Président
Mr. MOUSSAOUI Abdelouhab	Prof, Univ. Setif 1	Rapporteur
Mr. KHOLLADI Mohamed Kheireddine	Prof, Univ. El-Oued	Examinateur
Mr. CHIKHI Salim	Prof, Univ. Constantine 2	Examinateur
Mr. BOUBATRA Abdelhak	Prof, Univ. BBA	Examinateur
Mr. SAIDI Mohamed	Assoc.Prof , Univ.Setif 1	Examinateur

2015 - 2016

The People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University Ferhat Abbas of Setif- Setif 1
Faculty of Science
Department of Computer Science

A Dissertation submitted by:

Said Gadri

In partial fulfillment of the requirements for the degree of:

Doctor of Sciences
In Computer Science

Subject:

Automatic Contextual Categorization of multilingual semi-structured documents

Defended on …….

Board of Examiners:

Mr. Mohamed Touahria	Prof, Univ. Setif 1	Chairman
Mr. Abdelouhab Moussaoui	Prof, Univ. Setif 1	Supervisor
Mr. Mohamed Kheireddine Kholladi	Prof, Univ. El-Oued	Examiner
Mr. Salim Chikhi	Prof, Univ. Constantine 2	Examiner
Mr. Abdelhak Boubatra	Prof, Univ. BBA	Examiner
Mr. Mohamed Saidi	Assoc.Prof , Univ.Setif 1	Examiner

2015 - 2016

Acknowledgments

The defense of Ph. D thesis was always one of the many significant transitions in my life. That because the present thesis contains the outcome of past years research efforts I have done since my first inscription. It summarizes my contributions in the field of automatic text classification and other related fields. This work becomes a reality because of many wonderful people who have accompanied me during the establishment of this research. So, it's the occasion to express my deep gratitude and my warm thanks to all of them and estimate their support.

First and formost, I thank my thesis advisor Pr. Abdelouahab Moussaoui for his significant support during the establishment of this work, for his entire availability and his patience during this long time, for providing his advices and expressions of encouragements, for all his guidance and support which have been essential in my development as researcher.

My deep thanks to my wife for her invaluable help, her patience, and her famous encouragements.

My warm thanks are addressed to my dear mother for encouraging me since I was a small boy to achieve my objective and realize my dream.

I am very grateful to Dr. Bénoit Favre (LIF Laboratory, Aix-Marseille University, France) who has been of invaluable help in earlier stages of this work.

I thank also Pr. Frederic Béchet, Pr. Alexis Nacer (LIF Laboratory, Aix-Marseille University, France) who have received me for a short internship (Nov 2011) and provide me all kind of help.

I am indebted to Pr. Erich Neuhold (University of Vienna, Austria) for his collaboration in a part of this thesis.

I am very grateful to our colleage Dr. Belabdelouahab Linda for its estimated help during the revision stage (English revision).

I would like thank all the member of examination committee to take out time from their full schedule and to accept the evaluation of my work.

I also address my warm thanks to the staff members of computer science department at the university Ferhat Abbas of Setif 1, especially Mr. Lekhfif Abdelaziz who was very wonderful and helpful.

I knowledge the efforts from all my teachers who have taught me in: the primary school, the college, the secondary school, University of Setif, University of M'sila.

Said Gadri

Dedications

I dedicate this work to:

The soul of my father – may Allah bless his soul

My beloved mother – may allah protect her and prolong her life.

My beloved wife - may Allah protect her

My wonderful sons and daughters

All my family and my nearest persons

My dear colleagues without noting their names

My students of computer science department.

All scientists and learners

Said Gadri

Abstract

Text categorization is an important task in text mining process that consists in assigning a set of texts to a set of predefined categories based on learning algorithms. There exist two kinds of text categorization: monolingual and multilingual text categorization. The main problematic of this thesis is how to exploit concepts and algorithms of machine learning in contextual categorization of multilingual texts. Our study on this subject allowed us to propose many solutions and provide many contributions, notably: a simple, fast and effective algorithm to identify the language of a text in multilingual corpus. An improved algorithm for Arabic stemming based on a statistical approach, its main objective is to reduce the size of term vocabulary and thus increase the quality of the obtained categorization in TC and the effectiveness of search in IR. A new multilingual stemmer which is general and completely independent of any language. Application of new panoply of pseudo-distances to categorize texts of a big corpus such as Reuters21578 collection. All these solutions were the subject of many academic papers published in international conferences and journals.

Résumé

La catégorisation de textes est une tache très importante dans le processus de text mining. Cette tache consiste à affecter un ensemble de textes à un autre ensemble de catégories selon leurs thèmes et en exploitant les algorithmes d'apprentissage connus dans le domaine d'intelligence artificielle. Notre étude sur cet axe de recherche nous a permis de proposer quelques solutions et de porter certaines contributions, notamment: Proposer un algorithme simple, rapide et efficace pour identifier la langue d'un texte dans un corpus multilingue. Développer un algorithme amélioré pour la recherche des racines des mots arabes en se basant sur une approche complètement statistique. L'objectif principal de cet algorithme est de réduire la taille du vocabulaire de termes et par conséquent améliorer la qualité de la catégorisation obtenue dans le domaine de la catégorisation de textes et augmenter l'efficacité de la recherche dans le domaine de la recherche d'information. Développer un nouveau stemmer multilingue qui est plus général et indépendant de toute langue. Application d'une nouvelle panoplie de pseudo-distances pour catégoriser les textes d'un corpus de grande taille (Reuters21578). Toutes ces solutions étaient l'objet de papiers scientifiques publiés dans des conférences et des journaux internationaux indexés.

ملخّص

يعتبر التصنيف الآلـــي للنصوص من أهمّ المهـــامّ الرئيسية ضمن ما يعرف بعمليـــة التنقيب عن المعطيـات النصّيـة Text Mining. و يتمثّل أساسا في إلحاق مجموعة من النصوص مختلفة المواضيع بمجموعة أخرى من الأصناف يمثل كلّ واحد منها موضوعا معالجا محدّدا. و يتحقّق ذلك من خلال استعمال خوارزميات التعلّم الآلي المعروفة في مجال الذكاء الاصطناعي. و يمكن تمييز نوعين من التصنيف الآلي : تصنيف أحادي اللغة و تصنيف متعدّد اللّغات, و قد مكّنتنا الدراسة المعمّقة في هذا المجال من اقتراح مجموعة من الحلول و إضافة جملة من المساهمات نجملها فيما يلي: تطوير خوارزمية بسيطة، سريعة و فعّالة للتعرّف على لغة نصّ ضمن مصنّف نصوص متعدد اللغات. اقتراح خوارزمية مطوّرة تعتمد على مقاربة إحصائية للبحث عن جذور الكلمات العربية الهدف الرئيسي منها هو تقليص حجم قاموس المفردات المستخرج من مصنّف النّصوص و من ثمّ تحسين نوعية التصنيف الآلي لهذه النصوص و دقة البحث بالنسبة لنظم البحث عن المعلومات. تطوير نظامّ تجذيع متعدّد اللّغات يتميّز بالعمومية و الاستقلالية التامّة عن أيّ لغة. تطبيق تشكيلة جديدة من شيه المسافات pseudo–distances على التصنيف الآلي لمصنّف من الحجم الكبير مثل مصنّف رويترز 21578. كلّ هذه الحلول والمساهمات كانت مواضيع أوراق علمية تمّ نشرها في ملتقيات دولية و مجلات علمية محكّمة.

Author Biography

Said Gadri: Received his degree of engineer in computer science, field Hard&Software from the University Ferhat Abbas of Setif, Algeria in 1996, and the degree of magister (Bac + 7) from the University Mohamed Boudiaf of M'sila, Algeria in 2006. He is currently an assistant professor class A (permanent teacher) of computer science at the university Mohamed Boudiaf of M'sila, Algeria since 2007. Teaching different subjects like: Graph theory, Computer architecture, Databases, Web technology, programming languages (Pascal, C++, C#, Algorithmic), Expert systems, Modeling and simulation, Artificial Intelligence, Information Retrieval, etc. He is a member of the scientific council of mathematics and computer sciences faculty since 2009 to our days, member of the teaching committee of the ICST department (2013 – 2015). Currently, he is interested in many areas of research such as: Text categorization, machine learning, Text mining, Information retrieval, and natural language processing. Published more than 12 papers in different international conferences around the world.

Table of Content

Chapter 2: Fundamentals of text mining (41 – 70)

Chapter 3: Automatic Text Categorization (71 – 96)

Chapter 4: Machine Learning Algorithms for text categorization (97 – 140)

Chapter 5: Categorization of Semistructured documents (141 – 160)

Part II: Contribution Part

Chapter 6: Identifying the language of a text in a collection of multilingual documents

Chapter 7: Contextual Categorization Using New Panoply of Similarity Metrics

Chapter 8: Arabic Text Categorization: An Improved Stemming Algorithm to Increase the Quality of Categorization (180 – 188)

List of figures

List of tables

List of algorithms

List of examples

Introduction

1. Scope of the work

Automatic text categorization is a very important task in text mining. It is used in several applications and domains. It becomes more and more important as the amount of text in electronic format grows every day and its access becomes more necessary. This task is done by assigning labels (categories/classes) to documents based on a set of previous labelled documents[Sebastiani, 1999, Sebastiani, 2002]. Among the well-known applications we cite: spam filtering, assignation of web pages to yahoo directories (groups).[Nedjeh et al, 2009, Jackson and Moulinier, 2002] Thus, we can distinguish two kind of categorization; manual and automatic categorization. For example, web pages in yahoo directories can be assigned by human editors, so, we say that the classification was performed manually. On the other hand, users of outlook can write some simple rules to sort incoming e-mails into folders, or use predefined rules to delete junk e-mails. This is an example of automated text classification.

Some researchers make a distinction between text classification and text categorization. The term text categorization is sometimes used when assigning documents to a set of predefined set of categories. So, it's a kind of a supervised learning. However, the term text classification is used when categories are note predefined, but they will be created by the classifier during the classification process. So, it's a kind of unsupervised learning also called text clustering [Jackson and Moulinier, 2002]

Some other researchers use the term "text categorization" to designate the process of sorting documents by content, whereas the term "text classification" is used to perform any kind of assignment of documents to classes, not necessarily based on content, but any kind of sorting like: sorting by author, by publisher, or by language (English, French, Arabic, etc)[Jackson and Moulinier, 2002]. In the same context, the term classifier is used rather loosely to denote any process human or automatic or a mixture of the two which sorts documents to categories or subject matter labels.

In addition, current users are not always interested in accessing and manipulating information that are presented only in their native languages, but they tend more and more to take the plunge into other languages to meet their needs for relevant information. Hence, much importance is given to the processing of multilingual documents particularly the categorization task. This is justified by the growing number of documents from collections made in networks and shared across the world, the massive expansion of the Internet users and whose native languages are different [Lebart, 2001, Kodratoff, 2001, Jalam, 2003, Peter and Sheridan, 2001].

This had created the need to organize and process the large volume of multilingual textual data. Unfortunately, manual methods (human experts) are very costly in time and personnel[Jalam, 2003]. In addition, they are less fine stranded and their generalization to other areas is impossible. That is why we seek to implement more widespread automated methods [Moulinier, 1996, Sebastiani, 2002].

The categorization of multilingual texts is the extension of monolingual text categorization. It consists in assigning categories (classes/groups/ labels/themes) to texts written in different languages based on a predictive model that takes into account the variability of text languages. To build such a prediction model, there must be a set of texts previously labeledcalled "training set", on which we focus to estimate the parameters of the model that must be as powerful as possible, i.e., a model producing fewer errors in categorization [Jalam, 2003]. However, the definitive the goal is to categorize any new text as perfectly as possible.

2. Problematic

The main problematic of our thesis focusses on the exploitation of concepts and algorithms of machine learning in contextual categorization (thematic categorization) of multilingual texts. This problematic can be subdivided into the following sub-issues:

1. Design and implement a text classifier based on a prediction model that must be perfect as possible andcan support the variability of the language of input texts.
2. Propose effective solutions to improve the performance of the most critical phases of categorization include: the language identification phase and the selection of the most relevant terms phase (vocabulary reduction).
3. Apply the proposed solutions for the categorization of Arabic texts which generally pose enormous problems.
4. Improve the quality of the obtained categorization, including the categorization rate by exploiting linguistic and statistical techniques such as: stemming and the n-grams of characters techniques and statistical learning algorithms.

3. Our contribution

3.1. Language identification

Identifying the language of a text in a corpus of multilingual texts is an important and critical phase in the process of contextual categorization of multilingual documents. For this purpose, we studied the well-known algorithms, by comparing them, underlining their advantages and disadvantages, and especially proposing a simple, fast, and effective algorithm mainly inspired from Cavnar and Trenkle algorithm [Cavnar and Trenkle, 1994] and based on the technique of n-grams of characters [Gadri and Moussaoui, 2013, Gadri and Moussaoui, 2014b, Gadri and Moussaoui, 2014c].

3.2. Contextual text categorization

We have studied the problem of automatic supervised classification of documents (text categorization). Consequently, we proposed new panoply of similarity metrics which are often used in the field of language identification. We also propose a simple, optimal and effective method to improve the quality of categorization.[Gadri and Moussaoui, 2014a, Gadri and Moussaoui, 2014d].

3.3. Arabic Stemming

One of the methods used to reduce the size of terms vocabulary in Arabic text categorization is to replace the different variants (forms) of words by their common root. This process is called stemming based on the extraction of the root. Therefore, the search of the root in Arabic or Arabic word root extraction is more difficult than in other languages since the Arabic language has a very different and difficult structure, since it is a very rich language with complex morphology. Many algorithms are proposed in this field. Some of them are based on morphological rules and grammatical patterns, thus they are quite difficult and require deep linguistic knowledge. Others are statistical, so they are less difficult and based only on some calculations. In our work, we proposed an improved stemming algorithm based on the extraction of the root and the technique of n-grams which permit to return Arabic words' stems without using any morphological rules or grammatical patterns.[Gadri and Moussaoui, 2015a, Gadri and Moussaoui, 2015b]

3.4. Multilingual Stemming

Stemming is a very important step of pre-processing in text mining, and generally used in many areas of research such as: Natural language Processing NLP, Text Categorization TC, Text Summarizing TS, Information Retrieval IR, and other tasks in text mining. Stemming is useful in text categorization to reduce the size of terms vocabulary, and in information retrieval to improve the search effectiveness and then gives us relevant results.Severalmonolingual stemmers are developed for various languages as English, French, German and Arabic, but each one has its own advantages as well as limitations. Most of the stemming algorithms used in this field are language dependent. So, it is important to develop a new stemmer which is language independent. In this way, we proposed in our work a new multilingual stemmer based on the extraction of the word root, as well as the use of n-grams technique. The proposed stemmer was tested on three languages, namely: Arabic, French, and English that gave promising results[Gadri and Moussaoui, 2015c, Gadri and Moussaoui, 2015d, Gadri and Moussaoui, 2015e].

4. Thesis organization

Our thesis consists of two main parts:

4.1. A theoretical part

It consists of five chapters:

- Chapter 1: explores the techniques of knowledge discovery from data (KDD) and its various forms. It presents an overall picture of the data mining field by introducing its concepts techniques, tools, applications, and research directions.
- Chapter 2: in this chapter, we try to discover an interesting subfield of data mining which is text mining by presenting its definition, domains of application, used tools and techniques, different tasks and phases, others.
- Chapter 3: treats the following subjects: the definition of categorization process, description of their common applications, some difficulties of TC, monolingual TC Vs multilingual TC, the problem of text presentation and coding, language identification.
- Chapter 4: presents an overview of the main machine learning algorithms which are mostly used in text categorization in particular. It explains the principle of each algorithm, its advantages and weaknesses, as well as a short pseudo-code and some illustrative examples.
- Chapter 5: presents briefly semi-structured documents in particular XML documents giving a global overall on XML language and the semantic of its tags, the notion of XML mining and its relationship with other types of mining (data mining, text mining, web mining).

4.2. Contributions part

It consists of three chapters:

- Chapter 6: treats the problem of language identification of texts in a multilingual textual corpus. It explains the two most known approaches in the field to segment a text into basic units, including: “bag of words” and “n-grams” approaches, and presents our contribution to solve this problem.
- Chapter 7: In this chapter, we study the supervised classification. We use a k -NN approach based on similarity metrics known in the field of language identification. We also propose a new simple and effective method to improve the results of categorization.
- Chapter 8: In this chapter, we propose an improved statistical algorithm which permits to build an Arabic stemmer based on the extraction of words’ roots and the approach of n-grams of characters without using any morphological rule.

5. Author Publications

During the realization of this research, several papers have been published in international conferences, namely:

A. Conferences 2013

1. Said K., Abdelouahab M. : Une méthode flexible pour l'identification de la langue d'un texte dans un corpus hétérogène multilingue, 2ème Conférence nationale des études doctorales en Informatique CNEDI2013, Université de Sekikda, 20-21 mai 2013.
2. Said G., Abdelouahab M. :An Effective Method to Recognize the Language of a Text in a collection of Multilingual Documents, 10th International Conference on Electronics, Computer and Computation ICECCO 2013, **IEEE Conference**, Turgut Ozal University, Ankara, Turkey, 07-08 November 2013.
   IEEE Explorer Library:
   http://ieeexplore.ieee.org/search/searchresult.jsp?newsearch=true&searchWithin=%22First%20Name%22:Said&searchWithin=%22Last%20Name%22:kadri

B. Conferences 2014

3. Said K., Abdelouahab M. : Utilisation des Métriques de l'Identification de la Langue dans la Catégorisation Contextuelle de Documents, 1st International Symposium on Informatics and its Applications, ISIA 2014, University of M'sila, M'sila, Algeria, 25-26February,2014.
   Conference Link :http://www.univ-msila.dz/ISIA14/
4. Said G., Abdelouahab M., Linda B-F. : Language Identification: Proposition of a New Optimized Variant for the Method of Cavenar and Trenkle, International Conference on Artificial Intelligence and Information Technology ICAIIT'14, University KasdiMerbah, Ouargla, Algeria, 10-12 Mars 2014, Conference Link : http://www.icaiit2014.com
5. Said G., Abdelouahab M., Linda B-F. : Language Identification: A New Fast Algorithm to Identify the Language of a Text in a Multilingual Corpus, The 4thInternational Conference on Multimedia Computing and Systems ICMCS'14, **IEEE Conference**, University of Marrakesh, Marrakesh, Morocco, 14-16 Avril 2014.
   IEEE Explorer Library:
   http://ieeexplore.ieee.org/search/searchresult.jsp?newsearch=true&queryText=Said%20Gadri&fname=&mname=&lname=&title=&volume=&issue=&spage=
6. Said G., Abdelouahab M.: Contextual Categorization of Documents Using a New Panoply of Similarity Metrics, International Conference on Advanced Technology & Sciences (ICAT'14), 12-15 August, 2014, Antalya, Turkey,
   Conference Link :http://conference.atscience.org/

C. Conferences 2015

7. Said G., Abdelouahab M.: An Effective Multilingual Stemmer Based on the Extraction of the Root and the N-grams Technique, AIST'2015 international scientific conference Analysis of Images, Social networks, and Texts, 9-11th, April, Yekaterinburg, Russia.
   Conference Link: http://aistconf.org/

8. Said G., Abdelouahab M.: A new Multilingual Stemmer Based on the Extraction of the Root and the N-grams Technique, ICIPCE'2015 International Conference on Information Processing and Control Engineering, 15-17th, April, Moscow, Russia.
   Conference link: http://www.icipce2015.org
9. Said G., Abdelouahab M.,: Arabic Texts Categorization: Features Selection Based on the Extraction of Words' Roots, 5th IFIP International Conference on Computer Science and its Applications (CIIA'2015), 20, 21 May 2015, **Springer Conference**, TaharMoulay University, Saida - Algeria.
   Conference Link :http://ciia2015.ensma.fr/
   Springer Link: http://link.springer.com/search?query=said+gadri&search-within=Book&facet-book-doi=10.1007%2F978-3-319-19578-0
   La base DBLP: http://dblp.uni-trier.de/search?q=Said+Gadri
10. Said G., AbdelouahabM.,:Information Retrieval: A New Multilingual Stemmer Based on a Statistical Approach, 3rd International Conference on control, engineering & information Technology (CEIT2015), 25-27 may, **IEEE Conference**, Tlemcen, Algeria.
    IEEE Explorer Library:
    http://ieeexplore.ieee.org/search/searchresult.jsp?newsearch=true&queryText=Said%20Gadri&fname=&mname=&lname=&title=&volume=&issue=&spage=
11. Said G., Abdelouahab M.,: Multilingual information retrieval: increasing the effectiveness of search by stemming, 19th International Conference onCircuits, Systems, Communications and Computers (CSCC 2015), July 16-20, 2015, Zakynthos Island, Greece.
    Conference Link: http://2015.cscc.co/
12. Said G., Abdelouahab M.,: Multilingual Text Categorization: Increasing the Quality of Categorization by a Statistical Stemming Approach, International Conference on intelligent Information Processing Security and Advanced Communication (IPAC 2015), **ACM Conference**, November 23-25, 2015 Batna, Algeria.
    Conference Link: http://ipac.awict.net/Accepted_Papers.html.

Other papers have been published in estimated journals such as:

13. Said G., Abdelouahab M., : Arabic Text Categorization: An Improved Algorithm Based on N-grams technique to Extract Arabic Words Roots, International Arab Journal of Information Technology IAJIT (ISI Thomson, IF: 0.582, Jordan), Vol.15, No.02, March 2018.
14. Said G., Abdelouahab M.,: Application of a New Set of Pseudo-Distances in Document Categorization, Neural Network World NNW ((ISI Thomson, IF: 0.2, Czech Republic), accepted (in correction phase).

Theoretical Part

CHAPTER 1

Data Mining: Basic Concepts and Tools

Chapter Content

1.1. Introduction

Our capacities of both generating and collecting data have been increasing rapidly. Contributing factors include the computerization of business, scientific, and government transactions; the widespread use of digital cameras, publication tools, and bar codes for most commercial products; and advances in data collection tools ranging from scanned text and image platforms to satellite remote sensing systems. In addition, large use of the World Wide Web as a global information system has flooded us with a tremendous amount of data and information. This explosive growth in stored or transient data has generated an urgent need for new techniques and automated tools that can intelligently assist us in transforming the vast amounts of data into useful information and knowledge. [Han and Kamber, 2006, Pang_Ning et al., 2013, Witten and Frank, 2005]

Data mining refers to *extracting or "mining" knowledge from large amounts of data,* like mining of gold from rocks or sand is referred to as *gold* mining rather than rock or sand mining [Han and Kamber, 2006]. Thus, data mining should have been more appropriately named "knowledge mining from data," which is unfortunately somewhat long. "Knowledge mining," a shorter term may not reflect the emphasis on mining from large amounts of data [Han and Kamber, 2006]. Nevertheless, mining is a vivid term characterizing the process that finds a small set of precious nuggets from a great deal of raw material. Thus, such a misnomer that carries both "data" and "mining" became a popular choice. Many other terms carry a similar or slightly different meaning to data mining, such as knowledge mining from data, knowledge extraction, data/pattern analysis, data archeology, and data dredging. Many people treat data mining as a synonym for another popularly used term, Knowledge Discovery from Data, or KDD. Alternatively, others view data mining as simply an essential step in the process of knowledge discovery [Han and Kamber, 2006, Kantardzic, 2003].

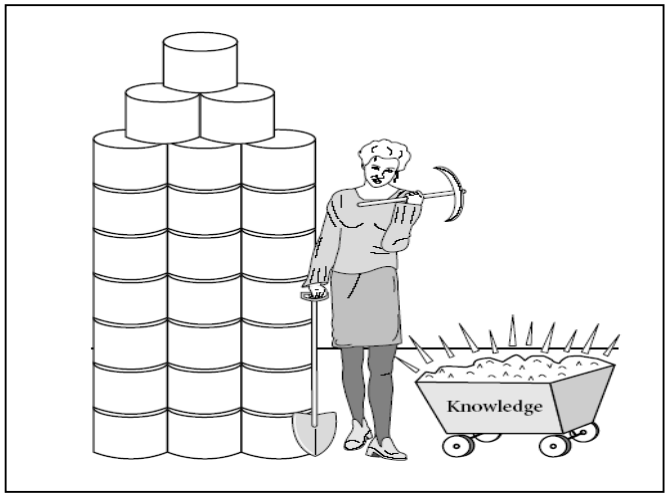


Figure 1.1 Data Mining: Searching for knowledge in large data. [Han and Kamber, 2006]

Some specialists define data mining as a multidisciplinary field, drawing work from areas including database technology, machine learning, statistics, pattern recognition, information

retrieval, neural networks, knowledge-based systems, artificial intelligence, high-performance computing, and data visualization [Kumar, 2007]. Data mining emerged during the late 1980s, made great strides during the 1990s, and continues to flourish into the new millennium [Steinbach et al., 2004].

This chapter explores the concepts and techniques of data mining, a flourishing frontier between data mining and information systems and their applications, knowledge discovery from data (KDD) and its various forms, techniques and technologies for the discovery of patterns hidden in large data sets, focusing on issues relating to their feasibility, usefulness, effectiveness, and scalability, the life cycle of a typical data mining project. So, this chapter presents an overall picture of the field, introducing interesting data mining techniques and systems and discussing applications and research directions.

1.2. Knowledge data discovery from data

Is the automated or convenient extraction of patterns representing knowledge implicitly stored or captured in large databases, data warehouses, the Web, other massive information repositories, or data streams? [Han and Kamber, 2006]

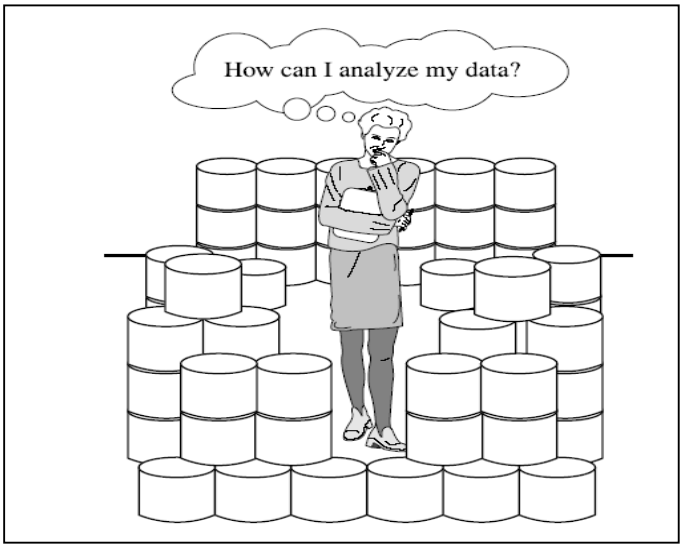


Figure 1.2. We are data rich, but knowledge poor [Han and Kamber, 2006]

1.3. Some popular definitions of Data Mining

Idea: The main idea of Data Mining is: mountains of data – where knowledge is mined.

We can also give many definitions of data mining such as:

- Non-trivial extraction of implicit, previously unknown and potentially useful information from data [Paulheim, 2015]
- Exploration & analysis, by automatic or semiautomatic means, of large quantities of data in order to discover meaningful patterns. [Kumar, 2007]

- Data Mining is a non-trivial process of identifying–valid–novel–potentially useful–ultimately understandable patterns in data. [Fayyad et al, 1996a, Agrawal and Srikant, 2000]
- Data mining is nothing else than torturing the data until it confesses...and if you torture it enough, you can get it to confess to anything [Steinbach et al., 2004].
- Data mining is a multidisciplinary field, drawing work from areas including database technology, machine learning, statistics, pattern recognition, information retrieval, neural networks, knowledge-based systems, artificial intelligence, high-performance computing, and data visualization. [Han and Kamber, 2006].
- Data mining is the process of automatically discovering useful information in large data repositories. Data mining techniques are deployed to scour large databases in order to find novel and useful patterns that might otherwise remain unknown [Pang-Ning et al, 2013].

1.4. What is Data Mining? An introductive Example

Let's begin with these two questions: why can't we dig out the knowledge by using SQL queries? In other words, what are the fundamental differences between data mining and relational database technologies? Let's have a look of the following example: [Zaki and Wagner, 2014]

Figure 1.3 displays a relational table containing a list of high school graduates. The table records information are: *gender*, *IQ*, *the level of parental encouragement*, and the *parental income* of each student along with that student's intention to attend college.

If someone asks you a question: ***What drives high school graduates to attend college?***

You may write a query to find out how many male students attend college versus how many female students do. You may also write a query to determine the impact of the Parent Encouragement column. But what about male students who are encouraged by their parents? Or female students who are not encouraged by their parents? You would need to write hundreds of these queries to cover all the possible combinations. Data in numerical forms, such as that in *Parent Income* or *IQ*, is even more difficult to analyze. You would need to choose arbitrary ranges in these numeric values. What if there are hundreds of columns in your table? You would quickly end up with an impossible to manage number of SQL queries to answer a basic question about the meaning of your data.

In contrast, the data mining approach to this question is rather simple. All you need to do is select the right data mining algorithm and specify the column usage, meaning the input columns and the predictable columns (which are the targets for the analysis). A decision tree model would work well to determine the importance of parental encouragement in a student's decision to continue to college. You would select *IQ, Gender, Parent Income*, and *Parent Encouragement* as the input columns and *College Plans* as the predictable column.

As the decision tree algorithm scans the data, it analyzes the impact of each input attribute related to the target and selects the most significant attribute to split. Each split divides the dataset into two subsets so that the value distribution of College Plans is as different as possible among these two subsets. This process is repeated recursively on each subset until

the tree is completely built. Once the training process is complete, you can view the discovered patterns by browsing the tree.

Table 1.1. List of high school graduates

Gender	ParentIncome	IQ	ParentEncouragement	CollegePlans
Male	46580	100	Not Encouraged	No
Male	39687	121	Not Encouraged	No
Male	63482	102	Encouraged	Yes
Female	40454	129	Not Encouraged	No
Male	7333	86	Not Encouraged	No
female	17617	105	Not Encouraged	No
Male	33540	110	Not Encouraged	No
Male	48171	102	Not Encouraged	Yes
Male	33656	79	Not Encouraged	No
Male	73325	120	Encouraged	yes
Male	33153	112	Not Encouraged	No
Male	10331	94	Not Encouraged	No
Female	33505	106	Not Encouraged	Yes
Female	30052	76	Encouraged	Yes
Male	24579	105	Not Encouraged	No
Male	37497	72	Not Encouraged	No
Male	31572	98	Encouraged	No
Female	41979	138	Not Encouraged	No
Female	11151	61	Not Encouraged	No
Female	9532	86	Encouraged	No
Male	73580	124	Encouraged	Yes
female	70149	104	Encouraged	No
Male	44316	122	Encouraged	Yes
Male	14915	100	Not Encouraged	No
Male	52417	68	Encouraged	No

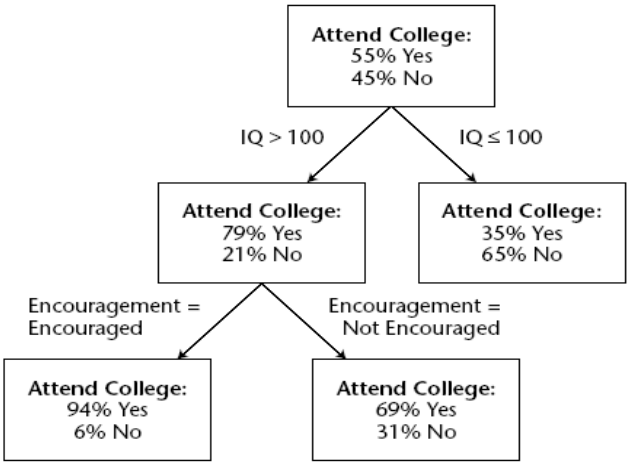


Figure 1.3. Student's decision tree

1.5. What is not Data Mining

It's important to say that not all tasks that we can do are considered as data mining tasks. In the following table we give some examples of tasks that require data mining techniques and some others tasks that don't require any of these techniques [Steinbach et al., 2004].

Table 1.2. What is not Data Mining and what is Data Mining.

What is not Data Mining	What is Data Mining
1. Look up phone number in phone directory 2. Query a Web search engine for information about "Amazon"	1. Certain names are more prevalent in certain US locations (O'Brien, O'Rurke, O'Reilly… in Boston area) **2.** Group together similar documents returned by search engine according to their context (e.g. Amazon rainforest, Amazon.com,)

1.6. Data mining and knowledge Discovery

Data mining is an integral part of knowledge discovery in databases (KDD), which is the overall process of converting raw data into useful information, as shown in figure 1.4 [Pang-Ning et al., 2013]. This process consists of a series of transformation steps, from data preprocessing to post processing of data mining results. The input data can be stored in a variety of formats (flat files, spreadsheets, or relational tables) and may reside in a centralized data repository or be distributed across multiple sites [Zytkow and Rauch, 1999, Kumar, 2007].

Figure 1.4. The process of knowledge discovery in databases KDD

It's important to note that the preprocessing step is the more interesting step in a KDD process. The purpose of this step is to transform the raw input data into an appropriate format for subsequent analysis. The steps involved in data preprocessing include fusing data from multiple sources, cleaning data to remove noise and duplicate observations, and selecting records and features that are relevant to the data mining task. Because of the many ways data can be collected and stored, data preprocessing is perhaps the time-consuming step in the overall knowledge discovery process. [Fayyad et al., 1996b, Bhandari et al., 1997, Pang-Ning et al., 2013]

1.7. Where Data Mining can be placed? Origins

Although data mining as a term is relatively new, most data mining techniques have existed for years. If we look at the roots of those popular data mining algorithms, we find that they are mainly derived from different disciplines, notably: statistics, machine learning, pattern recognition and database [Steinbach et al., 2004, Pang Ning et al., 2013, Paulheim, 2015].

1. ***Statistics:*** Most of data mining tasks listed in the previous section have been addressed in the statistics community. A number of data mining algorithms, including regression, time series, and decision trees, were invented by statisticians. Regression techniques have

existed for centuries. Time series algorithms have been studied for decades. The decision tree algorithm is one of the more recent techniques, dating from the mid-1980s. [Freidman, 1979, Glymour et al., 1997, Lambert, 2000]

2. ***Machine Learning:*** Data mining focuses on automatic or semiautomatic pattern discovery. Several machine learning algorithms have been applied to data mining. Neural networks are one of these techniques and are excellent for classification and regression, especially when the attribute relationships are nonlinear. The genetic algorithm is yet another machine learning technique. It simulates the natural evolution process by working with a set of candidates and a survival (fitness) function. The survival function repeatedly selects the most suitable candidates for the next generation. Genetic algorithms can be used for classification and clustering tasks. They can also be used in conjunction with other algorithms, for instance, helping a neural network to find the best set of weights among neurons [Mitchell, 1997, Cherkassky and Mulier, 1998, Hastie et al., 2001, Witten and Frank, 2005]
3. ***Database:*** A database is the third technical source for data mining. Traditional statistics assumes that all the data can be loaded into memory for statistical analysis. Unfortunately, this is not always the case in the modern world. Database experts know how to handle large amounts of data that do not fit in memory, for example, finding association rules in a fact table containing millions of sales transactions. As a matter of fact, the most efficient association algorithms come from the database research community. There are also a few scalable versions of classification and clustering algorithms that use database techniques, including the Microsoft clustering algorithm [Chen et al., 1996].

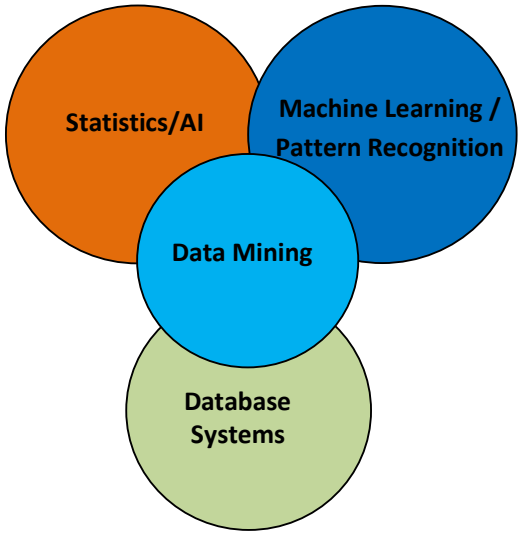


Figure 1.5. Origins of Data Mining

Data mining has also been quick to adopt ideas from other areas, including: optimization, evolutionary computing, information theory, signal processing, visualization, and information retrieval. Techniques from high performance (parallel) computing are often important in

addressing the massive size of some data sets. Distributed techniques can also help address the issue of size and are essential when the data cannot be gathered in one location. [Pang-Ning et al, 2013]

1.8. Data mining motivations

Data mining provides a lot of business value for enterprises. So, the main question that can be asked is why are we interested in data mining now? The following are a number of reasons: [Wu et al., 2000, Kumar, 2007, Pang_Ning et al., 2013, Zaki and Wagner, 2014, Paulheim, 2015]

1. ***Lots of data is being collected and warehoused:*** Over the last decade, the price of hardware, especially hard disk space, has dropped dramatically. In conjunction with this, enterprises have gathered huge amounts of data through many applications, such as:
    - Transaction data from banking, telecommunication, e-commerce
    - Scientific data from astronomy, physics, biology, computer science
    - Remote sensors on a satellite.
    - Telescopes scanning the skies.
    - DNA sequencing robots churning out new genomes.
    - The public Web, Twitter, Facebook, …
    - Social network sites.
    - Application logs.

According to the available statistics given by many specialized organizations, the amount of information in the world doubles every 20 Months. They are often collected and stored at enormous speeds (GB/hour). As examples of such data, we can give the following:

- Wikipedia ≈ 5.9 Tb (jan 2010 Dump)
- Human genome ≈ 4 Gb/person.
- US Library of congress ≈ 235 Tb archived
- Sloan Digital sky survey ≈ 200 Gb/day ≈ 73 Tb/year
- NASA center for climate simulation ≈ 32 Pb archived.
- Facebook ≈ 12 Tb/day added ≈ 4380 Tb/year (Mar 2010)
- Google ≈ 20 Pb/day processed ≈ 7300 Pb/year (Jan 2010)
- Large Hadron Collider ≈ 15 Pb/year
- Ineternet (2016) ≈ 1.3 Zb/year (2016 IP traffic; CISCO EST)

With all of this data to explore, enterprises want to be able to find hidden patterns to help guide their business strategies.

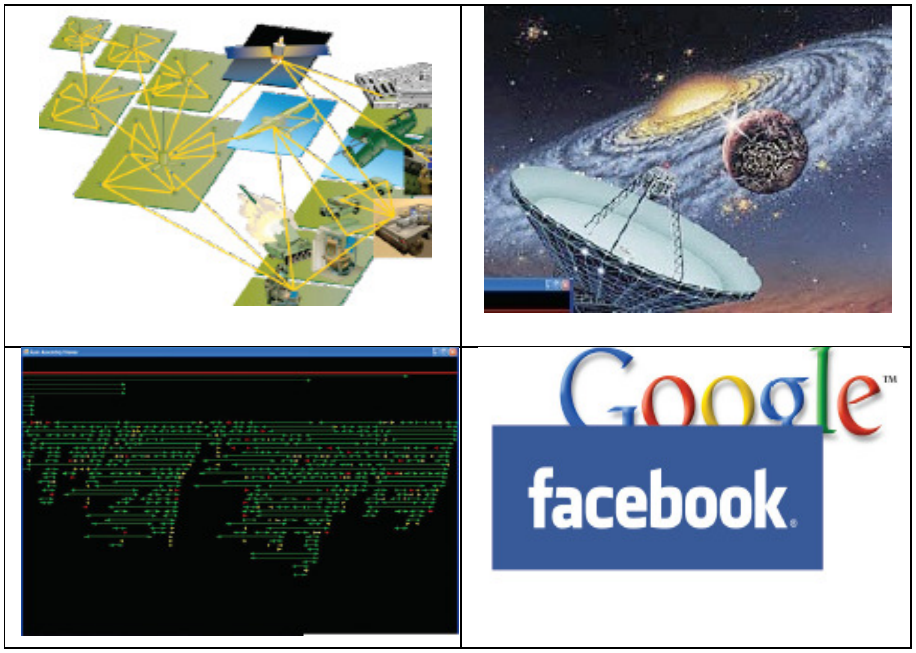


Figure 1.6. Examples of collected data.

2. ***High dimensionality:*** Now, it is common to process data sets with hundreds or thousands of attributes which was not possible a few decades ago. In bioinformatics, progress in microarray technology has produced gene expression data involving thousands of features. Data sets with temporal or spatial components also tend to have high dimensionality. For example, consider a data set that contains measurements of temperature at various locations. If the temperature measurements are taken repeatedly for an extended period, the number of dimensions (features) increases in proportion to the number of measurements taken. Traditional data analysis techniques that were developed for low-dimensional data often do not work well for such high dimensional data. Also, for some data analysis algorithms, the computational complexity increases rapidly as the dimensionality (the number of features) increases.

3. ***Heterogeneous and Complex Data:*** Traditional data analysis methods often deal with small data sets containing attributes of the same type. As data mining treats a big amount of complex data, so it has the need for techniques that can handle heterogeneous attributes and take into consideration relationships in the data, such as temporal and spatial autocorrelation, graph connectivity, and parent-child relationships between the elements in semi-structured text and XML documents. Examples of such non-traditional types of data include collections of Web pages containing semi-structured text and hyperlinks; DNA data with sequential and three-dimensional structure; and climate data that consists of time series measurements {temperature, pressure, etc.) at various locations on the Earth's surface.

4. ***Scalability:*** Because of the huge sizes of generated and collected data, big data sets (gigabytes, terabytes, petabytes) are becoming common. If data mining algorithms are

developed to to handle these massive data sets, then they must be scalable using efficient strategies. Scalability may also require the implementation of novel data structures to access individual records efficiently. For instance, out-of-core algorithms may be necessary to process data sets that cannot fit into main memory. These algorithms need efficient index structures to exploit the secondary memory. Scalability can also be improved by using sampling or developing parallel and distributed algorithms.

5. ***Data Ownership and Distribution***: Sometimes, the data needed for an analysis is not stored in one location or owned by one organization. Instead, the data is geographically distributed among resources belonging to multiple entities. This requires the development of distributed data mining techniques. Among the key challenges faced by distributed data mining algorithms include (1) how to reduce the amount of communication needed to perform the distributed computation, (2) how to effectively consolidate the data mining results obtained from multiple sources, and (3) how to address data security issues. [Pang-Ning et al., 2013].

6. ***Non-traditional analysis:*** The traditional statistical approach is based on a hypothesize-and-test paradigm. ln other words, a hypothesis is proposed, an experiment is designed to gather the data, and then the data is analyzed with respect to the hypothesis. Unfortunately, this process is extremely labor-intensive. Current data analysis tasks often require the generation and evaluation of thousands of hypotheses, and consequently, the development of some data mining techniques has been motivated by the desire to automate the process of hypothesis generation and evaluation. Furthermore, the data sets analyzed in data mining are typically not the result of a carefully designed experiment and often represent opportunistic samples of the data, rat her than random samples. Also, the data sets frequently involve non-traditional types of data and data distributions.

7. ***Competitive Pressure is Strong:*** Competition is high as a result of modern marketing and distribution channels such as the Internet and telecommunications. Enterprises are facing worldwide competition, and the key to business success is the ability to retain existing customers and acquire new ones. Data mining contains technologies that allow enterprises to analyze factors that affect these issues.

8. ***Technology ready:*** Data mining technologies previously existed only in the academic sphere, but now many of these technologies have matured and are ready to be applied in industry. Algorithms are more accurate, more efficient and can handle increasingly complicated data. In addition, data mining application programming interfaces (APIs) are being standardized, which will allow developers to build better data mining applications.

We can conclude that we use data mining when traditional techniques may be unsuitable due to: Enormity of data, High dimensionality of data, Heterogeneity, distributed nature of data and other reasons.

1.9. Data mining Tasks

Data mining tasks are generally divided into two major categories: [Pang_Ning et al., 2013, Paulheim, 2015]

1.9.1. Predictive Tasks: the objective of these tasks is to predict the value of a particular attribute based on the values of other attributes. The attribute to be predicted is called the target attribute (dependent variable), while the attributes used for making the prediction are known as the explanatory attributes (independent variables). As examples of predictive tasks we can note:

- Classification: predict the value discrete target variable with typically few values, often binary variable
- Regression: predict the value of a continuous target variable
- Anomaly/novelty detection: predict a deviation from normal/current state of affairs

1.9.2. **Descriptive tasks:** the objective of these tasks is to derive patterns (correlations, trends, clusters, trajectories, and anomalies) that summarize relationships in data. Descriptive tasks often require postprocessing techniques to validate and explain the results. As examples of descriptive tasks, we can note the following:

- Clustering: divide the data into internally coherent groups.
- Frequent pattern discovery: find combinations of variables' values that occur more frequently than expected by chance

We note also that in machine learning terminology; descriptive means unsupervised and predictive means supervised.

1.10. A classic example of data mining use

1. SARS disease [Paulheim, 2015]

- SARS severe acute respiratory syndrome.
- Outbreak: 2012 in Hong Kong

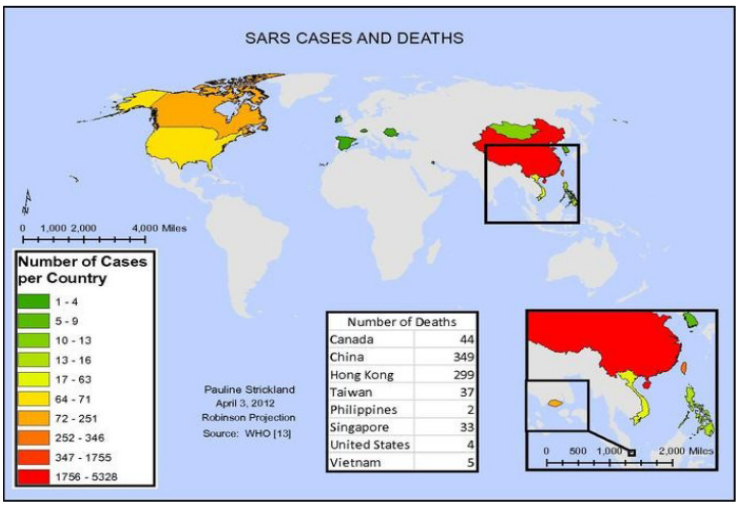


Figure 1.7. SARS cases end death around the world (in 2012)

Which paths does SARS infection take? According to the Max Planck Institute for Dynamics and Self-Organization: - SARS infections follow major airline routes.

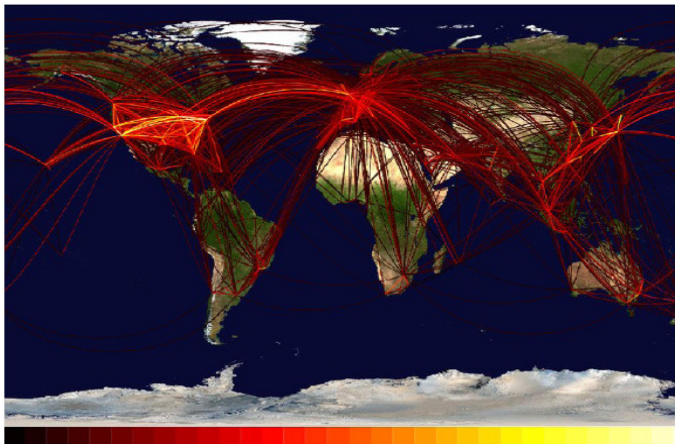


Figure 1.8. Airline paths of SARS infection around the world (in 2012)

1.11. Data mining applications

1.11.1. Business Problems

Data mining techniques can be applied to many applications, answering various types of businesses questions. The following list illustrates a few typical problems that can be solved using data mining: [Berry and Linoff, 2004, Sumathi and Sivanandam, 2006, Ranka, 2011, Hofmann and Klinkenberg, 2013, Pang_Ning et al., 2013, Zaki and Wagner, 2014].

Churn analysis: Which customers are most likely to switch to a competitor? The telecom, banking, and insurance industries are facing severe competition these days. On average, each new mobile phone subscriber costs phone companies over 200 dollars in marketing investment. Every business would like to retain as many customers as possible. Churn analysis can help marketing managers understand the reason for customer churn, improve customer relations, and eventually increase customer loyalty.

Cross-selling: What products are customers likely to purchase? Cross-selling is an important business challenge for retailers. Many retailers, especially online retailers, use this feature to increase their sales. For example, if you go to online bookstores such as Amazon.com or Barnes and Noble.com to purchase a book, you may notice that the Web site gives you a set of recommendations about related books. These recommendations can be derived from data mining analysis.

Fraud detection: Is this insurance claim fraudulent? Insurance companies process thousands of claims a day. It is impossible for them to investigate each case. Data mining can help to identify those claims that are more likely to be false.

Risk management: Should the loan be approved for this customer? This is the most common question in the banking scenario. Data mining techniques can be used to score the customer's risk level, helping the manager make an appropriate decision for each application.

Customer segmentation: Who are my customers? Customer segmentation helps marketing managers understand the different profiles of customers and take appropriate marketing actions based on the segments.

Targeted ads: What banner ads should be displayed to a specific visitor? Web retailers and portal sites like to personalize their content for their Web customers. Using customers' navigation or online purchase patterns, these sites can use data mining solutions to display targeted advertisements to their customers' navigators.

Sales forecast: How many cases of wines will I sell next week in this store? What will the inventory level be in one month? Data mining forecasting techniques can be used to answer these types of time-related questions.

1.11.2. Other problems for data mining

Science problems: Data mining helps scientists to analyze data and to formulate hypotheses. Astronomy, physics, bioinformatics, drug discovery, etc are examples of analyzed scientific data.

Web and Social Media: advertising, search engine optimization, spam detection, web site optimization, personalization, sentiment analysis, …

Government: surveillance, crime detection, profiling tax cheaters, etc.

1.12. Principle tasks of Data Mining

As we have seen previously, data mining can be used to solve hundreds of business problems. Based on the nature of these problems, we can group them into the following data mining tasks.

1.12.1. Classification (predictive task): Classification is one of the most popular data mining tasks. Business problems like churn analysis, risk management and ad targeting usually involve classification. Classification refers to assigning cases into categories based on a predictable attribute. Each case contains a set of attributes, one of which is the class attribute (predictable attribute). The task requires finding a model that describes the class attribute as a function of input attributes. In the *College Plans* dataset previously described, the class is the College Plans attribute with two states: *Yes* and *No*. To train a classification model, you need to know the class value of input cases in the training dataset, which are usually the historical data. Data mining algorithms that require a target to learn against are considered supervised algorithms. Typical classification algorithms include decision trees, neural network, and Naïve Bayes.

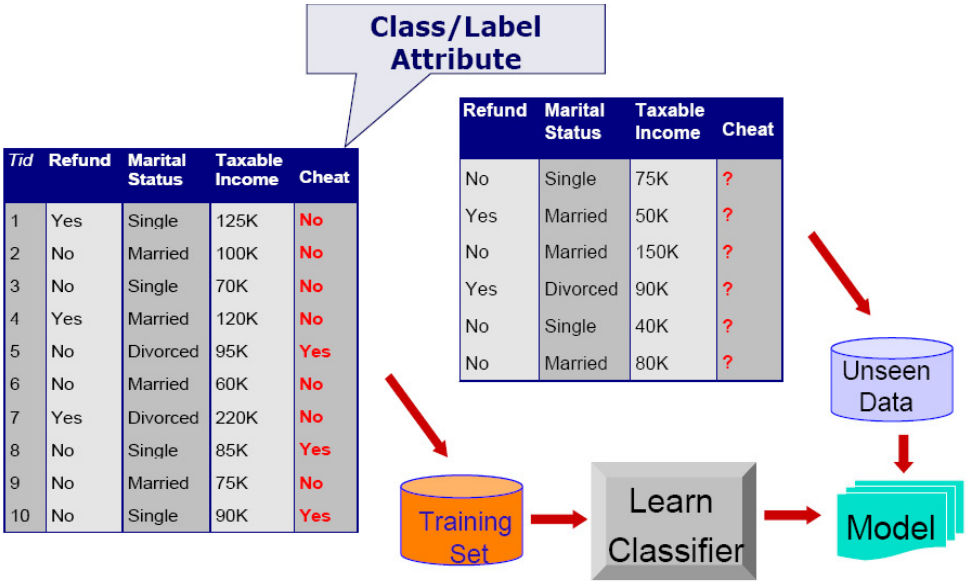


Figure 1.9. An example of classification problem.

Some classification applications

1. Direct Marketing

- Goal: Reduce cost of mailing by targeting a set of consumers which are likely to buy a new cell phone.
- Approach:
  - Use the data for a similar product introduced before.
  - We know which customers decided to buy and which did not. This (buy, don't buy) decision forms the class attribute.
  - Collect various demographic, lifestyle, and company-interaction related information about all such customers
    e.g., Type of business, where they stay, how much they earn, etc.
  - Use this information as input attributes to learn a classifier model.

2. Credit card fraud detection

- Goal: detect fraudulent use of credit card (card and/or card details stolen and misused)
- Approach:
  - Database of credit card purchase transactions (date, place, store details, card holder details, price of purchased item, transaction history of the credit card, …)
  - Transactions labeled as "normal" vs. "fraudulent"
  - Learn a classifier that predicts from the transaction data the correct label.
  - Use this model to detect fraud by observing credit card transactions on an account
  - Many methods: Decision trees, Support vector machine, Nearest neighbor classifier, Naïve Bayes.

Figure 1.10. Credit card fraud detection using Data Mining techniques

We can summarize clustering problem as follows:

- Given a collection of records (training set)
  - each record contains a set of attributes
  - one of the attributes is the class (label) that should be predicted
- Find a model for class attribute as a function of the values of other attributes
- Goal: previously unseen records should be assigned a class as accurately as possible
  - A test set is used to validate the accuracy of the model
  - Usually, the given data set is divided into training and test sets, with training set is used to build the model and the test set to validate it.

1.12.2. Clustering (descriptive Task): Clustering is also called segmentation. It is used to identify natural groupings of cases based on a set of attributes. Cases within the same group have more or less similar attribute values. Figure 1.11 displays a simple customer dataset containing two attributes: age and income. The clustering algorithm groups the dataset into three segments based on these two attributes. Cluster 1 contains the younger population with a low income. Cluster 2 contains middle-aged customers with higher incomes. Cluster 3 is a group of senior individuals with a relatively low income. Clustering is an unsupervised data mining task. No single attribute is used to guide the training process. All input attributes are treated equally. Most clustering algorithms build the model through a number of iterations and stop when the model converges, that is, when the boundaries of these segments are stabilized.

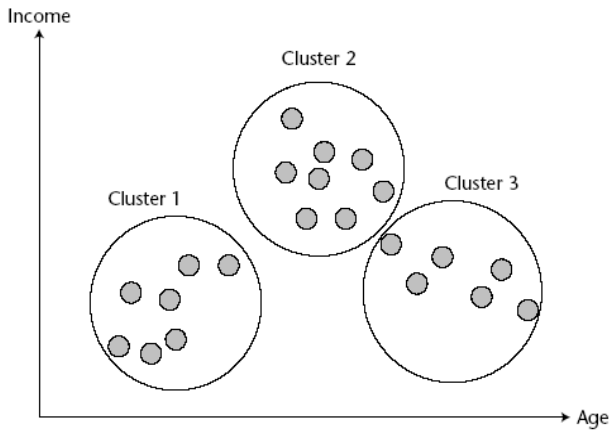


Figure 1.11. Principle of clustering problem.

Some clustering applications

1. Market segmentation [Berry and Linoff, 2004]

- Goal: Subdivide a market into distinct subsets of customers.
  - where any subset may be conceived as a marketing target to be reached with a distinct marketing mix
- Approach:
  - Collect information about customers.
  - Find clusters of similar customers.
  - Measure the clustering quality by observing buying patterns of customers in same cluster vs. those from different clusters.

Figure 1.12. Market segmentation problem.

2. ***Document Clustering:*** we have a collection of news articles (see table 1.3 bellow). These articles can be grouped according to their respective topics. Each article is represented as a set of word-frequency pairs (*w, f*), where *w* is a word and *f* is the frequency of the word (the number of times the word appears in the article). A correct clustering must give two clusters; the first one consists of the first four articles and corresponds to news about

economy, while the second cluster contains the last four articles and corresponds to news about health care [Pang_Ning et al., 2013].

A good clustering algorithm should be able to identify these two clusters based on the similarity between words that appear in the articles.

Table 1.3. Clustering of a collection of news articles

Articles	Words	Cluster
1	dollar: 1, industry: 4, country: 2, loan: 3, deal: 2, government: 2	Economy
2	machinery: 2, labor: 3, market: 4, industry: 2, work: 3, country: 1	Economy
3	job: 5, inAat.ion: 3, rise: 2, jobless: 2, market: 3, country: 2, index: 3	Economy
4	domestic: 3, forecast: 2, gain: 1, market: 2, sale: 3, price: 2	Economy
5	patient: 4, symptom: 2, drug: 3, health: 2, clinic: 2, doctor: 2	Health
6	pharmaceutical: 2, company: 3, drug: 2, vaccine: 1, flu: 3	Health
7	death: 2, cancer: 4, drug: 3, public: 4, heal th: 3, director: 2	Health
8	medical: 2, cost: 3, increase: 2, patient: 2, health: 3, care: 1	Health

We can summarize clustering problem as follows:

- Given a set of data points, and a similarity measure among them, find clusters such that:
  - Data points in one cluster are similar to one another.
  - Data points in separate clusters are different from each other result.
  - A descriptive grouping of data points.
- Similarity measures
  - Euclidean distance if attributes are continuous.
  - Other specific measures for other clustering problems

1.12.3. Association Rule Mining (descriptive task): Association is another popular data mining task. Association is also called market basket analysis. A typical association business problem is to analyze a sales transaction table and identify those products often sold in the same shopping basket (together). The common usage of association is to identify common sets of items (frequent itemsets) and rules for the purpose of cross-selling (see Figure 1.13). In terms of association, each product, or more generally, each (attribute/value) pair is considered an item. The association task has two goals: to find frequent itemsets and to find association rules. Most association type algorithms find frequent itemsets by scanning the dataset multiple times. The frequency threshold (support) is defined by the user before processing the model. For example, support = 2% means that the model analyzes only items that appear in at least 2% of shopping carts. A frequent itemset may look like {Product = “Pepsi”, Product = “Chips”, Product = “Juice”}. Each itemset has a size, which is the number of items that it contains. For example, the size of this particular itemset is 3. Apart from identifying frequent itemsets based on support, most association type algorithms also find rules. An association rule has the form A, B => C with a probability, where A, B, C are all frequent item sets. The probability is also referred to as the confidence in data mining literature. The probability is a threshold value that the user needs to specify before training an association model. For example, the following is a typical rule: Product = “Pepsi”, Product =“Chips” => Product =

"Juice" with an 80% probability. The interpretation of this rule is straightforward. If a customer buys Pepsi and chips, there is an 80% chance that he or she may also buy juice. Figure 1.14 displays the product association patterns. Each node in the figure represents a product, each edge represents the relationship. The direction of the edge represents the direction of the prediction. For example, the edge from Milk to Cheese indicates that those who purchase milk might also purchase cheese.

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Rules Discovered:
{Milk} --> {Coke}
{Diaper, Milk} --> {Beer}

Figure 1.13. An example of association rules.

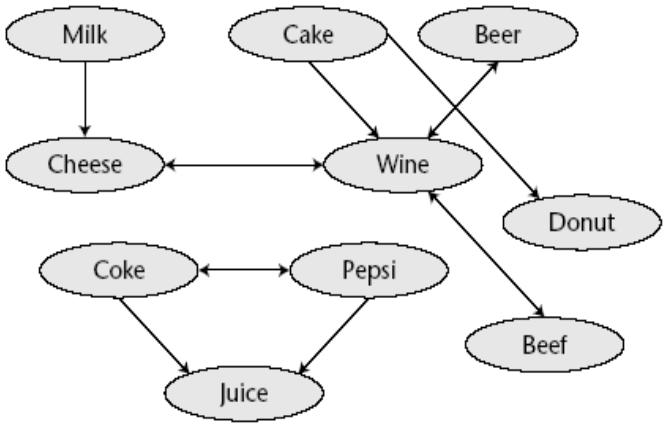


Figure 1.14. Graph of products associations.

Some association rule discovery applications

1. **Marketing and sales promotion** [Berry and Linoff, 2004]
   - Example rule discovered: {Bagels, Coke} ➔ {potato Chips}
   - Insights:
     - Promote bagels to boost potato chips sales
     - If selling bagels is discontinued, this will affect potato chips sales.
     - Coke should be sold together with bagels to boost potato chips sales.

2. Supermarket shelf management [Hofmann and Klinkenberg, 2013]

- Goal: To identify items which are bought together by sufficiently many customers.
- Approach: Process the point-of-sale data collected with barcode scanners to find dependencies among items.
- A classic rule:
  - If a customer buys diaper and milk, then he is very likely to buy beer.
  - So, don't be surprised if you find six-packs stacked next to diapers!

We can summarize the association rules as follows:

- Given a set of records each of which contain some number of items from a given collection.
- Produce dependency rules which will predict occurrence of an item based on occurrences of other items.

1.12.4. Regression (predictive task): The regression task is similar to classification. The main difference is that the predictable attribute is a continuous number. Regression techniques have been widely studied for centuries in the field of statistics. Linear regression and logistic regression are the most popular regression methods. Other regression techniques include regression trees and neural networks. Regression tasks can solve many business problems. For example, they can be used to predict coupon redemption rates based on the face value, distribution method, and distribution volume, or to predict wind velocities based on temperature, air pressure, and humidity.

Some regression applications

1. House pricing

- Goal: real estate agent wants to predict the selling price of a house in order to set an appropriate asking price
- Approach:
  - Database of transactions of previously sold property (location, type of house/apartment, details of the property, asking price, time on market,…)
  - Target variable: sale price of the property
- Learn a regression model that predicts the from the transaction data the sale price
- Many methods: Linear regression, Support vector regression, Nearest neighbor regression, Regression trees

We can summarize the regression problem as follows:

- Predict a value of a given continuous valued variable based on the values of other variables, assuming a linear or nonlinear model of dependency.
- Greatly studied in statistics, neural network fields.
- Examples:
  - Predicting sales amounts of new product based on advetising expenditure.
  - Predicting wind velocities as a function of temperature, humidity, air pressure, etc.
  - Time series prediction of stock market indices.

1.12.5. Anomaly Detection/Deviation analysis (descriptive task): Deviation analysis is for finding those rare cases that behave very differently from others. It is also called outlier detection, which refers to the detection of significant changes from previously observed behavior. Deviation analysis can be used in many applications. The most common one is credit card fraud detection. To identify abnormal cases from millions of transactions is a very challenging task. Other applications include network intrusion detection, manufacture error analysis, and so on. There is no standard technique for deviation analysis. It is still an actively researched topic. Usually analysts employ some modified versions of decision trees, clustering, or neural network algorithms for this task. In order to generate significant rules, analysts need to oversample the anomaly cases in the training dataset.

Some anomaly detection applications

Industrial Process Monitoring

- Goal: a tool that checks if the process is running within normal specifications.
- Data from problem situations not available.
  - e.g. nuclear reactor meltdown.
- Collect measurement data from the process and devise prototype profile(s) of normal operation.
- Large deviation from the prototype causes an alarm.

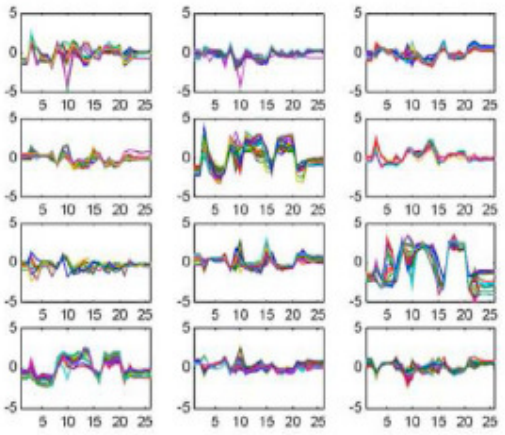


Figure 1.15. An example of anomaly detection application: Industrial process monitoring

1.12.6. Sequential Pattern Mining (descriptive task): Sequence analysis is used to find patterns in a discrete series. A sequence is composed of a series of discrete values (or states). For example, a DNA sequence is a long series composed of four different states: A, G, C, and T. A web click sequence contains a series of URLs. Customer purchases can also be modeled as sequence data. For example, a customer first buys a computer, then speakers, and finally a Webcam. Both sequence and time series data contain adjacent observations that are dependant. The difference is that the sequence series contains discrete states, while the time series contains continuous numbers. Sequence and association data are similar in the sense that each individual case contains a set of items or states. The difference between sequence

and association models is that sequence models analyze the state transitions, while the association model considers each item in a shopping cart to be equal and independent. With the sequence model, buying a computer before buying speakers is a different sequence than buying speakers before a computer. With an association algorithm, these are considered to be the same itemset. Figure 1.6 displays Web click sequences. Each node is a URL category. Each line has a direction, representing a transition between two URLs. Each transition is associated with a weight, representing the probability of the transition between one URL and the other. Sequence analysis is a relatively new data mining task. It is becoming more important mainly due to two types of applications: Web log analysis and DNA analysis. There are several different sequence techniques available today such as Markov chains. Researchers are actively exploring new algorithms in this field. Figure 1.6 displays the state transitions among a set of URL categories based on Web click data.

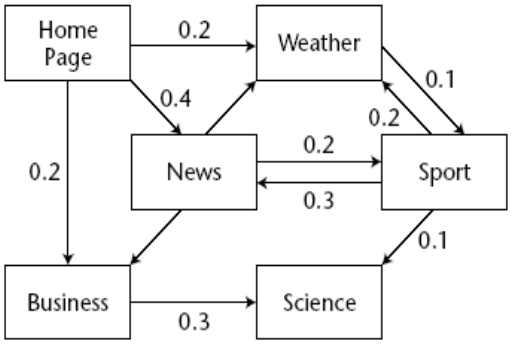


Figure 1.16. Web navigation sequence

Some pattern discovery applications

1. ***Basketball scout***
   - In NBA, rigorous statistics are kept of events of play and the actions of all players
   - Advanced Scout : system discovers interesting patterns
     - e.g."When player X is on the field the shooting accuracy of player Y drops from 75% to 30%

2. ***Scientific discovery***
   - Explorative analysis of scientific data can reveal unexpected associations.
   - Especially useful in "weak theory" domains where human experts do not yet know all the relevant variables

3. ***Motif discovery in bio-sequences***
   - In DNA data, binding sites of regulatory proteins are characterized by distinct subsequences
   - Biologically important motifs should occur more frequently than a random subsequence
   - The patterns are typically not completely fixed but allow variability in certain locations
   - Sequential pattern: order of items (letters) important

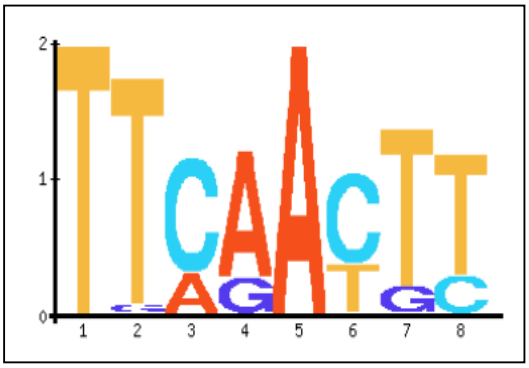


Figure 1.17. Motif discovery in bio-sequences

1.12.7. Time Series Prediction (Forecasting/predictive task): Forecasting is yet another important data mining task. What will the stock value of MSFT be tomorrow? What will the sales amount of Pepsi be next month? Forecasting can help to answer these questions. It usually takes as an input time series dataset, for example a sequence of numbers with an attribute representing time. The time series data typically contains adjacent observations, which are order-dependant. Forecasting techniques deal with general trends, periodicity, and noisy noise filtering. The most popular time series technique is ARIMA, which stands for Auto-regressive Integrated Moving Average model. Figure 1.17 contains two curves. The solid line curve is the actual time series data on Microsoft stock value, while the dotted curve is a time series model based on the moving average forecasting technique [Domingos and Hulten, 2000].

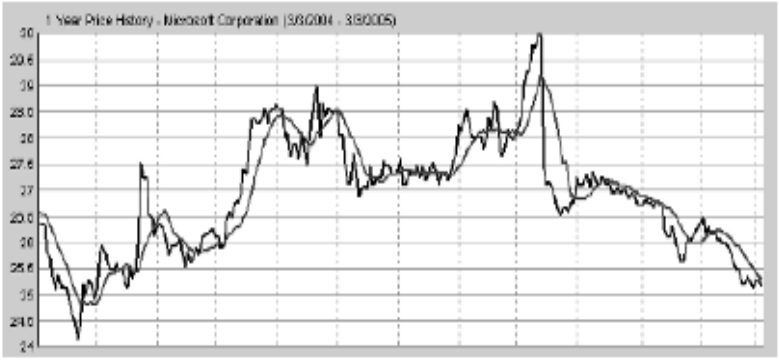


Figure 1.18 . Time series prediction (Forecasting problem)

***1.12.8.* Decision Making:** Decision-making is an important part of all science-based professions, where specialists apply their knowledge in a given area to make informed decisions. They make models that help to formulate and algorithmically solve decision making problems. i.e., that find a solution that maximizes the expected benefit of the outcome. Among the fields that require decision making, we note:

- ✓ Planning Problems and Algorithms.
- ✓ Probabilistic Graphical Models.

- ✓ Decision Theory and Decision Networks.
- ✓ Game Theory and Mechanism Design.

1.13. Data Mining Project Cycle

What is the life cycle of a data mining project? What are the challenging steps? Who should be involved in a data mining project? To answer these questions, let's go over a typical data mining project step by step [Hand et al., 2001, Zaki and Wagner, 2014, Han and Kamber, 2006].

Step 1: Data Collection

The first step of data mining is usually data collection. Business data is stored in many systems across an enterprise. For example, there are hundreds of OLTP databases and over 70 data warehouses inside Microsoft. The first step is to pull the relevant data to a database or a data mart where the data analysis is applied. For instance, if you want to analyze the Web click stream and your company has a dozen Web servers, the first step is to download the Web log data from each Web server. If the data warehouse on the subject of your analysis already exists. However, the data in the data warehouse may not be rich enough. You may still need to gather data from other sources. Suppose that there is a click stream data warehouse containing all the Web clicks on the Web site of your company. You have basic information about customers' navigation patterns. However, because there is not much demographic information about your Web visitors, you may need to purchase or gather some demographic data from other sources in order to build a more accurate model. After the data is collected, you can sample the data to reduce the volume of the training dataset. In many cases, the patterns contained in 50,000 customers are the same as in 1 million customers.

Step 2: Data Cleaning and Transformation

Data cleaning and transformation is the most resource-intensive step in a data mining project. The purpose of data cleaning is to remove noise and irrelevant information out of the dataset. The purpose of data transformation is to modify the source data into different formats in terms of data types and values. There are various techniques you can apply to data cleaning and transformation, including:

1. ***Data type transform:*** This is the simplest data transform. An example is transforming a Boolean column type to integer. The reason for this transform is that some data mining algorithms perform better on integer data, while others prefer Boolean data.
2. ***Continuous column transform:*** For continuous data such as that in Income and Age columns, a typical transform is to bin the data into buckets. For example, you may want to bin Age into five predefined age groups. Apart from binning, techniques such as normalization are popular for transforming continuous data. Normalization maps all numerical values to a number between 0 and 1 (or –1 to 1) to ensure that large numbers do not dominate smaller numbers during the analysis.
3. ***Grouping:*** Sometimes there are too many distinct values (states) for a discrete column. You need to group these values into a few groups to reduce the model's complexity. For example, the column Profession may have tens of different values such as Software Engineer, Telecom Engineer, Mechanical Engineer, Consultant, and so on. You can group

various engineering professions by using a single value: Engineer. Grouping also makes the model easier to interpret.

4. ***Aggregation:*** Aggregation is yet another important transform. Suppose that there is a table containing the telephone call detail records (CDR) for each customer, and your goal is to segment customers based on their monthly phone usage. Since the CDR information is too detailed for the model, you need to aggregate all the calls into a few derived attributes such as total number of calls and the average call duration. These derived attributes can later be used in the model.
5. ***Missing value handling:*** Most datasets contain missing values. There are a number of causes for missing data. For instance, you may have two customer tables coming from two OLTP databases. Merging these tables can result in missing values, since table definitions are not exactly the same. In another example, your customer demographic table may have a column for age. But customers don't always like to give you this information during the registration. You may have a table of daily closing values for the stock MSFT. Because the stock market closes on weekends, there will be null values for those dates in the table. Addressing missing values is an important issue. There are a few ways to deal with this problem. You may replace the missing values with the most popular value (constant). If you don't know a customer's age, you can replace it with the average age of all the customers. When a record has too many missing values, you may simply remove it. For more advanced cases, you can build a mining model using those complete cases, and then apply the model to predict the most likely value for each missing case.
6. ***Removing outliers:*** Outliers are abnormal cases in a dataset. Abnormal cases affect the quality of a model. For example, suppose that you want to build a customer segmentation model based on customer telephone usage (average duration, total number of calls, monthly invoice, international calls, and so on) There are a few customers (0.5%) who behave very differently. Some of these customers live aboard and use roaming all the time. If you include those abnormal cases in the model, you may end up by creating a model with majority of customers in one segment and a few other very small segments containing only these outliers. The best way to deal with outliers is to simply remove them before the analysis. You can remove outliers based on an individual attribute; for instance, removing 0.5% customers with highest or lowest income. You may remove outliers based on a set of attributes. In this case, you can use a clustering algorithm. Many clustering algorithms, including Microsoft Clustering, group outliers into a few particular clusters.

   There are many other data-cleaning and transformation techniques, and there are many tools available in the market. SQL Server Integration Services (SSIS) provides a set of transforms covering most of the tasks listed here.

Step 3: Model Building

Once the data is cleaned and the variables are transformed, we can start to build models. Before building any model, we need to understand the goal of the data mining project and the type of the data mining task. Is this project a classification task, an association task or a segmentation task? In this stage, we need to team up with business analysts with domain

knowledge. For example, if we mine telecom data, we should team up with marketing people who understand the telecom business.

Model building is the core of data mining, though it is not as time- and resource-intensive as data transformation. Once you understand the type of data mining task, it is relatively easy to pick the right algorithms. For each data mining task, there are a few suitable algorithms. In many cases, you won't know which algorithm is the best fit for the data before model training. The accuracy of the algorithm depends on the nature of the data such as the number of states of the predictable attribute, the value distribution of each attribute, the relationships among attributes, and so on. For example, if the relationship among all input attributes and predictable attributes were linear, the decision tree algorithm would be a very good choice. If the relationships among attributes are more complicated, then the neural network algorithm should be considered.

The correct approach is to build multiple models using different algorithms and then compare the accuracy of these models using some tool, such as a lift chart, which is described in the next step. Even for the same algorithm, you may need to build multiple models using different parameter settings in order to fine-tune the model's accuracy.

Step 4: Model Assessment

In the model-building stage, we build a set of models using different algorithms and parameter settings. So what is the best model in terms of accuracy? How do you evaluate these models? There are a few popular tools to evaluate the quality of a model. The most well-known one is the lift chart. It uses a trained model to predict the values of the testing dataset. Based on the predicted value and probability, it graphically displays the model in a chart. In the model assessment stage, not only do you use tools to evaluate the model accuracy but you also need to discuss the meaning of discovered patterns with business analysts. For example, if you build an association model on a dataset, you may find rules such as Relationship = Husband => Gender = Male with 100% confidence. Although the rule is valid, it doesn't contain any business value. It is very important to work with business analysts who have the proper domain knowledge in order to validate the discoveries. Sometimes the model doesn't contain useful patterns. This may occur for a couple of reasons. One is that the data is completely random. While it is possible to have random data, in most cases, real datasets do contain rich information. The second reason, which is more likely, is that the set of variables in the model is not the best one to use. You may need to repeat the data-cleaning and transformation step in order to derive more meaningful variables. Thus, data mining is a cyclic process; it usually takes a few iterations to find the right model.

Step 5: Reporting

Reporting is an important delivery channel for data mining findings. In many organizations, the goal of data miners is to deliver reports to the marketing executives. Most data mining tools have reporting features that allow users to generate predefined reports from mining models with textual or graphic outputs. There are two types of reports: reports about the findings (patterns) and reports about the prediction or forecast.

Step 6: Prediction (Scoring)

In many data mining projects, finding patterns is just half of the work; the final goal is to use these models for prediction. Prediction is also called scoring in data mining terminology. To give predictions, we need to have a trained model and a set of new cases. Consider a banking scenario in which you have built a model about loan risk evaluation. Every day there are thousands of new loan applications. You can use the risk evaluation model to predict the potential risk for each of these loan applications.

Step 7: Application Integration

Embedding data mining into business applications is about applying intelligence back to business. That is, closing the analysis loop. According to Gartner Research, in the next few years, more and more business applications will embed a data mining component as a value-added. For example, CRM applications may have data mining features that group customers into segments. ERP applications may have data mining features to forecast production. An online bookstore can give customers real-time recommendations on books. Integrating data mining features, especially a real-time prediction component into applications is one of the important steps of data mining projects. This is the key step for bringing data mining into mass usage.

Step 8: Model Management

It is challenging to maintain the status of mining models. Each mining model has a life cycle. In some businesses, patterns are relatively stable and models don't require frequent retraining. But in many businesses patterns vary frequently. For example, in online bookstores, new books appear every day. This means that new association rules appear every day. The duration of a mining model is limited. Anew version of the model must be created frequently. Ultimately, determining the model's accuracy and creating new versions of the model should be accomplished by using automated processes. Like any data, mining models also have security issues. Mining models contain patterns. Many of these patterns are the summary of sensitive data. We need to maintain the read, write, and prediction rights for different user profiles. Mining models should be treated as first-class citizens in a database, where administrators can assign and revoke user access rights to these models.

[Fayyad et al., 1996a] proposed to reduce these steps of data mining process to four main steps which are: data selection step, preprocessing step, data mining step, and post-processing step as it is shown in the figure bellow.

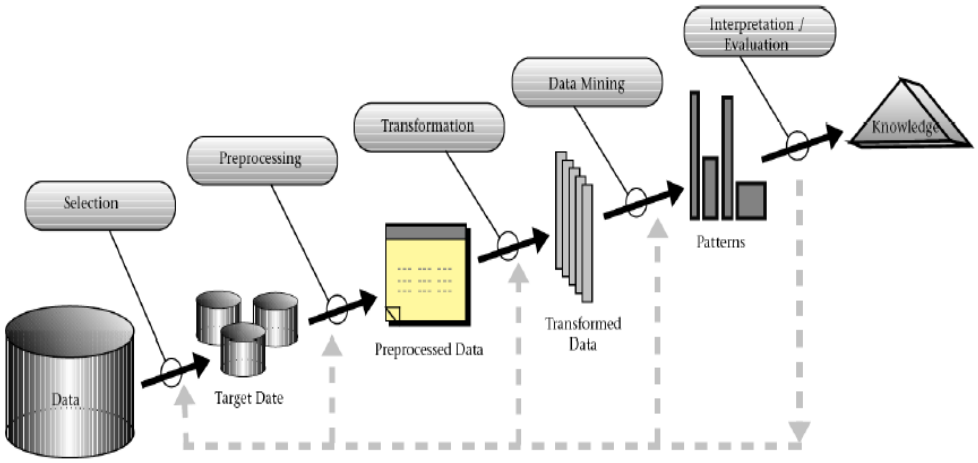


Figure 1.19. Data mining process step by step (Source [Fayyad et al. 1996])

1.14. Types of data sets used in Data mining field

Data sets used in data mining field are mostly different in a number of ways. For instance, the attributes used to describe data objects can be quantitative or qualitative type. Data sets may have special characteristics; e.g., some data sets contain time series or objects with explicit relationships to one another. Thus, each type of data determines which tools and techniques can be used to analyze the data. Furthermore, new research in data mining is often driven by the need to accommodate new application areas and their new types of data. [Pang-Ning et al., 2013].

We note also that data is often far from perfect. While most data mining techniques can tolerate some level of imperfection in the data, a focus on understanding and improving data quality can typically improve the quality of the resulting analysis. Data quality can be damaged by the presence of noise and outliers; missing, inconsistent, or duplicate data. [Pang-Ning et al., 2013]. For this purpose, preprocessing steps is likely required to make date suitable for analysis.

There are many types of data sets, and more the field of data mining develops, a greater variety of data sets become available for analysis. In this section we try to describe some of the most common types, notably: record data, graph-based data, and ordered data.

1. ***Record data:*** data set is a collection of records (data objects), each of which consists of a fixed set of data fields (attributes). Record data is usually stored either in flat files or in relational databases. Relational databases are certainly more than a collection of records, but data mining often does not use any of the additional information available in a relational database. Rather, the database serves as a convenient place to find records. Record data include:
   a. ***Transaction or market basket data:*** Transaction data is a special type of record data, where each record (transaction) involves a set of items. We take an example of a grocery

store. The set of products purchased by a customer during one shopping trip constitutes a transaction, while the individual products that were purchased are the items.

b. ***Data Matrix:*** If the data objects in a collection of data all have the same fixed set of numeric attributes, then the data objects can be thought of as points (vectors) in a multidimensional space, where each dimension represents a distinct attribute describing the object. A set of such data objects can be interpreted as an *m* by *n* matrix, where there are *m* rows, one for each object, and *n* columns, one for each attribute. This matrix is called a data matrix or a pattern matrix. A data matrix is a variation of record data, but because it consists of numeric attributes, standard matrix operation can be applied to transform and manipulate the data. Therefore, the data matrix is the standard data format for most statistical data.

c. ***Document Data:*** is a special case of a data matrix in which the attributes are of the same type and are asymmetric; i.e., only non-zero values are important. if the order of the terms (words) in a document is ignored, then a document can be represented as a term vector, where each term is a component (attribute) of the vector and the value of each component is the number of times the corresponding term occurs in the document. This representation of a collection of documents is often called a document-term matrix.

Tid	Refund	Marital status	Texable income	Defaulted borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	single	90K	yes

(a) Record Data

Tid	Items
1	Bread, Soda, Milk
2	Bear, Bread
3	Beer, Soda, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Soda, Diaper, Milk

(b) Transction Data

Projection of X load	Projection of Y load	Distance	Load	Thick -ness
10.23	5.27	15.22	27	1.2
12.65	6.25	16.22	22	1.1
13.54	7.23	17.34	23	1.2
14.27	8.43	18.45	25	0.9

(c) Data matrix

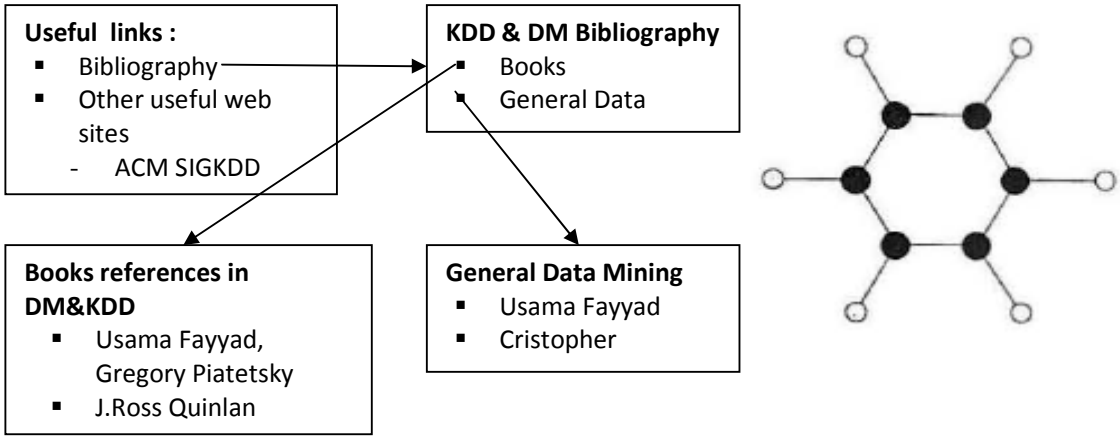
	team	coach	play	ball	score	game	win	lost
D1	3	0	5	0	2	6	0	2
D2	0	7	0	2	1	0	0	3
D3	0	1	0	0	1	2	2	0

(d) Document-term matrix.

Figure 1.20. Variations of record data.

2. **Graph-based data:** A graph can sometimes be a convenient and powerful representation for data. We consider two specific cases: (1) the graph captures relationships among data objects and (2) the data objects themselves are represented as graphs. As examples of cases

in which a graph is used to represent data, we have: World Wide Web links, molecular Structures.



(a) Linked Web pages. (b) Benzene molecule.

Figure 1.21. Different variations of graph data. (Source: [Pang-Ning et al., 2013])

3. ***Ordered Data:*** For some types of data, the attributes have relationships that involve order in time or space. Different types of ordered data are described next and are shown in Figure 1.22.

   a. ***Sequential Data:*** called also temporal data, is an extension of record data, where each record has a time associated with it. For instance, a retail transaction data set that also stores the time at which the transaction took place. This time information makes it possible to find patterns such as "candy sales peak before Halloween." [Domingos and Hulten, 2000]

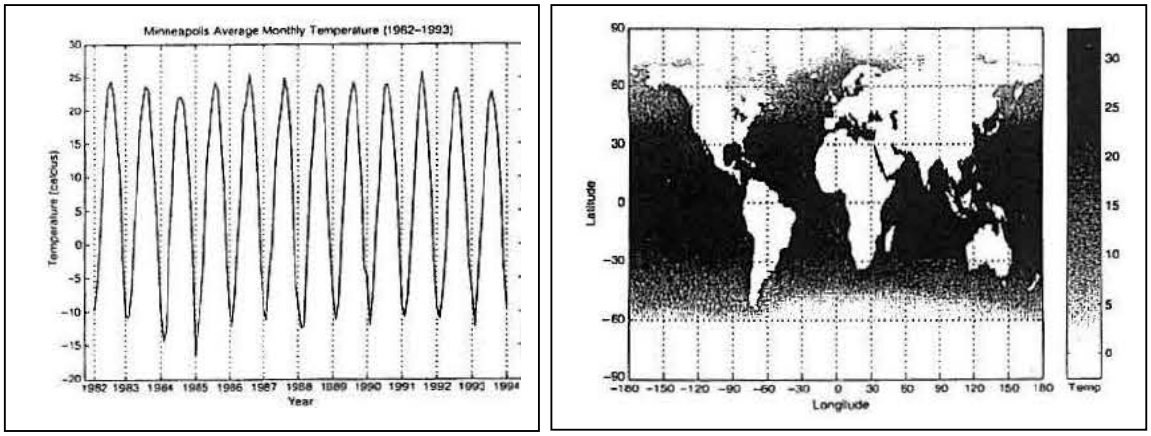
Time	Customer	Items purchased
t1	C1	A, B
t2	C3	A, C
t2	C1	C, D
t3	C2	A, D
t4	C2	E
t5	C1	A, E

Customer	Time and Items Purchased
C1	(t1 : A, B) (t2 : C, D) (t5 : A, E)
C2	(t3 : A, D) (t4 : E)
C3	(t2 : A, C)

(a) Sequential transaction data

```
GGTTCCGCCTTCAGCCCCGCGCC
CGCAGGGCCCGCCCCGCGCCGTC
GAGAAGGGCCCGCCTGGCGGGCG
GGGGGAGGCGGGGCCGCCCGAGC
CCAACCGAGTCCGACCAGGTGCC
CCCTCTGCTCGGCCTAGACCTGA
GCTCATTAGGCGGCAGCGGACAG
GCCAAGTAGAACACGCGAAGCGC
TGGGCTGCCTGCTGCGACCAGGG
```

(b) Genomic sequence data



(a) Temperature time series (d) Spatial temperature data

Figure 1.22. Different variations of ordered data

b. ***Sequence Data:*** Sequence data consists of a data set that is a sequence of individual entities, such as a sequence of words in a sentence or letters in a word. It is quite similar to sequential data, except that there are no time stamps; instead, there are positions in an ordered sequence. For example, the genetic information of plants and animals can be represented in the form of sequences of nucleotides (genes). Many of the problems associated with genetic sequence data involve predicting similarities in the structure and function of genes from similarities in nucleotide sequences. Figure 1.22(b) shows a section of the human genetic code expressed using the four nucleotides from which all DNA is constructed: A, T, G, and C. [Domingos and Hulten, 2000]

c. ***Time Series Data:*** Time series data is a special type of sequential data in which each record is a time series, i.e., a series of measurements taken over time. For example, Figure 1.22(c), shows a time series of the average monthly temperature for Minneapolis during the years 1982 to 1994. When working with temporal data, it is important to consider temporal autocorrelation; i.e., if two measurements are close in time, then the values of those measurements are often very similar.

d. ***Spatial Data:*** Some objects have spatial attributes, such as positions or areas, as well as other types of attributes. An example of spatial data is weather data (precipitation, temperature, pressure) that is collected for a variety of geographical locations. An important aspect of spatial data is spatial autocorrelation; i.e., objects that are physically close tend to be similar in other ways as well. Thus, two points on the Earth that are close to each other usually have similar values for temperature and rainfall. For instance, Earth science data sets record the temperature or pressure measured at points (grid cells) on latitude-longitude spherical grids of various resolutions, e.g., 1° by 1°. (See figure 1.22(d).)

1.15. Major Vendors and Products

One research conclusion is shared by various analysts: the data mining market is the fastest growing business intelligence component (reporting, OLAP, packaged data marts, and so on). Data mining currently represents about 15% of the business intelligence market. It is evolving from transitional horizontal packages toward embedded data mining applications, integrated with CRM, ERP, or other business applications. In fact, there are hundreds of data mining product and consulting companies. KDNuggets (kdnuggets.com) has an extended list of most of these companies and their products in the data mining field. Here we list a few the major data mining product companies. [Han et al., 2002]

1. ***SAS:*** SAS is probably the largest data mining product vendor in terms of the market share. SAS has been in the statistics field for decades. SAS Base contains a very rich set of statistical functions that can be used for all sorts of data analysis. It also has a powerful script language called SAS Script. SAS Enterprise Miner was introduced in 1997. It provides the user with a graphical flow environment for model building, and it has a set of popular data mining algorithms, including decision trees, neural network, regression, association, and so on. It also supports text mining.

2. ***SPSS:*** SPSS is another major statistics company. It has a number of data mining products including SPSS base and Answer Tree (decision trees). SPSS acquired a British company ISL in late 1998 and inherited the Clementine data mining package. Clementine was one of the first companies to introduce the data mining flow concept, allowing users to clean data, transform data, and train models in the same workflow environment. Clementine also has tools to manage data mining project cycle.

3. ***IBM:*** IBM has a data mining product called Intelligent Miner, developed by an IBM German subsidiary. Intelligent Miner contains a set of algorithms and visualization tools. Intelligent Miner exports mining models in Predictive Modeling Markup Language (PMML), which was defined by the Data Mining Group (DMG), an industry organization. PMML documents are Extensible Markup Language (XML) files containing the descriptions of model patterns and statistics of training dataset. These files can be loaded by DB2 database for prediction purpose.

4. ***Microsoft Corporation:*** Microsoft was the first major database vendor to include data mining features in a relational database. SQL Server 2000, released in September 2000, contains two patented data mining algorithms: Microsoft Decision Trees and Microsoft Clustering. Apart from these algorithms, the most important data mining feature is the implementation of OLE DB for Data Mining. OLE DB for Data Mining is an industry standard that defines a SQL-style data mining language and a set of schema rowsets targeted at database developers. This API makes it very easy to embed data mining components, especially prediction features, into user applications.

5. ***Oracle:*** Oracle 9i shipped in 2000, containing a couple of data mining algorithms based on association and Naïve Bayes. Oracle 10g includes many more data mining tools and algorithms. Oracle also incorporated the Java Data Mining API, which is a Java package for data mining tasks.

6. ***Angoss:*** Angoss' KnowledgeSTUDIO is a data mining tool that includes the power to build decision trees, cluster analysis, and several predictive models, allowing users to mine and understand their data from many different perspectives. It includes powerful data visualization tools to support and explain the discoveries. Angoss also has a set of content viewer controls, which work with data mining algorithms in SQL Server 2000. Its algorithms can also be plugged into the SQL Server platform.

7. ***KXEN:*** KXEN is a data mining software provider based in France. It has a number of data mining algorithms, including SVM, regression, time series, segmentation, and so forth. It also provides data mining solutions for OLAP cubes. It developed an Excel add-in that allows users to do data mining in a familiar Excel environment.

1.16. Text Mining, Web Mining, XML Mining: New applications of Data Mining

Sometimes, the processed data are note numerical or categorical data as it is the case for data mining, but unstructured or structured text. We talk in this case about new extensions or applications of data mining which are: Text Mining, Web Mining, XML Mining, and Web Log Mining. The essential difference between all these types of mining is the type and the source of processed data.

1.16.1. Text Mining: text mining refers to data mining using text documents as source of data. Most text mining tasks use information retrieval IR methods to pre-process text documents. These methods are quite different from traditional data pre-processing methods used for relational tables. Text mining is useful for many topics such as: [Feldman and Singer, 2007, Weiss et al., 2010]

- Text categorization and clustering.
- Named entity extraction.
- Entity relation modeling.
- Sentiment analysis.

Among the major applications of text mining, we can note the following:

- Biomedical data mining
- Media monitoring
- Marketing

1.16.2. Web Mining: one of the applications of data mining techniques. It can be defined as the discovery and analysis of useful information from the World Wide Web. This describe three topics: [Chakrabarti, 2003, Scime, 2005, Zhang et al., 2007]

- ***Web content mining:*** explores the data contained in related sites in order to provide better resources for visitors (related to the field of text mining).
- ***Web usage mining:*** determines the navigation patterns of users on a site using the log files of web. This can help in modeling and optimization of web site.
- ***Web structure mining:*** examines the link hierarchy (structure) of a web in order to improve navigation (e.g. Google Page Rank).

1.16.3. XML Mining: becomes one of the most important applications of data mining. Oriented to structured and semi-structured texts such as XML documents. Because of the particularity of XML documents as well as the specificity of their structure, this new discipline requires the development of new approaches and tools which should be more appropriate to satisfy the needs of new formats of data.

1.17. Future perspectives in Data Mining

Data mining is relatively young compared to database technology. It is still considered a niche and emerging market. One of the reasons for this designation is that most of the data mining packages are targeted toward statisticians or data miners. Application developers find it too challenging to master these technologies. More recently, a number of data mining vendors, including Microsoft, realized this and initialized data mining APIs that are directed toward the developer communities. We believe that in the next few years there will be more and more developers who will be able to build mining models, and as a consequence a large number of applications will include data mining features. We foresee the following perspectives in data mining field over the next few years: [Smyth, 2001, Dunham, 2002, Han et al., 2002]

1. ***Embedded data mining:*** More and more business applications will include data mining features, such as the prediction feature, for added value. For example, a business application will allow users to forecast product sales. Online retailers will recommend products to customers for cross-selling purposes. This will be due to the fact that industrial data mining APIs, such as OLE DB for Data Mining, enable database and application developers to use data mining features and embed them in a line of business applications. Embedded data mining will increase the overall size of data mining market.

2. ***Data mining packages for vertical applications:*** Data mining is becoming popular as major database vendors add it to their database management system (DBMS) packages. Today, data mining can be applied to almost every sector, in finance, insurance, and telecom. Therefore, there is a growing need for specialized data mining techniques to solve business problems in many vertical sectors. For example, in the health care field, we need special data mining techniques to analyze DNA sequences. In network security applications, we need real time training algorithms to detect network intrusion. We need also nontraditional data mining techniques to analyze the unstructured data in the World Wide Web. Text mining is yet another vertical sector to which we need to apply data mining. So, traditional horizontal data mining packages are too general to solve these

problems. We foresee there will be more new data mining packages specialized for these vertical sectors.

3. ***Products consolidation:*** Hundreds of software vendors are providing horizontal data mining packages. Many packages include only one or two algorithms. The data mining market is still very fragmented. Just as with other software sectors, consolidation is inevitable. Small vendors will find more competitive pressure in the horizontal market, especially when major database vendors add data mining features to DBMSs.

4. ***PMML***: Although big vendors, such as Microsoft, Oracle, IBM, and SAS, are competing on various data mining APIs, they are all member of the same club: the DMG (Data Mining Group). They all support PMML as the model persistence format. PMML offers many advantages in terms of model exchange and model deployment. Because it is an XML document, it is also editable by advanced users. PMML will become more popular in the near future.

1.18. Summary

In this chapter, we have presented an extended introduction to data mining. In fact, we have presented the basics of data mining. The principle idea behind data mining is not very complex; it is about discovering hidden patterns from historical datasets and applying these patterns for predictions. There are a handful of data mining tasks, including: classification, clustering, regression, association, forecasting, fraud detection, and visualization. These tasks cover hundreds of business scenarios. After this short overview on data mining, we can say that we have learned the basic concepts of the set of data mining techniques and the typical life cycle of a data mining project.
The chapter also told us about the current data mining market and major product vendors. We learned about the trends for data mining over the next few years.

CHAPTER 2

Fundamentals of Text Mining

Chapter Content

2.1. Introduction

The analysis and the treatment of textual information became a major stake with the explosion of the Web, because most accessible information is textual (electronic libraries, HTML pages, forums of discussion, etc.). Besides, it is not sufficient to analyze, but also to extract hidden and useful information in these data: "It is not sufficient to read lines of texts anymore but also to read what is written between these lines". [Song et al., 2009]

The extraction of Knowledge from Data (Knowledge Data Discovery KDD) designates the global process of knowledge discovery that permits to pass from raw data to knowledge, whereas the extraction of knowledge from text (often named text mining) is only a stage of the KDD during what a model is constructed.

The extraction of knowledge from data (often named" dated mining") is the exploration and the analysis of big quantities of data in order to discover implicit information there. This information can be of different nature, for example one will search for rules of association, one classification or a segmentation of population. [Berry and Kogan, 2010]

There is also a branch of the extraction of knowledge from data that focalize on the analysis of free texts: the extraction of knowledge from textual data or" text mining ". This branch aims to facilitate the extraction of knowledge “hidden" in documents. This domain of research can refer to some techniques of linguistic processing, data mining, machine learning, and of statistics.

In this chapter, we try to discover this interesting field of research by presenting its definition, domains of application, tools and techniques used in it, different tasks and phases, others.

2.2. Some definitions of Text Mining

Several definitions are given here. In this section we present the most popular among them.

- Text mining is used to extract information from free form text data such as that in claim description fields. [Grivel, 2007]
- Text mining refers to a collection of methods used to find patterns and create intelligence from unstructured text data. [Witten, 2004]
- The extraction of implicit, previously unknown and potentially useful information from a large amount of textual resources. [Paulheim et al., 2014]
- Text Mining is a knowledge-intensive process in which a user interacts with a document collection over time by using a suite of analysis tools [Feldman and Sanger, 2007].
- In a manner analogous to data mining, text mining seeks to extract useful information from data sources through the identification and exploration of interesting patterns. But In the case of text mining, however, the data sources are document collections, and interesting patterns are found not among formalized database records but in the unstructured textual data in the documents in these collections. [Feldman and Sanger, 2007]

2.3. Data Mining Vs Text Mining

The concept of data mining refers to finding valuable patterns from large volumes of highly structured data necessitating extensive preparation. Data mining methods learn from samples of past experience and their data will be in numerical form. [Weiss et al., 2010]

The “text miners” do not expect an orderly series of numbers. They look at collections of documents, where the contents are readable and their meaning is obvious.

So, the first distinction between data and text mining is: numbers vs. text. But both are based on samples of past examples. The composition of the examples is very different, yet many of the learning methods are similar. And for text mining, the text will be processed and transformed into a numerical representation. [Weiss et al., 2010]

2.4. Structured and unstructured data

1. ***Structured data:*** If data can be described by a spreadsheet with its tabular format, then the problem is highly structured. So, structured data are characterized by:
   - Loadable into spreadsheet
     - Rows and columns
     - Data is consistent, uniform
     - Each cell filled, or could be filled
     - A completed row is a complete example of past experience.
     - A column is an example of one pattern (measurement/attribute).
   - Data mining friendly
   - Structured data include :
     - ✓ data base
     - ✓ Data warehouse, …

Gender	BP	Weight	Code
M	175	65	3
F	141	72	1
…	…	…	…
F	160	59	2

Figure. 2.1 A spreadsheet example of medical data (Systolic disease)

2. Unstructured data (documents)

A text document may be seen, from many perspectives, as a structured object. From a linguistic perspective, a document demonstrates a rich amount of semantic and syntactical structure which is implicit and hidden in its textual content. In addition, typographical elements such as punctuation marks, capitalization, numeric, and special characters - particularly when coupled with layout artifacts such as: white spacing, carriage returns, underlining, asterisks, tables, columns, and so on – can often serve as a kind of "soft markup" language, providing clues to help identify important document subcomponents such as paragraphs, titles, publication dates, author names, table records, headers, and footnotes. Word sequence may also be a structurally meaningful dimension to a document. At the other end of the "unstructured" spectrum, some text documents, like those generated from a WYSIWYG HTML editor, actually possess from their inception more overt types of embedded metadata in the form of formalized markup tags. For this purpose, we can distinguish two levels of unstructured data or documents [Feldman and Sanger, 2007]

a. Weakly structured documents: documents that have relatively little in the way of strong typographical, layout, or markup indicators to denote structure are sometimes referred to as free format or weakly structured documents. Under this category of documents, we can note:
   - Scientific research papers.
   - Business reports.

- Legal memoranda.
- News Stories.
- Technical documents.
- Corporate documents.
- Books
- Customer complaint letters.

b. ***Semi-structured data:*** On the other hand, documents with extensive and consistent format elements in which field-type metadata can be more easily inferred – such as:
   - Email
   - HTML web pages
   - PDF files
   - Word processing files with heavy document templating or style-sheet constraints.

2.5. Why Text Mining? Motivation

- The large volume of data !! (> 1.8 zettabytes (1.8 trillion GB))
- Most of the world's data is unstructured.
  Approximately 90% of the world's data is held in unstructured formats.

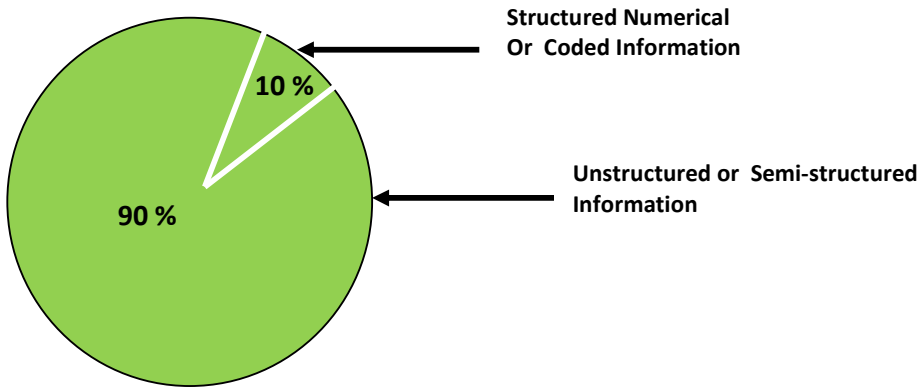


Figure 2.2. Proportion of structured and unstructured data
(source: Oracle Corporation)

- Structured data often misses elements critical to predictive modeling.
- Leveraging text should improve decisions and predictions.
- Text mining is gaining momentum.
  - Sentiment Analysis (twitter, facebook)
  - Predicting stock market
  - Predicting churn
  - Customer influence
  - Customer Service and Help Desk

2.6. Where Text Mining can be placed?

Text mining is a new field issued from data mining. But its techniques and algorithms existed for many years ago. The most of them are derived from old disciplines such as: statistics, machine learning, databases, library and information sciences, natural language processing NLP, information retrieval, and recently data mining. The figure bellow shows clearly where text mining can be placed.

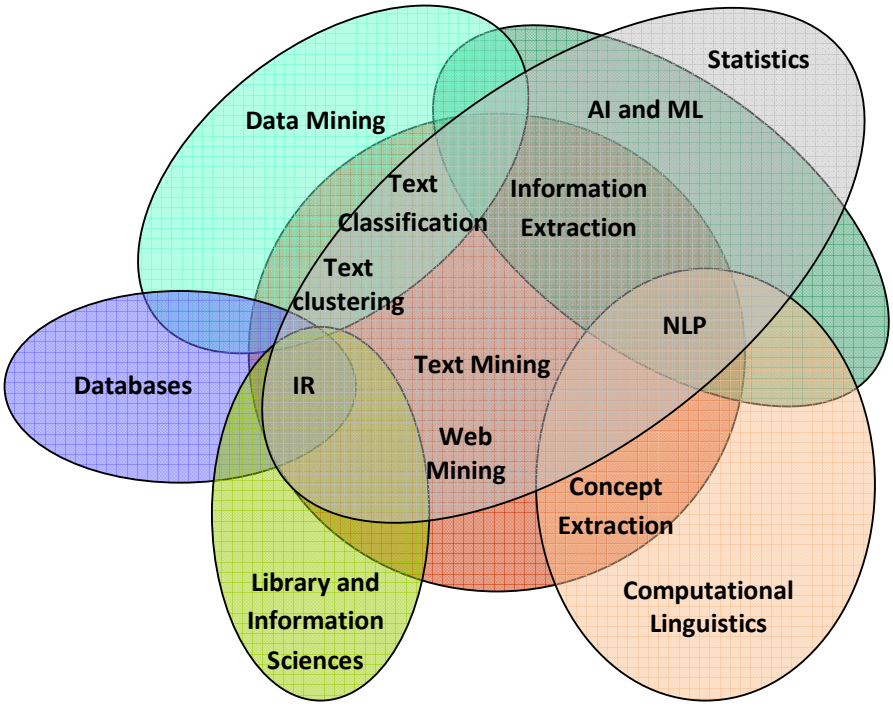


Figure 2.3. Origins of text mining

2.7. Why Text Mining is hard? Major difficulties

We have seen that data mining techniques are applied on highly structured data, subject to a very valuable preparation phase. This can influence positively on the good conduct of these techniques and their efficiency, as well as the quality of expected results. On the other hand, text mining techniques are applied on free texts, not or few structured, with a possibly obscure meaning. In addition, these texts can be written in different languages and represented by various encodings (ASCII, UNICODE, UTF, etc), without forgetting the format in which a text is saved (.Doc, .Xls, .Pdf, .HTML, .XML, etc). All of these elements and others can be a source of huge problems for users of text mining techniques. In the following section, we present some of these difficulties: [Abbott, 2013]

1. Language is ambiguous

- Context is needed to clarify
  Example 2.1: language ambiguity.
  "*john saw the man in the park with the telescope.*"
  To whom does the telescope belongs? John? The park? The man in the park?
  (Depending upon the use of the prepositional phrase "with the telescope")

- The same words can mean different things (homographs)
  Example 2.2: Homograph problem
  Bear (verb)-to support or carry
  Bear (noun)-a large animal

More subtly, sentences that no human would judge ambiguous can cause problems to computer.

2. ***Concept/word extraction*** usually results in huge number of dimensions
   - Thousands of new fields.
   - Each field typically has low information content (sparse).

3. Misspellings, abbreviations, spelling variants.

4. ***Homonymy:*** same words having different meaning.

   Example 2.3: Problem of homonymy
   - Mary walked along the bank of the river (bank: a river margin).
   - Harbor Bank is the richest bank in the city (refers to a financial institution)

5. ***Synonymy:*** synonyms are different words having similar or same meaning; can substitute one word for the other without changing the meaning of the sentence substantively. Synonyms can have differing connotations:

   Example 2.4: Problem of synonymy
   a. Miss Nelson became a kind of big sister to Benjamin.
   b. Miss Nelson became a kind of large sister to Benjamin.

6. ***Polysemy:*** same word or form, but different, albeit related meaning

   Example 2.5: Problem of polysemy.
   - The bank raised its interest rates yesterday (a financial institution).
   - The store is next to the newly constructed bank.
   - The bank appeared first in Italy in the renaissance.

7. ***Hyponymy:*** concept hierarchy or subclass (subordinates)

   Example 2.6: Problem of hyponymy
   - Animal (noun): Dog, Cat, …

2.8. Text Mining Applications

In this section we present some areas where text mining methods are used and can work very well.

2.9.3. Document classification: This is among the most widely studied and applied methods and applications of data mining. Given a sample of past experience and correct answers for each example, the objective is to find the correct answers for new examples. It's a kind of supervised classification or also called categorization. The same concept can be extended to data that do not have clearly labeled answers. Our task in this case would be to organize the data in such a way that we can make up labels or answers and expect these to hold in the future. This process is referred to as unsupervised classification or clustering. Although similarity between documents is an essential ingredient in organizing unlabeled documents into distinct groups, measuring similarity of documents is an end in itself. [Hayes and Weinstein, 1990, Lewis, and Ahn, 1995, Giorgetti and Sebastiani, 2003a, Giorgetti and Sebastiani , 2003b,]

Originally, this type of problem was considered a form of indexing, much like the index of a book. As more and more documents have become available online, the applicability of this task has broadened. Some of the more obvious tasks are related to e-mail: for example, automatically forwarding e-mail to the appropriate company department or detecting spam mail.

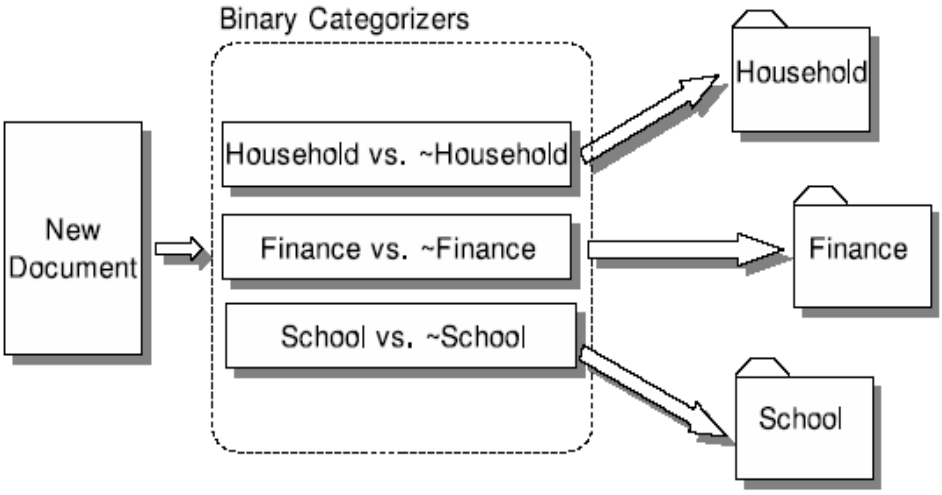


Figure 2.4 Text categorization

Figure 2.4 illustrates the document classification application. Documents are organized into folders, one folder for each topic. A new document is presented, and the objective is to place this document in the appropriate folders. For example, we might have a folder for household or financial documents and we want to add new documents to the correct folder. The application is almost always binary classification because a document can usually appear in multiple folders.

2.9.4. Information retrieval: Information retrieval is the topic most commonly associated with online documents. What is more fundamental to browsing the Internet than a search engine? The general task of information retrieval is illustrated in Fig. 2.5. A collection of documents is obtained, we give clues as to the documents that we want to retrieve from the collection, and then documents matching the clues are presented as answers to our query. The clues are words that help identify the relevant stored documents. In a typical instance of invoking a search engine, a few words are presented, and these words are matched to the stored documents. The best matches are presented as the responses. The process can be generalized to a document matcher, where instead of few words, a complete document is presented as a set of clues. The input document is then matched to all stored documents, retrieving the best-matched documents. A basic concept for information retrieval is measuring similarity: a comparison is made between two documents, measuring how similar the documents are. For comparison, even a small set of words input into a search engine can be considered as a document that can be matched to others. From one perspective, measuring similarity is related to predictive methods for learning and classification that are called nearest-neighbor methods. [Luhn, 1959, Maron and Kuhns, 1960, Salton, 64, Salton, 65, Salton, 71, Feldman and Hirsh, 1996a, Feldman and Hirsh, 1996b, Manning et al., 2008]

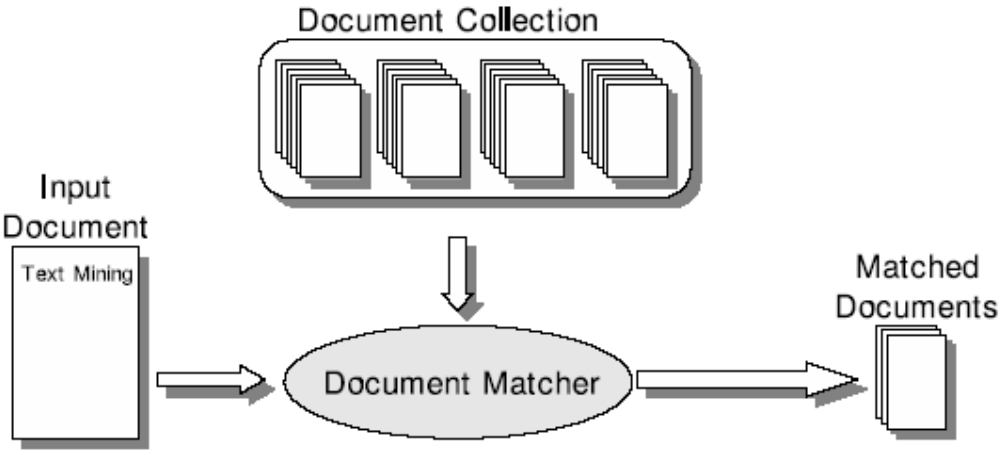


Figure 2.5. Information retrieval system

2.9.5. Clustering and organizing documents: For text categorization, we saw that the objective was to place new documents into Known folders created by someone who knew the expected topics. What if we have a collection of documents with no known structure? For example, a company may have a help desk that receives and records calls by users of their products. The company might want to learn about the categories and types of complaints. [Jardine and Rijsbergen, 1971]

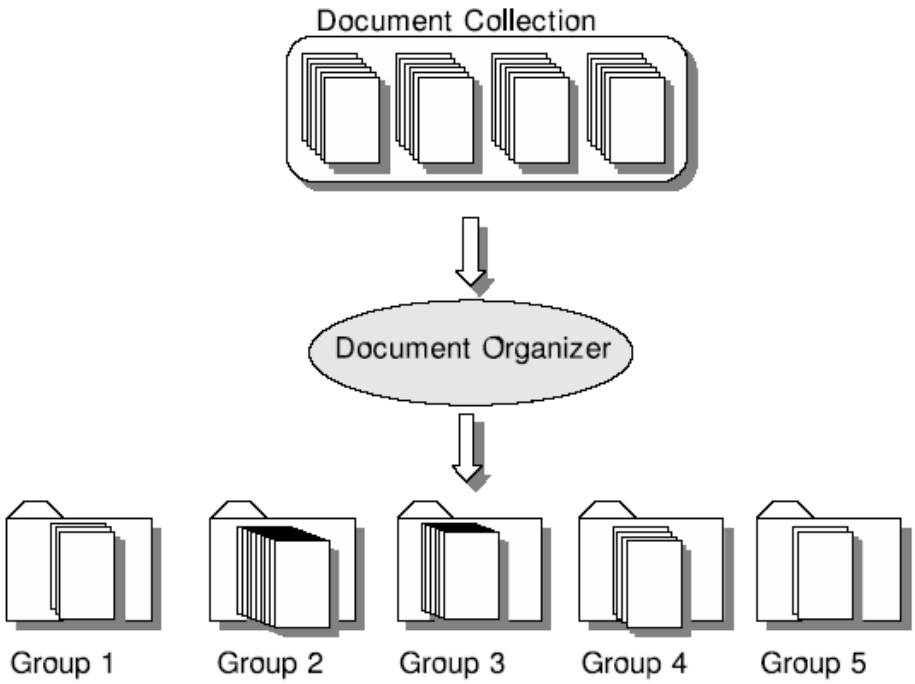


Figure. 2.6 Organizing documents into groups (clusters)

The general objective is illustrated in Figure 2.6. Given a collection of documents, find a set of folders such that each holds similar documents. The clustering process is equivalent to assigning the labels needed for text categorization. Because there are many ways to cluster documents, it is not quite as

powerful a process as assigning answers (i.e., known correct labels) to documents. Still, clustering can be insightful. In the help-desk example, a computer company might find that the largest cluster of complaints is for networking problems. It might also identify an unexpected type of problem, documents indicating many calls for help, for which they do not have a good solution.

***2.9.6. Information extraction*:** Data mining expects highly structured data, and text is naturally unstructured. To make text structured, we have employed a very shallow representation that measures the simple occurrence of words. Information extraction is a subfield of text mining that attempts to move text into an equal footing with the structured world of data mining. The objective is to take an unstructured document and automatically fill in the values of a spreadsheet. For example, we may examine documents about companies and extract the sales volumes and industry codes from text that has not been structured for storage in a database. The attribute that is measured will not have a fixed position in the text and may not be described in the same way in different documents. In terms of our spreadsheet model of data, the columns are not just words but can be higher-level concepts that are found by the information extraction process. This process is equivalent to populating a database table. Once the process is completed, the usual learning methods can be applied to the spreadsheet. [Cowie and Lehnert, 1996, Grishman, 1997, Feldman et al., 1998]

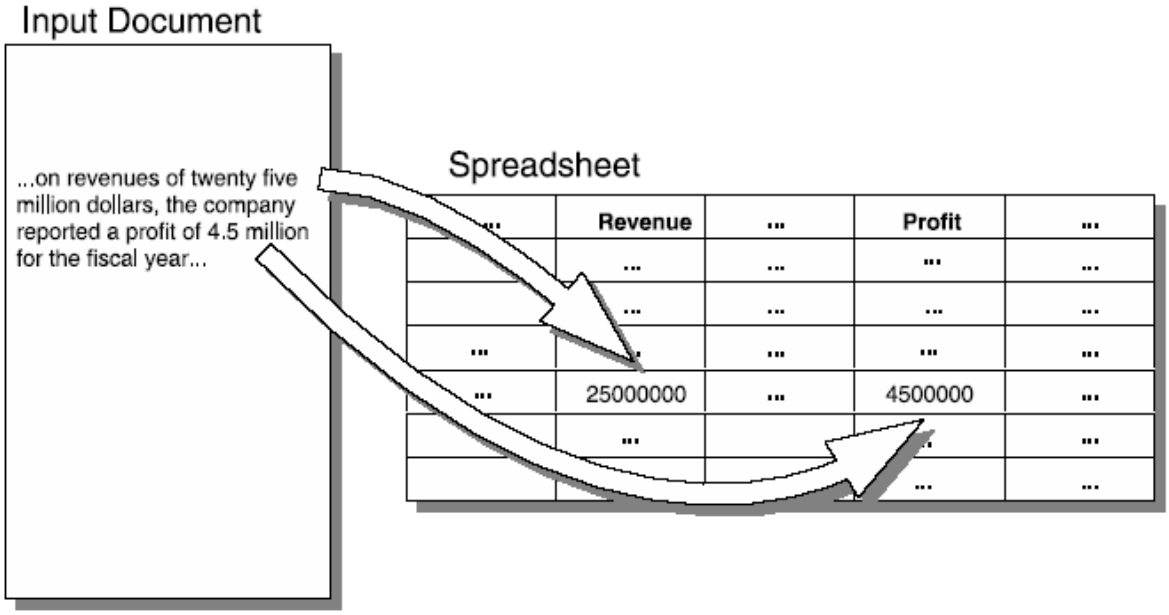


Figure 2.7 Extracting information from a document.

2.10. Architecture of text mining systems

2.10.1. General architecture

At an abstract level, a text mining system takes in input (raw documents) and generates various types of output (e.g., patterns, maps of connections, trends). [Brachman and Anand, 1996]

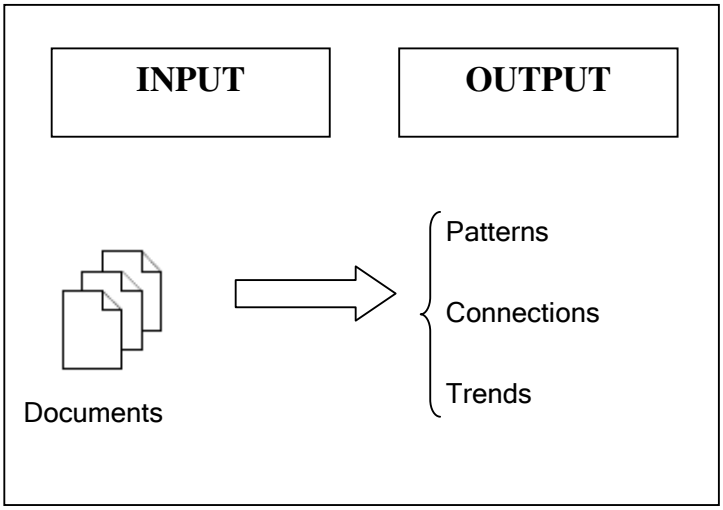


Figure 2.8. Simple input–output model for text mining.

2.10.2. Functional Architecture

On a functional level, text mining systems follow the general model provided by some classic data mining applications and are thus roughly divisible into four main areas: [Feldman and Sanger, 2007]

a. Preprocessing tasks.
b. Core mining operations.
c. Presentation layer components and browsing functionality.
d. Refinement techniques.

a). Preprocessing Tasks: include all those routines, processes, and methods required to prepare data for a text mining system's core knowledge discovery operations. These tasks typically center on data source preprocessing and categorization activities. Preprocessing tasks generally convert the information from each original data source into a canonical format before applying various types of feature extraction methods against these documents to create a new collection of documents fully represented by concepts.

b). Core Mining Operations: represent the heart of a text mining systems and include: pattern discovery, trend analysis, and incremental knowledge discovery algorithms. Among the commonly used patterns for knowledge discovery in textual data are distributions (and proportions), frequent and near frequent concept sets, and associations. Core mining operations can also concern themselves with comparisons between some of these patterns. Advanced or domain-oriented text mining systems, or both, can also augment the quality of their various operations by leveraging background knowledge sources.

c). Presentation Layer Components: include GUI and pattern browsing functionality as well as access to the query language. Visualization tools and user-facing query editors and optimizers also fall under this architectural category. Presentation layer components may include character-based or graphical tools for creating or modifying concept clusters as well as for creating annotated profiles for specific concepts or patterns.

d). Refinement Techniques: at their simplest, include methods that filter redundant information and cluster closely related data but may grow, in a given text mining system, to represent a full,

comprehensive suite of suppression, ordering, pruning, generalization, and clustering approaches aimed at discovery optimization. These techniques have also been described as post processing. Preprocessing tasks and core mining operations are the two most critical areas for any text mining system and typically describe serial processes within a generalized view of text mining system architecture.

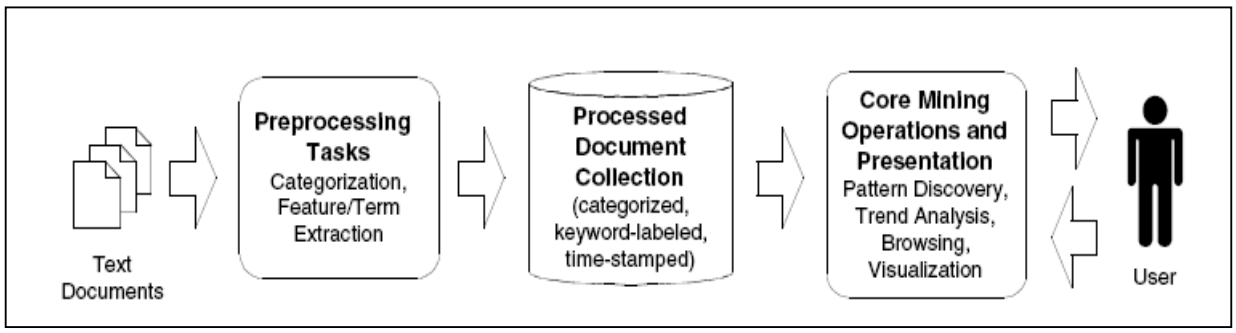


Figure 2.9. High-level text mining functional architecture.

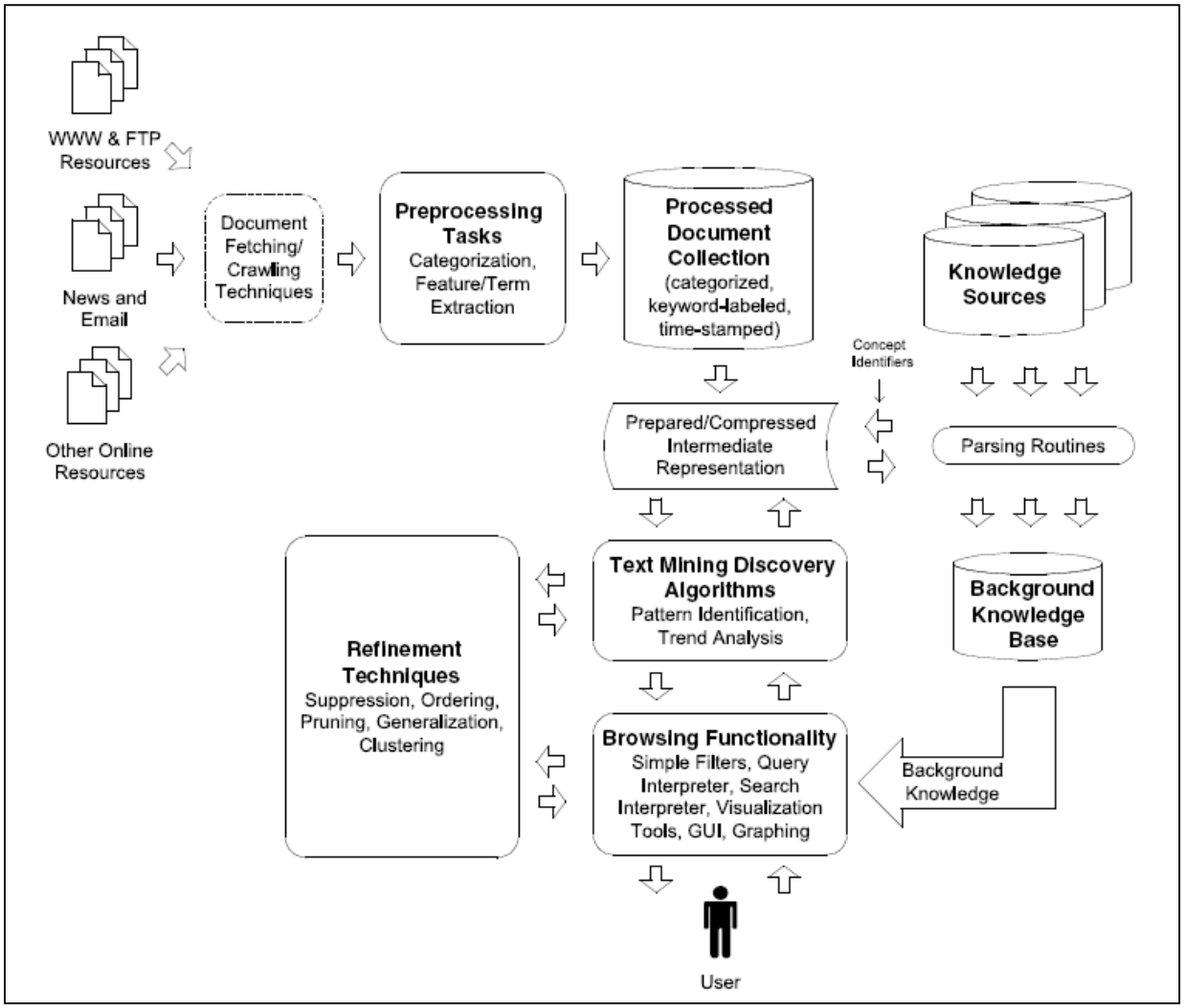


Figure 2.10. System architecture for an advanced or domain-oriented text mining system with background knowledge base.

2.11. Text Mining process step by step

Text mining process is constituted of many tasks as it is described in the following figure: [Paulheim et al, 2014]

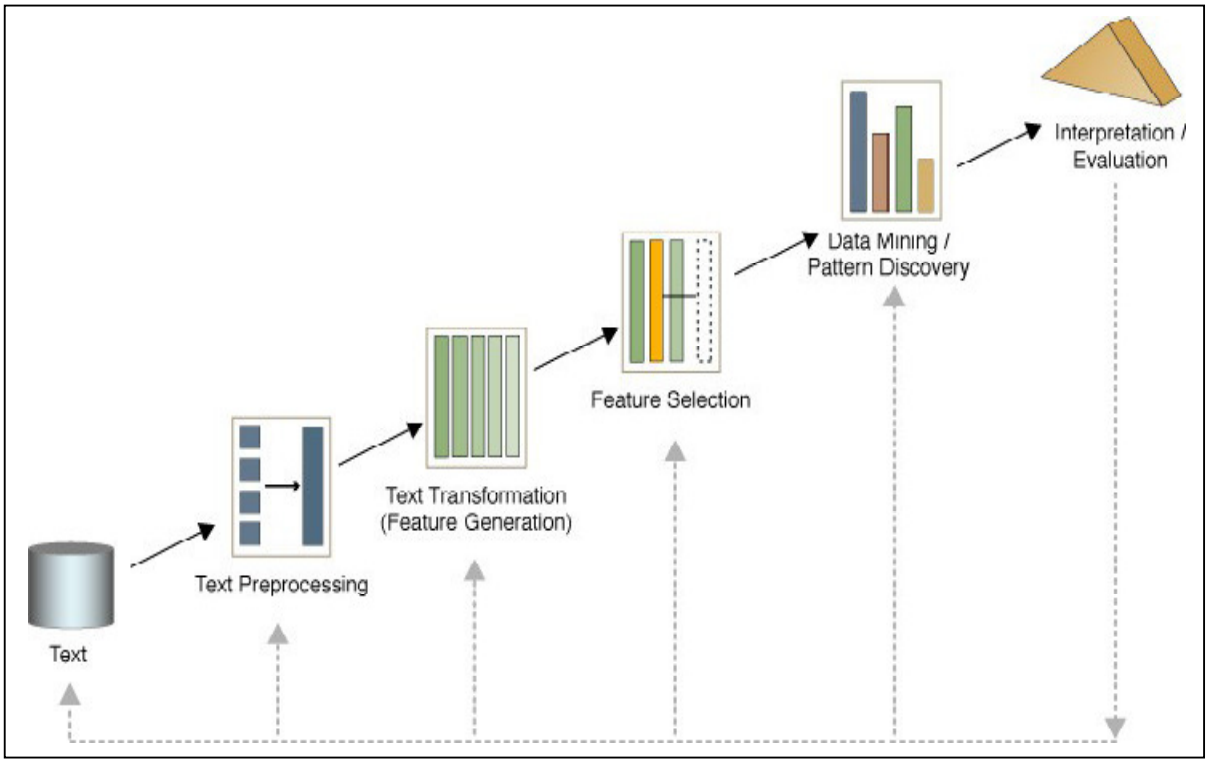


Figure 2.11. Description of text mining process

2.11.1. Collecting documents

The first step in text mining is to collect the relevant documents which may already be given or they may be part of the problem description. For example, the web pages on the internet for a retrieval application. If the documents are ready, the main issue is to cleanse the samples and ensure that they are of high quality. Sometimes, the documents may be obtained from document warehouses or databases. In this case, it is reasonable to expect that data cleansing was done before deposit and we can be confident in the quality of the documents. In some other applications, one may need to have a data collection process. For instance, for a Web application comprising a number of autonomous Web sites, one may deploy a software tool such as a Web crawler that collects the documents. Sometimes the set of documents can be extremely large and data-sampling techniques can be used to select a manageable set of relevant documents. These sampling techniques will depend on the application. The documents may also be more recent, more useful and have a higher relevance. For research and development of text-mining techniques, more generic data may be necessary. This is usually called a corpus. In this field, many collections of documents or corpora are used as it's shown in table 2.1.

Table 2.1. The most known document collections used in text mining field

The name of the Collection	Designation	Language	Source	Size	Date
RCV	Reuters news stories	English	Reuters Agency	806.791 docs	1987
20NewsGroup		English		11.314 docs	
BP	Brown Corpus	American English		01 million words	1950s – 1960s
LOB	Lancaster-Oslo-Bergen	British English	European Corpus		
PTB	Penn Tree Bank	English	A collection of parsed sentences from the Wall Street journal		1970s – 1980s
TREC	Text Retrieval and Evaluation Conferences	English	Selections from: Wall Street journal, New York Times, Ziff-Davis Publications, the federal register, others		
CPEF	The proceedings of the Canadian parliament in Parallel English-French translations	English			
GP	Gutenberg Project	English			
Reuters-21578 Dist 1.0	Collections of Reuters Stories	English	University of Pennsylvania, USA		1987, 1996
LDC	Linguistic Data Consortium	English			
ICAME	The International Computer Archive of Modern and Medieval English	English	Bergen, Norway		
TEI	The Text Encoding Initiative	English			
MEDLINE		English	National Institutes of Health		
PUBMED	The National Library of Medicine's online repository of citation-related information for biomedical research papers	English	National Center for Biotechnology Information (NCBI)- the U.S. National Institutes of Health (NIH)	12 millions papers	

Another resource to consider is the World Wide Web WWW itself. Web crawlers can build collections of pages from a particular site such as yahoo, or from the archives of USENET news groups. The main problem with this approach of document collections is the data may be of dubious quality and require extensive cleansing before use. Finally, institutions such as government agencies and corporations often have large document collections. Corporate collections are usually not available outside the corporation, but government collections often are.

2.11.2. Text Preprocessing tasks

Include all those routines, processes, and methods required to prepare data for a text mining system's core knowledge discovery operations. These tasks typically center on data source preprocessing and categorization activities. Preprocessing tasks generally convert the information from each original data source into a canonical format before applying various types of feature extraction methods against these documents to create a new collection of documents fully represented by concepts. A large variety of preprocessing tasks must be done before applying any predictive methods on the document, among these tasks: [Weiss et al., 2010]

2.11.2.1. ***Document Standardization:*** The collected documents are often in a variety of formats such as: word document, ASCII format, PDF format and others. Clearly if we are to process all the documents, it's helpful to convert them to a standard format. The computer science community has adopted XML (Extensible Markup Language) as a standard format. The main advantage of standardizing the data is that the mining tools can be applied without having to consider the original format of the document.

2.11.2.2. **Tokenization:** Prior to more sophisticated processing, the continuous character stream must be segmented into meaningful constituents. This can occur at several different levels. For example, documents can be segmented into chapters, sections, paragraphs, sentences, words, n-grams, characters and even syllables or phonemes. The approach most frequently found in text mining systems involves breaking the text into sentences, words and n-grams, which is called tokenization. The main challenge in identifying sentence boundaries especially in the English language is distinguishing between a period that signals the end of a sentence and a period that is part of a previous token like: Mr., Dr., O'Malley, 555-1212, and so on. Some languages do not have white space as delimiter such as: Chinese, Japanese, Korean; German. It is common for the tokenizer also to extract token features which are usually simple categorical functions about the tokens describing some superficial property of the sequence of characters that make up the token. Among these features are: types of capitalization, inclusion of digits, punctuation, special characters, and so on.

2.11.2.3. **Simple Syntactic Analysis:** Comprise several small tasks, such as:

1. Text Cleanup: consists in removing useless characters, such as:
   - ✓ Punctuation (".", ",", ";", "?", "!", ":")
   - ✓ HTML tags (<HTML>, </HTML>, <HEAD>, </HEAD>, <BODY>, </BODY>, <TITLE>, </TITLE>, <AUTHOR>, </AUTHOR>, etc.
2. Normalize case: Make all lower case if don't care about proper nouns, titles, etc.
   (e.g., "TO BE or do NOT be" → "to be or do not be")
3. Clean up transcription and typing errors: (e.g., do n't, movei,...)
4. Correct misspelled words:
   - Phonetically: Use fuzzy matching algorithms such Soundex, Metaphone, or string-edit distance.
   - Dictionaries: Use POS and context to make good guess
5. Stop words removal: Many of the most frequently used words in English are likely to be useless for text mining, these words are called stop words. (Ex: the, of, and, to, an, is, that, but, by, for, such, etc). Typically a text contains about 400 to 500 different stop words, for an application, an additional domain specific stop words list may be constructed. We note also that stop words are very common and rarely provide useful information for information extraction or concept extraction.

Advantages of stop words removal

- Reduce data set size: because stop words account for 20-30% of total word counts.
- Improve efficiency and effectiveness of predictive algorithms.
- Stop words may confuse the mining algorithm.
- Just like irrelevant features in standard data mining

2.11.2.4. Advanced Linguistic Analysis

a. Part Of Speech (POS) tagging

Once a text has been segmented into tokens and sentences, the next step depends on what is to be done with the text. If no further linguistic analysis is necessary, one might proceed directly to feature generation, in which the features will be obtained directly from the tokens. However, if the goal is more specific, say recognizing names of people, places, titles and organizations, it is usually desirable to perform additional linguistic analyses of the text and extract more sophisticated features. Toward this end, the next logical step is to determine the part of speech (POS) of each token [Maltese and Mancini, 1991, Kupiec, 1992, Weiss et al., 2010].

POS tagging is the annotation of words with the appropriate POS tags based on the context in which they appear. POS tags divide words into categories based on the role they play in the sentence in which they appear. POS tags provide information about the semantic content of a word. Nouns usually denote "tangible and intangible things," whereas prepositions express relationships between "things". Most POS tag sets make use of the same basic categories. The most common set of tags contains seven different tags (Article, Noun, Verb, Adjective, Preposition, Number, and Proper Noun). Some systems contain a much more elaborate set of tags [Schutze, 1993, Brill, 1995, Feldman and Sanger, 2007]. For example: Penn Tree Bank contains 36 tags, the complete Brown Corpus tag set has no less than 87 basic tags. As summary, pos tagging or:

- ✓ Recognizing names of people, places, organizations, titles.
- ✓ Finding the structure of a sentence and parsing it according to grammar.
- ✓ Determine word classes and syntactic function of each word.

Examples: "John (noun) gave (verb) the (det) ball (noun)".
"That (det) man (noun) read (verb) this (det) book (noun)"

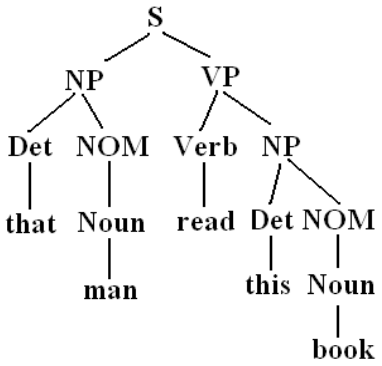


Figure 2.12. POS tagging of sentence words

Sometimes, multiple results are possible, and thus language is ambiguous!

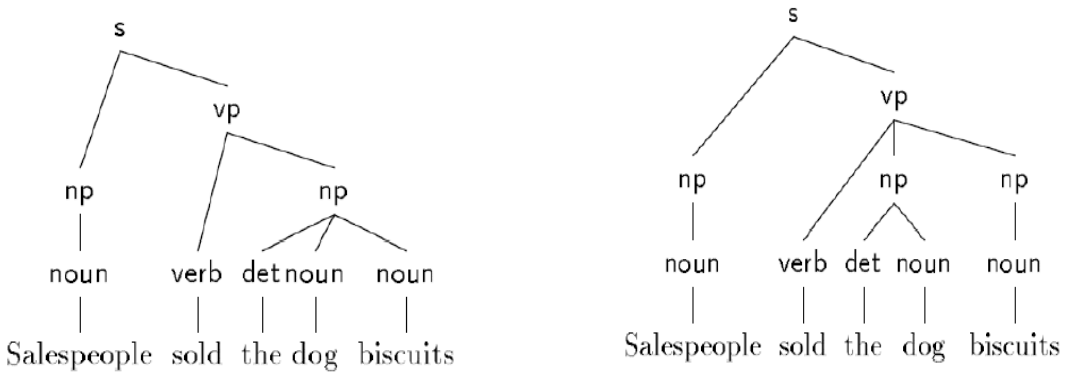


Figure 2.13. Multiple POS tagging for the same sentence

Table 2.2 represents one of the most known set of categories which is the Penn Tree Bank constructed from the Wall Street journal. [Charniak, 1997, Ratnaparkhi, 1999, Chiang, 2000]

Table 2.2 Some categories in the Penn tree bank POS set (source: [Weiss et al., 2010])

Tag	Description	Tag	Description
CC	Coordinating conjunction	NN	Noun, singular or mass
CD	Cardinal number	NNS	Noun, plural
DT	Determiner	NNP	Proper noun, singular
EX	Existential there	NNPS	Proper noun, plural
FW	Foreign word	POS	Possessive ending
IN	Preposition/subordinating conjunction	PRP	Personal pronoun
JJ	Adjective	RB	Adverb
VB	Verb, base form	RBR	Adverb, comparative
VBD	Verb, past tense	RP	Particle
VBG	Verb, gerund or present participle	VBZ	Verb, 3rd pers singular present
VBN	Verb, past participle	WDT	Wh-determiner
VBP	Verb, non-3rd person singular present	WP	Wh-Pronoun

Dictionaries showing (word, POS) correspondence can be useful but are not sufficient. All dictionaries have gaps, but even for words found in the dictionary, several parts of speech are usually possible. For example, the word “bore” could be a noun, a present tense verb, or a past tense verb. The goal of POS tagging is to determine which of these possibilities is realized in a particular text instance. Although it is possible, in principle, to manually construct a part-of-speech tagger, the most successful systems are generated automatically by machine-learning algorithms from annotated corpora. Almost all POS taggers have been trained on the Wall Street Journal corpus available from LDC (Linguistic Data Corporation, www.ldc.upenn.edu) because it is the most easily available large annotated corpus.

Example 2.7: Example of Brill Tagging [Brill, 1995, Ratnaparkhi , 1995, Abbott, 2013]

> *In this talk, Mr. Pole discussed how Target was using Predictive Analytics including descriptions of using potential value models, coupon models, and...yes...predicting when a woman is due.*

Figure 2.14. Text before POS tagging.

> ***In/IN this/DT talk/NN ,/, Mr./NNP Pole/NNP discussed/ VBD how/WRB Target/NNP was/VBD using/VBG Predictive/NNP Analytics/NNP including/VBG descriptions/NNS of/IN using/VBG potential/JJ value/NN models/NNS ,/, coupon/NN models/NNS ,/, and...yes...predicting/VBG when/WRB a/DT woman/NN is /VBZdue/JJ./.***

Figure 2.15. Text after POS tagging.

Example 2.8: Example of stochastic tagging

1)	*Secretariat*	*is*	*expected*	*to*	*race*	*tomorrow*
	(NNP)	(VBZ)	(VBN)	(TO)	(VB)	(NR)

2)	*Secretariat*	*is*	*expected*	*to*	*race*	*tomorrow*
	(NNP)	(VBZ)	(VBN)	(TO)	(NN)	(NR)

P(NN|race) = 0.00047
P(VB|race) = 0.83 ➔ "race" is most likely a verb

Characteristics of POS tagging

1. ≈ 89% of English words have only one part of speech (unambiguous).
   - ✓ However, many common words in English are ambiguous.
   - ✓ But even the secan largely be disambiguated by rules or probabilistically
2. Taggers can be rule-based, stochastic (training on a labeled set of words using HMMs),or a combination (most popular combination is the "Brill" tagger)

b. Syntactical parsing

Syntactical parsing components perform a full syntactical analysis of sentences according to a certain grammar theory. The basic division is between the constituency and dependency grammars. Constituency grammars describe the syntactical structure of sentences in terms of recursively built phrases – sequences of syntactically grouped elements. Most constituency grammars distinguish between noun phrases, verb phrases, prepositional phrases, adjective phrases, and clauses. Additionally, the syntactical structure of sentences includes the roles of different phrases. Thus, a noun phrase may be labeled as the subject of the sentence, its direct object, or the complement [Feldman and Sanger, 2007, Jurafsky and Martin, 2008, Indurkhya and Damerau, 2010].

Dependency grammars, on the other hand, do not recognize the constituents as separate linguistic units but focus instead on the direct relations between words. A typical dependency analysis of a sentence consists of a labeled DAG (Dependency Analysis Graph) with words for nodes and specific relationships (dependencies) for edges. For instance, a subject and direct object nouns of a typical sentence depend on the main verb, an adjective depends on the noun it modifies, and so on.

c. Shallow Parsing

Standard algorithms are too expensive for use on very large corpora and are not robust enough. Shallow parsing compromises speed and robustness of processing by sacrificing depth of analysis. Instead of providing a complete analysis (a parse) of a whole sentence, shallow parsers produce only parts that are easy and unambiguous. Typically, small and simple noun and verb phrases are generated, whereas more complex clauses are not formed [Tzoukermann et al., 1997, Munoz et al., 1999, Punyakanok and Roth, 2000].

Similarly, most prominent dependencies might be formed, but unclear and ambiguous ones are left unresolved. For the purposes of information extraction, shallow parsing is usually sufficient and therefore preferable to full analysis because of its far greater speed and robustness.

d. Word Sense Disambiguation

The main objective of this phase is to determine which sense a word has [Ide and Véronis, 1998]. English words for example, besides being ambiguous when isolated from their POS status, are also very often ambiguous as to their meaning or reference. If we take for example the word "*bore*," one cannot tell without context, even after POS tagging, if the word is referring to a person ("*he is a bore*") or a reference to a hole, as in "*the bore is not large enough*." The main function of ordinary dictionaries is to catalog the various meanings of a word, but they are not organized for use by a computer program for disambiguation. One of the projects that focused on word meanings and their interrelationships is Wordnet, which aimed to fill in this gap. As useful as Wordnet is, by itself it does not provide an algorithm for selecting a particular meaning for a word in context [Weiss et al., 2010]. In spite of substantial work over a long period of time, there are no algorithms that can completely disambiguate a text. Many other objectives are expected when proceeding to this phase, notably:

1. Normalize synonyms: for this end, we can use simple and large-scale catalogs such as WordNet, Wikipedia Surface Forms, or normalized forms extracted from titles of Wikipedia pages (e.g., (United States → USA, US)
2. Normalize pronouns: make correspondence between pronouns and the real names they replace. (e.g., "he" → "Nelson Mandilla"

Example 2.9: Word Sense Disambiguation

"The music of Nine Inch Nails has reportedly been used by the U.S. military as music torture to break down the resolve of detainees."

(e.g., extracted normalization pattern: "U.S. military" → "United States Armed Forces")

We note here that in synonym normalization, catalogs work great for common knowledge, but not so well for special domains ➔ possible solution: calculation of string similarity using: [Paulheim, 2014]

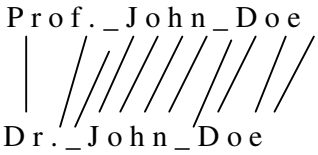
1. ***Edit distance:*** the edit distance is the minimum number of edits needed to transform one string into the other, among the allowed edit operations we can note:
   - Insert a character into the string.
   - Delete a character from the string.
   - Replace one character with a different character.

Example 2.10: Edit distance

- Levenshtein('John Smith', 'John K. Smith ') = 3 (3 inserts: ('K', '.', ' '))
- Levenshtein('John Smith', 'Jack Smith') = 3 (3 substitutions: (o, a), (h, c), (n, k))

2. ***Jaro Distance:*** This distance is characterized by:
   - Measures the dissimilarity of two strings.
   - Developed for name comparison in the U.S. Census.
   - Optimized for comparing person names.
   - Based on the number of common characters within a specific distance.

Example 2.11: Jaro Distance



3. ***N-gram Based Similarity:*** This similarity is defined as follows:
   - Measures the similarity of two strings.
   - Split string into set of trigrams:
     – e.g., "similarity" ➔➔ "sim", "imi","mil","ila", "lar", ..,
   - measure overlap of trigrams
     – e.g., Jaccard overlap: |common trigrams| OR |all trigrams|

Example 2.12: N-gram Based Similarity

1. *"Clustering similar product offers on eBay"*
2. *"iPhone5 Apple" vs. "Apple iPhone 5"*
   – Common trigrams: "iPh", "Pho", "hon", "one", "App", "ppl", "ple"
   – Other trigrams: "ne5", "e5 ", "5 A", " Ap" (1), "le ", "e i", " iP", "e 5" (2)
   – Jaccard = (common trigrams/all trigrams) = 7/15 = 0.47

2.10.2.5. Lemmatization and stemming

Once a character stream has been segmented into a sequence of tokens, the next possible step is to convert each of the tokens to a standard form. This is usually referred to as stemming or lemmatization task. Stemming is a technique that permits to find out the root/stem of a word. [Schmid, 1994, Hull, 1996, Xu and Croft, 1998, Porter, 1980]

Example 2.13: Example of stems

1. *Words: User, users, used, using → Stem: use*
2. *Words: walking, walks, walked, walker => Stem: walk).*

Lemmatization consists in converting verbs to the infinitive form, and nouns into singular form.

Example 2.14: Example lemmas

(written → write), (opened → open), (computers →computer).

Stemming and lemmatization provide many advantages:

- ✓ Are useful for Text Mining.
- ✓ Improve the effectiveness of text mining methods.
- ✓ Reduce term vector size by matching similar words (40-50%).
- ✓ Increase the frequency of occurrence of some individual types.

There exist many stemming methods:

a) ***Lookup-based Stemming:*** one way to do stemming is to ***c***reate a lookup table with all base form terms and their inflected forms (e.g., WordNet, Wiktionary, etc).

Example 2.15: Lookup-based stemming

Table 2.3. Lookup-based stemming

Base Form	Inflected forms
Move	moves, moved, moving
go	goes, went, gone, going
Apple	apples
…	…

b). Rule-base stemming (affix removal stemming): this method removes affixes (suffixes or prefixes) from words to convert them into common stem. For example porter stemmer, use many rules to find the stem of a given word as it will be explained bellow: [Dawson, 1974, Porter, 1980]

- Removes word endings
  - – If a word ends with a consonant other than s, followed by an s, then delete s (puts → put)
  - – If a word ends in es, drop the s (uses → use)
  - – If a word ends in ing, delete the ing unless the remaining word consists only of one letter or of two (reading → read)
  - – If a word ends with ed, preceded by a consonant, delete the ed unless this leaves only a single letter (founded → found)
- Transform words
  - – If a word ends with "ies" but not "eies" or "aies" then "ies" → "y" (flies → fly)

– ...

c) ***Inflectional stemming (lemmatization):*** In English, as in many other languages, words occur in text in more than one form. For example: "book" and "books" are two forms of the same word. Often, but not always, it is advantageous to eliminate this kind of variation before further processing (i.e., to normalize both words to the single form "book"). When the normalization is confined to regularizing grammatical variants such as singular/plural and present/past, the process is called "inflectional stemming". In some languages, such as Spanish, morphological analysis is comparatively simple. For a language such as English, with many irregular word forms and non intuitive spelling, it is more difficult. There is no simple rule, for example, to bring together "seek" and "sought." In other languages, inflections can take the form of infixing, as, in the German "angeben" (declare), for which the past participle is "angegeben.". The inflectional stemmer is not expected to be perfect, it will correctly identify quite a significant number of stems.

d) ***Stemming to a root:*** The goal of this kind of stemming is to reduce word to its root form with no inflectional or derivational prefixes and suffixes. For example, "denormalization" is reduced to the stem "norm." The end result of such aggressive stemming is to reduce the number of types in a text collection very drastically, thereby making distributional statistics more reliable.
Additionally, words with the same core meaning are combined, so that a concept such as "apply" has only one stem, although the text may have "reapplied", "applications", etc. We cannot make any broad recommendations as to when or when not to use such stemmers because the usefulness of stemming is very much application dependent.

2.10.3. Feature Generation and data representation

For a problem of document categorization, the characteristic features of documents are the tokens (words, phrases, n-grams) they contain. So, we must perform a deep linguistic analysis on the documents. This allows us to extract the features that represent the most frequent tokens in the documents (see the algorithm bellow). The collected set of features is called features dictionary. The tokens of this dictionary form the basis to create a spreadsheet of numeric data corresponding to the document collection. Each row is a document, and each column represents a feature. Thus, a cell in the spreadsheet is a measurement of a feature for a document. We will soon introduce the predictive methods that learn from the obtained spreadsheet [Lewis, 1992, Apté *et al.,* 1994].

```
Input:
    ts, all the tokens in the document collection
    k, the number of features desired
Output:
    fs, a set of k features
Initialize:
    hs := empty hashtable
for each tok in ts do
  If hs contains tok then
     i := value of tok in hs
     increment i by 1
  else
    i := 1
  endif
    store i as value of tok in hs
endfor
 sk := keys in hs sorted by decreasing value
 fs := top k keys in sk
output fs
```

Algorithm. 2.1 Generating features from tokens

2.10.3.1. Global dictionary vs. local Dictionary

If a learning method can deal with the high dimensions of such vector, we use a global dictionary which contains a huge number of tokens extracted from the document collection. This simple model of data can be very effective. Sometimes we may work with a smaller dictionary. In such cases, we might try to reduce the size of the global dictionary by various transformations on its words. Among these transformations we can note: the use of a local dictionary associated to each predicted class, the removal of stopwords, the use of frequent words, feature selection and stemming.

2.10.3.2. Features reduction

One way to reduce the features of the global dictionary is to consider only words found in the class that we are trying to predict. This will give us a reduced dictionary generally called a local dictionary. Another obvious reduction in dictionary size is to remove stop words. These are words that almost never have any predictive capability as we have seen in section 2.10.2.3. This task can be performed before the feature generation process, but it's more effective to generate the features first, apply all the other transformations, and at the very last stage reject the ones that correspond to stop words. The word frequency can be quite useful in reducing dictionary size and sometimes improves predictive performance for some methods. In this case, the most frequent words are often stopwords and can be deleted. The remaining most frequently used words are often the important words that should remain in a local dictionary. The very rare words are often typos and can also be dismissed. For some learning methods, a local dictionary of the most frequent words, perhaps less than 200, can be surprisingly effective [Forman, 2003].

2.10.3.3. Data representation

In machine learning approach, it's quite difficult to apply such learning algorithm directly on a text as a sequence of characters. Thus, it's necessary to proceed to a preliminary phase consisting in the representation of each text by a vector of terms. Terms can be: words, phrases, or any other lexemes. After that, we must assign a numerical value to each term called term weight. There are many models to represent textual data on the spreadsheet, notably: [Salton, 1975, Salton, 1980]

a) Binary model: The most basic model is the binary model, in which each document is treated as "a bag" of words or terms. Word order is not considered. Given a collection of documents D, and let $V=\{t_1, t_2, \ldots, t_{|V|}\}$ be the set of distinctive words/terms in the collection. V is called the collection vocabulary. A term weight $w_{ij}=1$ is associated with each term t_i that appears in the document $d_j \in D$. for the term that doesn't appear in the document d_j, the weight $w_{ij} = 0$. So, in the binary model, we simply check for the presence or the absence of word/term in the document. Thus, for a single document d_j, the representing vector is given by: $dj = (w_{1j}, w_{2j}, \ldots, w_{|V|j})$.

For a set of documents (a collection), we obtain a matrix of ones and zeros as it's given bellow:

	Company	Income	Job	Overseas
D1	0	1	0	1
D2	1	0	1	1
D3	1	1	1	0
D4	0	0	0	1

Figure 2.16. A Binary model representation of a collection of Documents

b) Three values model: another model to map the presence of words in a document is the three valued model. In this model, 0 expresses that the word didn't occur, 1 the word occurred once, 2 the word occurred 2 or more times.

Table 2.4 Thresholding frequencies to three values model.

Value	Description
0	Word didn't occur
1	Word occurred once
2	Word occurred 2 or more times

Another variant of this model involves zeroing all values below a certain threshold. That means tokens should have a minimum frequency before being considered of any use. This can reduce the complexity of the spreadsheet significantly and might be a necessity for some predictive methods.

c) Term frequency model (tf): Instead of zeros or ones as entries in the cells of the spreadsheet, the frequency of occurrence of the word could be used. If a word occurs ten (10) times in a document, this count would be entered in the cell. For some learning methods, the count does give a slightly better result. Overall, the frequencies are helpful in prediction but add complexity to the proposed solutions.

	Company	Income	Job	Overseas
D1	0	5	0	2
D2	7	0	11	3
D3	10	1	3	0
D4	0	0	0	4

Figure 2.17. term frequency model.

d) Tf-idf model (Term Frequency-Inverse Document Frequency): This model is focused on the perceived importance of the word. The well-known *tf-idf* formulation has been used to achieve this goal. We also note that this specific frequency takes positive values. So, that we capture the presence or absence of the word in a document. [Salton and Buckley, 1987, Salton and Buckley, 1988, Joachims, 1997, Joachims, 1998]

$$tf - idf_i = tf_{i,j} * idf_i \tag{2.1}$$

Where:
$Tf_{i,j}$: Frequency of the term *i* in the document *j*.
idf_i: Inverse Document Frequency of the term *i*.

We have seen in (2.1) that the *tf-idf* weight assigned to word *i* is the term frequency $tf_{i,j}$ (i.e., the word count) modified by a scale factor for the importance of the word. The scale factor is called the Inverse Document Frequency *idf*, which is given in (2.2) and (2.3).

$$idf_i = Log[\frac{The\ total\ number\ of\ documents\ in\ the\ collection}{the\ number\ of\ documentss\ with\ the\ term\ i}] \tag{2.2}$$

Thus, we can write:

$$idf_i = \log(\frac{N}{df_i}) \tag{2.3}$$

Idf simply checks the number of documents containing the word *i* and reverses the scaling. Thus, when a word appears in many documents such as stop words, it is considered less important and the scale is lowered, perhaps near zero. When the word is relatively unique and appears in few documents, the scale factor has a high value and the word in this case is more important.
Another variant of weighting models is to weight the tokens from different parts of the document differently. For example, the words in the subject line of a document could receive additional weight. An effective variant is to generate separate sets of features for each category (the set of features is derived only from the tokens of documents of that category) and then pool all the feature sets together.

All of these models of data are modest variations of the basic binary model for the presence or absence of words. [Manning et al., 2008]

Although we describe data as populating a spreadsheet, we expect that most of the cells will be zero. Most documents contain a small subset of the dictionary's words. For example, in the case of text classification, a text corpus (collection) might have thousands of word types. But, each individual document, however, has only a few hundred unique tokens. So, in the spreadsheet, almost all of the entries for that document will be zero. Rather than store all the zeros, it is better to represent the spreadsheet as a set of sparse vectors, where a row is represented by a list of pairs, one element of the pair being a column number and the other element being the corresponding nonzero feature value. By not storing the zeros, savings in memory can be immense. We note that all of our proposed data representations are consistent with such a sparse data representation.

0	15	0	3
12	0	0	0
8	0	5	2

(2, 15) (4,3)
(1, 12)
(1, 8) (3, 5) (4, 2)

Figure. 2.18 Transformation of spreadsheet to sparse vectors

2.10.3.4. Multiword Features

Generally, features are associated with single words (e.g., tokens delimited by white space). There are cases where it helps to consider a group of words as a feature. This happens when a number of words are used to describe a concept that must be made into a feature. The simplest scenario is where the feature space is extended to include pairs of words. In this case, we can consider not only pairs of words, but more general multiword features. The most common example is a named entity (ex: Don Smith or United States of America). Unlike word pairs, the words need not necessarily be consecutive. Another example of a multiword feature is an adjective followed by a noun, such as "broken vase". In this case, we must permit some flexibility in the distance between the adjective and noun (ex: broken and dirty vase). Thus, more generally, multiword features consist of x number of words occurring within a maximum window size of y (with $y \geq x$). [Weiss et al, 2010]

The key question is how such features can be extracted from text? Named entities can be extracted using specialized methods. For other multiword features, a more general approach might be to treat them like single-word features. If we use a frequency approach, then we will only include those combinations of words that occur relatively frequently.

Generally, multiword features are not found too frequently in a document collection, but when they do occur they are often highly predictive. They are also particularly satisfying for explaining a learning method's proposed solution. The downside to using multiword is that they add an additional layer of complexity to the processing of text, and some practitioners may feel it's the job of the learning methods to combine the words without a preprocessing step to compose multiword features.

2.10.3.5. Labels for the Right Answers

For prediction, an extra column must be added to the spreadsheet. This last one contains the label. It is 1 or 0 indicating that the correct answer is either true or false.

What is the label? This label represents a topic or category to which the document will be assigned (e.g., Sports or financial stories). Any other answer that can be measured as true or false is acceptable. In the sparse vector format, the labels are appended to each vector separately as either 1 (positive class) or 0 (negative class).

2.10.3.6. Named Entity Recognition (NER)

The task of named entity recognition (NER) requires a program to process a text and identify expressions that refer to people, places, companies, organizations, products, and so forth. Thus the program should not merely identify the boundaries of a naming expression, but also classify the expression, e.g., so that one knows that "Los Angeles" refers to a city and not a person. This is not as easy as one might think. Most commercially available software packages for NER concentrate upon identifying proper names that refer to people, places and companies. They may also try and find relationships between entities, e.g., "Bill Gates, President of Microsoft Corporation" will yield the person Bill Gates standing in a President relationship to the company Microsoft. A variety of methods are used to achieve such extractions, which we shall now summarize. The best MUC-7 system came from Edinburgh University and employed a variety of methods, combining lists, rules, and probabilistic techniques, applied in a particular order. Modern systems, such as CogNIAC, offer 90% or better precision on common pronoun usages that do not require either specialist knowledge or general knowledge for their resolution. CogNIAC works by performing a linguistic analysis and then applying a set of decision rules to the analyzed text. [Bikel *et al.,* 1997, Bikel *et al.,* 1998]

Example2.16: Named Entity Recognition

– "Stock quote of Apple Inc. expected to exceed $600."
⇔ "Stock quote of <*ORGANIZATION*>Apple Inc.</*ORGANIZATION*>expected to exceed <*AMOUNT*>$600</*AMOUNT*>."

2.10.4. Feature selection:

After generating a global dictionary, special feature selection methods can be used to select a subset of words that appear to have the greatest potential for prediction. Among the most popular methods are: the mutual information MI [Lewis, 1992, Mouliner, 1997, Dumais *et al.*, 1998], the information gain IG [Yang and Pedersen, 1997, Sebastiani, 2002], chi-square law χ^2 [Schütze *et al.*, 1995, Wiener *et al.*, 1995], odds ratio, and others. These selection methods are often complicated and independent of the prediction method. Any of the feature selection methods that have been used in alternative statistical or machine-learning settings may be tried. Many of these methods have been developed for real variables and without an emphasis on discrete or binary attributes. [Sahami, 1998, Zaragoza, 1999]
Lemmatization and stemming techniques can occasionally be harmful for some words. Overall, stemming will achieve a large reduction in dictionary size and is modestly beneficial for predictive performance when using a smaller dictionary. All these transformations will improve the manageability of learning and perhaps improve accuracy. If nothing else is gained, learning can proceed more rapidly with smaller dictionaries. Once the set of features has been determined, the document collection can be converted to spreadsheet format.

2.10.5. Data Mining (pattern discovery)

It's the most important, critical and complex phase in text mining process. It consists in building a predictive model from texts and using it in making strategic decisions. From a statistical perspective, it's a straightforward problem that has a solution. Of course, the solution may not always be very good. It's simple to define the problem, given a sample of examples of past experience, we project to new examples. If the future is similar to the past, we may have an opportunity to make accurate predictions. But, making a prediction requires more than a lookup of past experience because the test of success is on new examples. Therefore, the past experience provide patterns that will hold in the future, leading

to accurate results on new, unseen examples. Another thing is if a new example is radically different from prior experience, learning from past experience becomes inadequate. We note also that if samples Are organized in the right format, finding patterns is almost effortless, even in very high-dimensional feature spaces. Finally, Prediction from text can be just as ambitious as prediction for numerical data mining. One classical prediction problem for text is called text categorization. Here, a set of categories is predefined, and the objective is to assign a category or topic to a new document. Many examples of problems be presented in this phase, including: [Paulheim, 2014, Kumar, 2007, Steinbach, 2004]

2.10.5.1. ***Classification:*** called also text categorization. It's a classical prediction problem in text mining field, its principle is as follows:

- Given: set of predefined categories (labels), a collection of labeled documents (training set)
- Find: Model for the class (as a function of the values of the features).
- Goal: Previously unseen documents should be assigned a class or topic as accurately as possible (test set/new document).

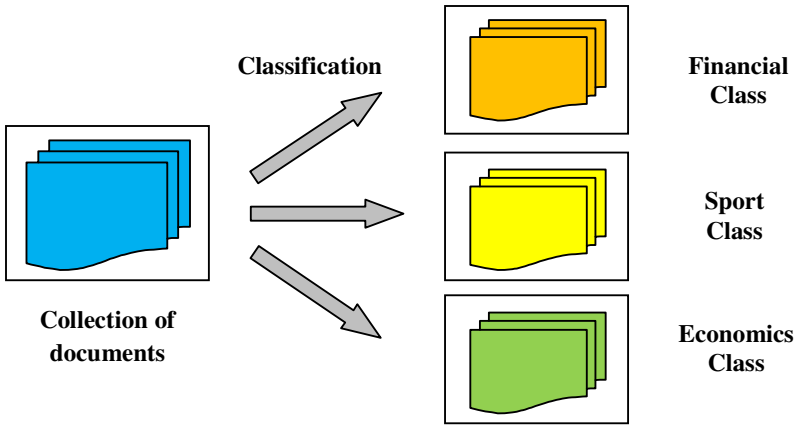


Figure 2.19. Description of classification problem

Several methods are commonly used for text classification such as:

- Naive Bayes NB
- kNN
- Support Vector Machines SVM
- Decision Trees may also work
- Neuronal network
- Others

2.10.5.2. ***Clustering:*** given a set of documents and a similarity measure among documents, and try to find small groups (sub-collections of documents) called clusters such that:

- Documents in one cluster are more similar to one another.
- Documents in separate clusters are less similar to one another.
- Labels of clusters are not predefined by humans as in text categorization problem, but defined automatically by the classifier.
- The similarity measure is used to define different clusters and assign documents to them.

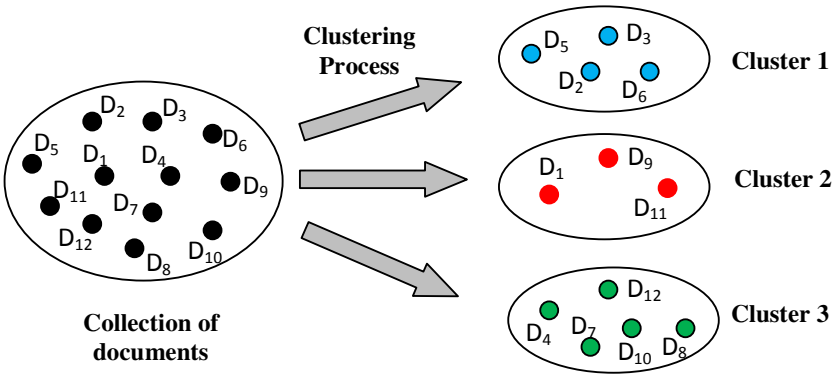


Figure 2.20. Description of clustering problem

Many similarity measures are used to compare document vectors. Among them:

a) Jaccard Coefficient: Jaccard coefficient is a popular measure defined as follows:

$$Dist\left(d_i, d_j\right) = \frac{M_{11}}{M_{01}+M_{10}+M_{11}} \tag{2.4}$$

$$Dist\left(d_i, d_j\right) = \frac{Number\ of\ (11)\ matches}{number\ of\ not-both-zero\ attributes\ values} \tag{2.5}$$

Example 2.17: Jaccard Coefficient

Let the following of documents :

– *d1 = {"Saturn is the gas planet with rings."}*

– *d2 = {"Jupiter is the largest gas planet."}*

– *d3 = {"Saturn is the Roman god of sowing."}*

First, we represent documents as vectors:

– Vector structure (Collection vocabulary):

V = {Saturn, is, the, gas, planet, with, rings, Jupiter, largest, Roman, god, of sowing}

Thus, vectors corresponding to documents are:

- $V_1 = (1,1,1,1,1,1,1,0,0,0,0,0)$
- $V_2 = (0,1,1,1,1,0,0,1,1,0,0,0)$
- $V_3 = (1,1,1,0,0,0,0,0,0,1,1,1)$

- $Sim(d_1,d_2) = \frac{M_{11}}{M_{01}+M_{10}+M_{11}} = \frac{4}{3+2+4} = (4/9) = 0.44$
- $Sim(d_1,d_3) = 0.27$
- $Sim(d_2,d_3) = 0.18$

b) Cosine Similarity: Often used for computing the similarity of documents.

If d_1 and d_2 are two document vectors, then:

$$Cosin(d_1, d_2) = \frac{\overrightarrow{d_1} \cdot \overrightarrow{d_2}}{\|\overrightarrow{d_1}\|\|\overrightarrow{d_2}\|} \quad (2.6)$$

Where:
" •": indicates vector dot product (Inner Product).

$$\vec{A}.\vec{B} = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \ldots\ldots + a_n b_n \quad (2.7)$$

|| d || : is the length of vector d.

$$\left\|\vec{A}\right\| = \sqrt{{a_1}^2 + {a_2}^2 + \ldots\ldots + {a_n}^2} \quad (2.8)$$

Example 2.18: Cosine similarity
- *d_1 = (3, 2, 0, 5, 0, 0, 0, 2, 0, 0)*
- *d_2 = (1, 0, 0, 0, 0, 0, 0, 1, 0, 2)*

Then:
- $\overrightarrow{d_1} \cdot \overrightarrow{d_2}$ *= 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5*
- $\left\|\overrightarrow{d_1}\right\|$ *= (3*3+2*2+0*0+5*5+0*0+0*0+0*0+2*2+0*0+0*0)1/2 = (42)1/2 = 6.481*
- $\left\|\overrightarrow{d_2}\right\|$ *= (1*1+0*0+0*0+0*0+0*0+0*0+0*0+0*0+1*1+0*0+2*2)1/2 = (6)1/2 = 2.245*
- *Cos(d_1, d_2) = 0.3150*

c) Cosine Similarity and TF-IDF : A commonly used combination for text clustering. Each document is represented by vectors of TF-IDF weights.

Example 2.19: Cosine similarity
We take the previous set of documents:

d_1: "Saturn is the gas planet with rings."
d_2: "Jupiter is the largest gas planet."
d_3: "Saturn is the Roman god of sowing."

Words vector structure (Vocabulary)
V = {Saturn, is, the, gas, planet, with, rings, Jupiter, largest, god, of, sowing}

Tf normalized vectors:
- *V_1 = (1/7, 1/7, 1/7, 1/7, 17, 17, 17,0,0,0,0,0)* *(Norm.Fact: $\sum_{i=1}^{m} t_i$= 7)*
- *V_2 = (0, 1/6, 1/6, 1/6, 1/6, 0, 0, 1/6, 1/6, 0, 0, 0)* *(Norm.Fact: 6)*
- *V_3 = (1/6,1/6,1/6,0,0,0,0,0,0,1/6,1/6,1/6)* *(Norm.Fact: 6)*

We recall:

$$w_{i,j} = tf_{i,j} * idf_i = tf_{i,j} * \log\left(\frac{N}{df_i}\right) \quad (2.9)$$

Tf-idf vectors:

*V1 = (1/7 * log(3/2) , 1/7 *log(3/3) , 1/7 *log(3/3) , … , 0, 0, 0, ...)*

Saturn is the Jupiter largest Roman.

- *V1 = (0.03, 0, 0, 0.03, 0.03, 0.07, 0.07, 0, 0, 0, 0, 0, 0)*
- *V2 = (0, 0, 0, 0.03, 0.03, 0, 0, 0.08, 0.08, 0, 0, 0, 0)*
- *V3 = (0.03, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.07, 0.07, 0.07, 0.07)*

Thus, similarities between documents are:
- *Sim(d1,d2) = 0.13*
- *Sim(d1,d3) = 0.05*
- *Sim(d2,d3) = 0.0*

2.10.5.3. Sentiment Analysis: is defined as a specific classification task, characterized by:

- Input data: a text
- Target: a class of sentiments.
  – e.g., positive, neutral, negative.
  – e.g., sad, happy, angry, surprised.
- Can be implemented as supervised classification task.
  – Requires training data.
  – i.e., pairs like <text, sentiment>
- Labeling data for sentiment analysis
  – Is expensive
  –Like every data labeling task
- Can require a preprocessing phasee like text categorization, including:
  – Text Cleanup (remove punctuation, HTML tags, …)
  – Normalize case
  – …
- However, reasonable features for sentiment analysis might include
  – Punctuation: use of “!”, “?”, “?!”
  – Smiles (usually encoded using inter-punctuation: ;-))
  – Use of visual markup, where available (red color, bold face, ...)
  – Amount of capitalization (“screaming”)

Example 2.20: Product review:
Let the following text: <*The image quality is good, but the zoom sucks*>

- Putting the pieces together:
  – POS tagging.
  – Keyphrase extraction.
  – Marking sentiment words.

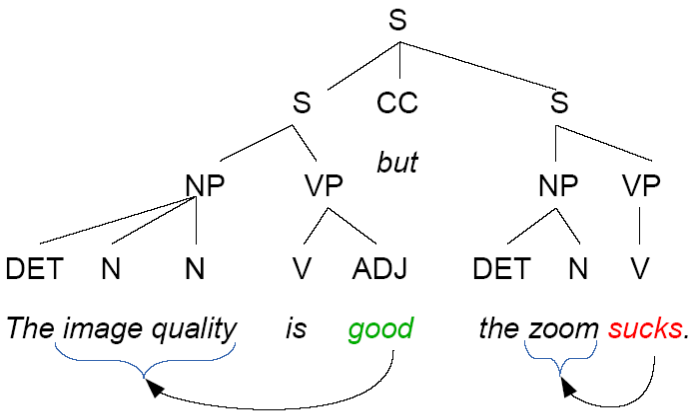


Figure 2.21. POS tagging for sentiment analysis

2.11. Summary

Text mining is contrasted relative to automated prediction. Model is constructed by training on samples of unstructured documents, and results are projected to new text. A standard data format for input to prediction methods is described. An important phase is the data preparation which consists in transforming text into a numerical format. That means sharing a common representation with numerical data mining.

Several text-mining problems are introduced that fit within the prediction framework. These include: document classification, information retrieval, clustering documents, information extraction, and performance evaluation.

CHAPTER 3

Automatic Text Categorization

Chapter content

3.1. Preface

Among the most common themes in analyzing complex data is the classification, or categorization of elements. Defined abstractly, the task is to classify a given data instance into a pre-specified set of categories. Applied to the domain of document management, the task is known as text categorization (TC) – given a set of categories (subjects, topics) and a collection of text documents, the process of finding the correct topic (or topics) for each document.

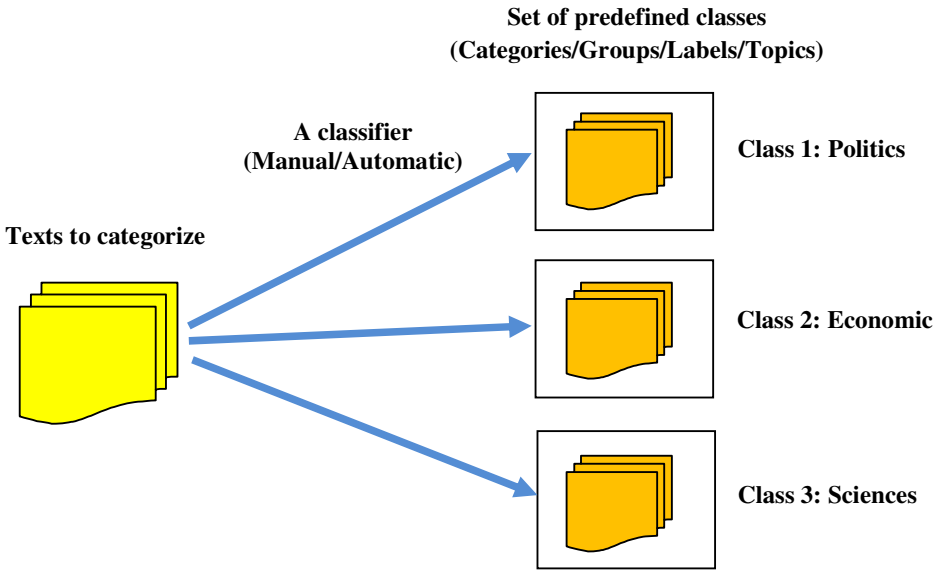


Figure 3.1. Principle of text categorization process

The field of automated text categorization dates back to the early 1960s [Maron, 1961]. Then, used for indexing scientific literature by means of controlled vocabulary. It was only in the 1990s that the field fully developed with the availability of ever increasing numbers of text documents in digital form especially on WWW network, and so, the necessity to organize them for easier use. Currently automated TC is applied in a variety of contexts – from the classical automatic or semiautomatic (interactive) indexing of texts to personalized commercials delivery, spam filtering, Web page categorization under hierarchical catalogues, detection of text genre, and many others fields. As with many other artificial intelligence (AI) tasks, there are two main approaches to text categorization. The first is the knowledge engineering approach in which the expert's knowledge about the categories is directly encoded into the system either declaratively or in the form of procedural classification rules. The other is the machine learning (ML) approach in which a general inductive process builds a classifier by learning from a set of pre-classified examples. Most of the recent work on categorization is concentrated on the ML approach, which requires only a set of manually classified training instances that are much less costly to produce [Feldman and Sanger, 2007].

In this chapter, we treat the following subjects: definition of the categorization task, description of several common applications, some difficulties of TC, monolingual TC vs multilingual TC, the formal problem of text representation, language identification step, and finally we survey the proposed approaches commonly used to solve the TC problem.

3.2. Definition of the problem

The general text categorization task can be formally defined as the task of approximating an unknown category assignment function *F* [Sebastiani, 2002, Jalam, 2003, Réhel, 2005, Nakache, 2007].

$$F : D \times C \rightarrow \{0, 1\}, \tag{3.1}$$

Where:
D: is the set of all possible documents.
C: is the set of predefined categories.
The value of *F(d, c)* is 1 if the document *d* belongs to the category c and 0 otherwise.
The approximating function

$$M : D \times C \rightarrow \{0, 1\} \tag{3.2}$$

is called a classifier, and the task is to build a classifier that produces results as "close" as possible to the true category assignment function *F*.

3.2.1. Single-Label versus Multilabel Categorization

Depending on the properties of *F*, we can distinguish between single-label and multilabel categorization. In multilabel categorization the categories overlap, and a document may belong to any number of categories. In single-label categorization, each document belongs to exactly one category. Binary categorization is a special case of single-label categorization in which the number of categories is two. The binary case is the most important because it is the simplest, most common, and most often used for the demonstration of categorization techniques. Also, the general single-label case is frequently a simple generalization of the binary case. The multilabel case can be solved by |*C*| binary classifiers (|C| is the number of categories), one for each category, provided the decisions to assign a document to different categories are independent from each other. [Sebastiani, 1999, Sebastiani, 2002]

3.2.2. Document-Pivoted versus Category-Pivoted Categorization

Usually, the classifiers are used in the following way: Given a document, the classifier finds all categories to which the document belongs. This is called a document-pivoted categorization. Alternatively, we might need to find all documents that should be filed under a given category. This is called a category-pivoted categorization. The difference is significant only in the case in which not all documents or not all categories are immediately available. For instance, in "online" categorization, the documents come in one-by-one, and thus only the document-pivoted categorization is possible. On the other hand, if the categories set is not fixed, and if the documents need to be reclassified with respect to the newly appearing categories, then category-pivoted categorization is appropriate. [Sebastiani, 2002]

3.3. Applications of text categorization

There are many common TC applications: text indexing, document sorting and text filtering, And Web page categorization. These are only a small set of possible applications, but they demonstrate the diversity of the domain and the variety of the TC subcases.

3.3.1. Indexing of texts using controlled vocabulary

The topic of most of the early research in the TC field is text indexing. In Boolean information retrieval systems (IRS), each document in a big collection is assigned one or more key terms describing its content. Then, the IR system is able to retrieve the documents according to the user queries, which are based on the key terms. The key terms all belong to a finite set called controlled vocabulary, which is often a thematic hierarchical thesaurus such as the NASA aerospace thesaurus or the MESH thesaurus for medicine. The task of assigning keywords from a controlled vocabulary to text documents is called text indexing. If the keywords are viewed as categories, then text indexing is an instance of the general TC problem and can be addressed by the automatic techniques described in this chapter. Typically, each document should receive at least one, and not more than k, keywords. Also, the task can be solved either fully automatically or semiautomatically. [Maron, 1961]

3.3.2. Document sorting and text filtering

Another common problem related but distinct from document indexing is sorting the given collection of documents into several (categories). For instance, in a newspaper, the classified ads may need to be categorized into "Personal," "House Sale," "news," and so on. Another example is e-mail coming into an organization, which may need to be sorted into categories such as "Complaints," "Deals," "Job applications," and others. The document sorting problem has several features that distinguish it from the related tasks. The main difference is the requirement that each document belong to exactly one category. Other typical features are relatively small numbers of categories and the "online" nature of the task: The documents to be categorized are usually presented to the classifier one by one, not as a single batch. Text filtering activity can be seen as document sorting with only two categories – the "relevant" and "irrelevant" documents. Examples of text filtering abound, a sports related online magazine should filter out all non sport stories it receives from the news feed. An e-mail client should filter away spam. A personalized ad filtering system should block any ads that are uninteresting to the particular user. [Schütze et al. 1995, Cohen, 1996, Sahami *et al.*, 1998, Zarogoza, 1999, Iyer et al., 2000]

3.3.3. Hierarchical Web Page Categorization

A common use of TC is the automatic classification of Web pages under the hierarchical catalogues posted by popular Internet portals such as Yahoo. Such catalogues are very useful for direct browsing and for restricting the query-based search to pages belonging to a particular topic. The other applications described in this section usually constrain the number of categories to which a document may belong. Hierarchical Web page categorization, however, constrains the number of documents belonging to a particular category to prevent the categories from becoming excessively large. Whenever the number of documents in a category exceeds *k*, it should be split into two or more subcategories. Thus, the categorization system must support adding new categories and deleting obsolete ones. Another feature of the problem is the hypertextual nature of the documents. The Web documents contain links, which may be important sources of information for the classifier because linked documents often share semantics. The hierarchical structure of the set of categories is also uncommon. It can be dealt with by using a separate classifier at every branching point of the hierarchy.

3.4. Particular difficulties of text categorization

The use of machine learning methods to treat textual data is more difficult than the processing of numeric data, many difficulties can be met, such as:

3.4.1. Big size of vectors

The vector model proposed by [Salton and McGill, 1983] consists in the representation of every text by a vector whose components are the terms constituting the vocabulary of the corpus. However, for a corpus with reasonable size, the table "*terms x texts*" can have hundreds of thousands of lines (texts) and thousands of columns (terms); but this table is often hollow, e.g., the majority of its cells are empty. Thus, It affects the process of training while returning some inoperative algorithms and while increasing the risk of overfitting [Jalam, 2003].

1. ***Algorithms complexity:*** In Text categorization, the big dimensionality can decrease the efficiency of learning algorithms. Because, most sophisticated learning algorithms such as: decision trees, k-NN, LLSF (Linear Least Square Fit) [Yang and Chute, 1992] are sensitive to the corpus size |*Tr*| which is the number of variables used to represent texts. So, |*Tr*| is an important parameter in the complexity of the used algorithm. For this purpose, several methods are used to reduce the dimension of term vocabulary before applying any algorithm.
2. ***Overfitting:*** occurs when a classifier classifies the examples of the training set correctly, but classifies new examples badly. To avoid this kind of overfitting, we must limit the number of used terms (describers) according to the number of examples in the training sample. [Fuhr and Buckely, 1991] propose the use of at least 50 to 100 times more texts than of terms to avoid this phenomenon. It is also necessary to signal that the reduction of dimension must be used with precaution to avoid the removal of pertinent terms [Sebastiani 2002].

3.4.2. Imbalance of classes

In practice, class sizes are often unbalanced, and for some classes, the number of positive examples is weak compared to that of negative examples. This creates an additional difficulty because the small classes are badly represented. A proposed solution to overcome this problem is to use recovery techniques.

3.4.3. Term Ambiguity

A same word or a same expression can have many different senses. The inherent ambiguity of words or phrases appears in two levels: lexical and syntactic levels. It is necessary to add the ambiguities related to the context. For the lexical level, there is the problem of homonymy. Homonyms are words that have the same form, the same spelling, but different senses. In the following sentences, the word bank have two different meanings (*Mary walked along the bank of the river, Harbor Bank is the richest bank in the city*). [Lefèvre, 2000, Jackson and Moulinier, 2002]

3.4.4. Problem of synonymy

Two words or two expressions are considered synoymes if they have the same meaning. So, that it is possible to substitute one by the other without changing the meaning. For example: "*I bought a pair of shoes of* ***big size****, I bought a pair of shoes of* ***large size".*** [Joachims, 2001]

3.4.5. Decision Subjectivity

In some situations, the adherence of an element to a class or a category is completely objective, for example a customer buys or not such a product. Instead, the assignment of a category to a text is subjective [Uren, 2000], because the category is assigned according to the semantic content of this text which is a subjective notion, and depends of an expert's judgment.

3.5. How to categorize a monolingual text: the general process of TC

The categorization of a text consists in building a prediction model which receives as input a text whose category is unknown and associates to it one or more categories as output. The categorization process is carried out in many phases and steps.

Phase 1: Learning

This phase itself is comprised of several steps which are:

1. Have a set of labeled texts (whose categories are known) called learning set or learning collection.
2. Perform a preprocessing on these texts
   - Punctuation removal.
   - Stopwords removal.
   - Standardization of some characters for certain languages (Arabic for example)
   - Segmenting each text into lexical units (words, phrases, n-grams, etc.). This operation is also called tokenization
3. Representing each text with a vector of terms or a term can be: a word, a phrase, a root, a lemma, a concept, an n-gram, others.
4. Coding each text vector in a digital form by associating to each term a weight using one of the well-known coding models (binary model, model 3 values, the term frequency TF, TF-IDF Frequency)
5. If the vocabulary obtained from the learning collection or the set of texts to categorize is large, we can proceed to a phase of feature selection which permits to select the most relevant terms using specific methods (MI, IG, χ^2). This allows to optimize the categorization process.
6. Applying one of the well-known learning algorithms on the subset of words selected in (5) to build a model that permits to predict the text category (algorithms like: kNN, Naive Bayes, SVM, Decision trees, Neural Networks, etc.).

Phase 2: Categorization of a new text

Includes the following steps:

1. Repeat steps 2, 3, 4, 5 seen previously.
2. Apply the prediction model built in (6) to predict the new text category.

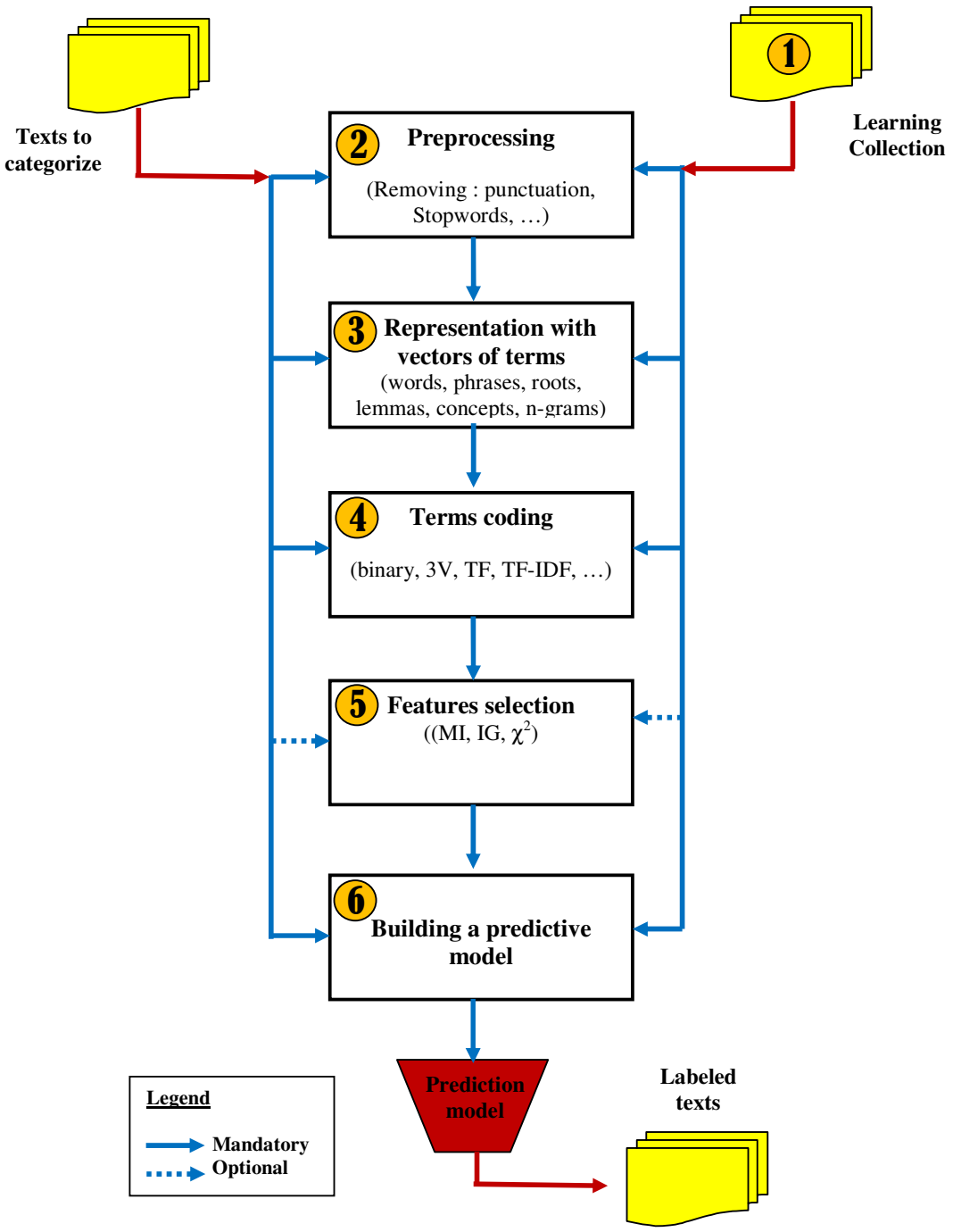


Figure 3.2. The general process of automatic text categorization

3.6. The problem of document representation

3.6.1. Choice of document features (terms)

In the field of text categorization, training algorithms are not able to treat texts directly. This is why a very important preliminary phase known as representation phase is necessary [Sebastiani, 2002]. This phase generally consists in representing each document to categorize by a vector, whose components are: words, sentences, or other lexemes in order to make it exploitable by the training algorithms.

3.6.1.1. Representation with bag of words: It is the simplest representation of texts which was introduced with the appearance of the vector model [Salton, 1971, Salton et al., 1975, Salton and Mc-Gill, Salton, 1980, Salton, 1983, Salton and Buckley, 1988]. All texts are transformed into vectors in which each component represents a term. Thus, terms are the words which constitute the text. With this representation, the size of the vector representing the document is equal to the size of the vocabulary which is in general very large and often consists of several tens of thousands of words. However, the great dimension of these data lets the majority of classification algorithms difficult to apply, without forgetting that the representation of textual data is typically hollow. [Young-Min, 2008]

3.6.1.2. Representation with Sentences: Some researchers propose to use the sentences as units to represent texts instead of words as it was seen with the "bag of words" representation, because sentences are more informative than words only, for example, sentences like: "Search for Information", "World Wide Web" have a smaller degree of ambiguity than the separated words, in addition, the sentences have the advantage of preserving the information relative to the position of words in the text [Fuhr and Buckley, 1991, Schutze et al., 1995, Caropreso et al., 2000, Furnkranz et al., 1998]

3.6.1.3. Representation with lexical roots (Stems): In the representation "Bag of Words" each inflection (derived word) is regarded as a different term; in particular the various forms of a verb are considered as different words (cross, crosses, crossed) although they are inflected forms of the same verb. So, that can increase the dimension of the vector space representing different texts. To deal with this problem, it is necessary to consider only the roots of words (stems) than the whole words. Several algorithms were proposed to substitute the words by their roots; the most known for English is the algorithm of Porter algorithm. [Porter, 1980, Schmid, 1994, Hull, 1996, de Loupy, 2001]

3.6.1.4. Representation with lemmas: The lemmatization consists in using the grammatical analysis in order to replace verbs by their infinitive forms and nouns by their forms in singular. Lemmatization is thus more complicated to implement than the search of roots because it requires a grammatical analysis of texts. An effective algorithm named "TreeTagger" was developed for many languages: English, German, and Italian. This algorithm uses the decision trees to carry out the grammatical analysis with files of parameters specific to each language. [Schmid, 1994] [Mathieu, 2000]

3.6.1.5. Representation Based on N-grams: An n-gram of X is defined as a sequence of N consecutive X. X can be a character or a word [Nicolas, 2008]. An n-gram of character is thus a consecutive sequence of N characters [Shannon, 1948, Cavnar and Trenkle, 1994] which cannot be ordered. (ex: the 3-grams of the sentence "Hello Sir" are: "Hel, ell, llo, lo_, o_S, _Si, Sir [Miller et al., 1999, Biskri and Delisle, 2001, Jalam and Teytaud, 2002]. The n-grams profile of a document consists of the list of the most frequent in the reverse order of their frequencies. The approach of text segmentation into characteristic n-grams has several advantages, in particular: [Jalam and Chauchat, 2002]

- Avoid the use of lemmatization and stemming on the text which requires an algorithmic and linguistic effort.
- Tolerant to spelling, typing, and OCR mistakes.
- This approach operates independently from languages. [Dunning, 1994]
- Segmentation into words is difficult for some languages e.g in Arabic the names and additional subjects are in some cases attached to verbs and the string is thus a sentence like: (katabtouhou) (I wrote it) [Jalam, 2003]

3.6.1.6. Conceptual representation: Is based on the vector formalism, but the elements of the vector are not directly associated any more with index terms but rather with concepts. The idea is to gather the synonymous words and to associate an under-adjacent lexical concept to them which requires the construction of a lexical base for each language (e.g., Wordnet ontology) [Feldbaum, 1998, Jaillet, 2004,]. For example we associate with the synonyms (*summit, top, peak*) the concept "*peak*". The advantage of the conceptual representation is to reduce the representation vector space by gathering the synonymous words and attributing to them a common concept. Contrary to the representation "Bag of Words" which associates to each word a dimension in the vector. However, the major disadvantage of the conceptual representation is the absence of lexical bases for all languages which allow such representations [Réhel, 2005]

3.6.2. Terms coding

Once the list of the determined terms (attributes/Features) is built, the document is represented as a vector of these terms, it remains to give a weighting to every component of the vector. There are several models to compute the term weighting: [Salton, 1975, Salton, 1980, Salton and Mc-Gill, 1983]

***3.6.2.1. Binary model*:** is the most basic model, in which we simply check for the presence or the absence of word in the document. Thus, we obtain a matrix of ones and zeros.

Example 3.1: Binary model
let the following collection constituted of three documents:

– *d1 = {Chinese Beijing Chinese}*
– *d2 = {Chinese Chinese Shanghai}*
– *d3 = {Chinese Chinese Chinese Tokyo Japan}*

Collection vocabulary:

V = {Chinese, Beijing, Shanghai, Tokyo, Japan}

Thus, vectors corresponding to documents d_1, d_2, d_3 according to the binary model:

- $V_1 = (1, 1, 0, 0, 0)$
- $V_2 = (1, 0, 1, 0, 0)$
- $V_3 = (1, 0,0, 1, 1)$

3.6.2.2. Three values model: Another model to map the presence of words in a document is the three valued model. In this model, 0 expresses that the word didn't occur, 1 the word occurred once, 2 the word occurred 2 or more times.

Table 3.1 Thresholding frequencies to three values model**.**

Value	Description
0	Word didn't occur
1	Word occurred once
2	Word occurred 2 or more times

Example 3.2: The three value model
We take the same collection seen in the previous example:
The corresponding vectors are:

- $V_1 = (2, 1, 0, 0, 0)$
- $V_2 = (2, 0, 1, 0, 0)$
- $V_3 = (2, 0, 0, 1, 1)$

Another variant of this model involves zeroing all values below a certain threshold. That means tokens should have a minimum frequency before being considered of any use. This can reduce the complexity of the spreadsheet significantly and might be a necessity for some predictive methods.

3.6.2.3. Term frequency code (tf): the weight of a term t_i in a document d_j is not 0 or 1 as in the binary model. The weight here is the number of times t_i appears in d_j denoted by f_{ij}. If a word occurs ten (10) times in a document, this count would be used. For some learning methods, the count does give a slightly better result. A normalization may also be applied.

$$tf_{ij} = f_{ij} = Number\ of\ times\ t_i\ appears\ in\ d_j \tag{3.3}$$

other normalized forms are possibles:

$$tf_{ij} = \frac{f_{ij}}{Max(f_{1j}, f_{2j}, \ldots, f_{|V|j})} \tag{3.4}$$

Or

$$tf_{ij} = \frac{f_{ij}}{\sum_{i=1}^{|V|} f_{ij}} \tag{3.5}$$

Where:
f_{ij}: The frequency of the term t_i in the document d_j.
$|V|$: the size of the document collection

Example 3.3: Term frequency code
For the same example seen previously, we have:
The corresponding vectors without normalization using (3.5) are:

- $V_1 = (2, 1, 0, 0, 0)$
- $V_2 = (2, 0, 1, 0, 0)$
- $V_3 = (3, 0,0, 1, 1)$

The corresponding vectors with normalization are:

- $V_1 = (2/3, 1/3, 0, 0, 0) = (0.67, 0.33, 0, 0, 0)$
- $V_2 = (2/3, 0, 1/3, 0, 0) = (0.67, 0, 0.33, 0, 0)$
- $V_3 = (3/5, 0, 0, 1/5, 1/5) = (0.6, 0, 0, 0.2, 0.2)$

3.6.2.4. Tf-idf coding (Term Frequency-Inverse Document Frequency)
a) Document Frequency (df_t): as term frequency souffers from some critical problem: all document terms are equally important when they are used to assess relevance in text categorization or even in information retrieval. In fact, certain terms have little or no discriminating power in determining relevance [Manning et al, 2008]. For example a collection of documents on education is likely to have

the term "student" or the term "teacher" in almost every document. To this end, we introduce a method for attenuating the effect of terms that occur to often in the collection to be more significative for relevance determination. One way is to decrease the term weights of these terms by a factor that grows with its collection frequency. For instance, the total number of terms weights in the collection *cf* or the number of documents containing the term *t* denoted df_t.

So, document frequency df_t is the number of documents in the collection that contain the term t. we almost prefer the use of this factor instead of *cf* because we want a few documents which can highly discriminate between different documents [Manning et al, 2008].

b) Inverse Document Frequency (idf_t): after calculating the document frequency df_t, we can define the inverse document frequency of the term *t* denoted idf_t as follows:

$$idf_t = Log\ (\frac{N}{df_t}) \tag{3.6}$$

Where:

N: is the total number of documents in the collection (the collection size).

df_t: the document frequency of the term *t*.

Log: the decimal logarithm (*Log(x) = ln(x)/ln(10)*)

We note that idf_t of a rare term is high, whereas the idf_t of a frequent term is likely to be low.

c) tf-idf coding (Term Frequency-Inverse Document Frequency): It's a more complex weighting scheme that take into account the frequencies of the term in the document, in the category, and in the whole collection. The most common *tf-idf* scheme gives the term t_i in the document d_j the weight: [Salton and Buckley, 1987, Salton and Buckley, 1988, Joachims, 1997, Joachims, 1998]

$$tf - idf_{ij} = tf_{ij} * idf_i \tag{3.7}$$

$$tf - idf_{ij} = tf_{ij} * Log(\frac{N}{df_i}) \tag{3.8}$$

Where:

tf_{ij}: the frequency of the term t_i in the document d_j.

N: the total number of documents in the collection (training set).

df_i: the number of documents in which the term t_i occurs at least once.

We note that:

1. tf-idf is highest when a term t occurs many times within a small number of documents.
2. Lower when the term occurs fewer times in a document, or occurs in many documents.
3. Lowest when the term occurs in virtually all documents.

Example 3.4: *tf-idf* weighting

For the same example, we have:

The collection of documents:

– *d1 = {Chinese Beijing Chinese}*

– *d2 = {Chinese Chinese Shanghai}*

– *d3 = {Chinese Chinese Chinese Tokyo Japan}*

N = 3 documents

The corresponding *tf-idf* vectors without normalization are:

- V_1 = (2*log(3/3), 1*log(3/1), 0, 0, 0)= (0, 0.48, 0, 0, 0)
- V_2 = (2*log(3/3), 0, 1*log(3/1), 0, 0)= (0, 0, 0.48, 0, 0)
- V_3 = (3*log(3/3), 0, 0, 1*log(3/1), 1*log(3/1))= (0, 0, 0, 0.48, 0.48)

d) Variants of tf and tf-idf weighting: a number of alternatives to *tf* and *tf-idf* have been considered.

Table 3.2. *tf* and *tf-idf* variants

Term frequency tf	Document frequency df	Normalization
Natural value $tf_{t,\ d}$: number of times term *t* occurs in document *d*	$idf_t = Log\ (\frac{N}{df_t})$	None
Boolean value (binary model): $tf_{t,\ d} = \begin{cases} 1 & if\ f_{t,\ d} > 0 \\ 0 & otherwise \end{cases}$	1	1
Normalized with max term frequencies $tf_{t,\ d} = \frac{tf_{t,\ d}}{Max(tf_{t1,\ d}, tf_{t2,\ d}, \dots, tf_{\lvert V\rvert,\ d})}$	$idf_t = Log\ (\frac{N}{df_t})$	Cosine normalization $tf_{t,\ d} = \frac{tf_{t,\ d}}{\sqrt{(tf_{t1,\ d})^2 + (tf_{t2,\ d})^2 + \dots + (tf_{\lvert V\rvert,\ d})^2}}$
Normalized with sum of term frequencies $tf_{t,\ d} = \frac{tf_{t,\ d}}{\sum_{i=1}^{\lvert V\rvert} tf_{t_i,\ d}}$	$idf_t = Log\ (\frac{N}{df_t})$	
Normalized with logarithm value $tf_{t,\ d} = (1 + \text{Log}\ (tf_{t,\ d}))$	$idf_t = Max[0, Log\ (\frac{N - df_t}{df_t})]$	
Augmented value $tf_{t,\ d} = 0.5 + \frac{0.5 * tf_{t,\ d}}{Max(tf_{t,\ d})}$	$idf_t = Max[0, Log\ (\frac{N - df_t}{df_t})]$	

3.6.2.5. TFC coding: The TFC coding is similar to the *tf-idf* coding, but it also has the advantage of correcting the lengths of texts by the cosines normalization, and this to avoid to promote the longest documents.

$$TFC(t_i, d_j) = \frac{tf-idf(t_i, d_j)}{\sqrt{\sum_{k=1}^{\lvert V\rvert}(tf-idf(t_k, d_j))^2}} \quad (3.9)$$

Other coding are used, such as: the LTC coding [Buckley et al., 1994] which tempts to reduce the effects of the frequency differences, or the coding based on the entropy. [Dumais, 1991, Aas and Eikvil, 1999, Jalam 2003].

3.6.2.6. LNU Coding: The different texts that compose a corpus have different sizes. So, it is necessary to take this into account when coding the terms. According to Singhal [Singhal, 1996a, Ng *et al.*, 2000, Mathieu, 2000], there are two factors to be considered when we work on long texts:

- The present words have the tendency to have higher frequencies.
- The long texts are more susceptible to contain different key-words.

Thus, to take into account these two factors, they propose the LNU coding, defined as follows:

$$LNU = L * U \tag{3.10}$$

$$L = \frac{1+Log\,(TF(m,t))}{1+Log(\overline{TF}(m))} \tag{3.11}$$

$$U = \frac{1}{0.8+0.2*\frac{U(t)}{\overline{U}}} \tag{3.12}$$

Where:
$TF(m,t)$: the number of words in the text t.
$\overline{TF}(m)$: The average frequency in the text t.
U(t): the number of unique terms in the text t.
$\overline{U}$: the average number of words in the corpus.

3.6.2.7. The entropy: One more significant weighting approach is based on the use of the entropy. This last one measures the dispersion of a term in a corpus and can be important information as part of the term. The entropy *E* for a term t_i is described by the following formula: [Nicolas, 2009].

$$E(t_i) = \sum_j \frac{p_{ij}Log_2(p_{ij})}{Log_2(N)} \tag{3.13}$$

$$p_{ij} = \frac{tf_{ij}}{GF_{ij}} \tag{3.14}$$

Where:
tf_{ij}*:* the number of times the term t_i occurs in the document d_i
GF_i is the total number of times for which the term t_i occurs in the corpus of *N* documents.

A possible representation using this approach can be defined as follows:

$$w_{ij} = \big(1 + E(t_i)\big)Log(tf_{ij} + 1) \tag{3.15}$$

Where: $E(t_i)$: the entropy of the term t_i calculated according to (3.13)

3.7. Feature reduction

A major problem in text categorization is the large size of document representation vectors (size of terms vocabulary). For example, the dimension of the bag-of-words feature space for a big collection can reach hundreds of thousands; moreover, document representation vectors, although sparse, may still have hundreds and thousands of nonzero components. This will pose some problems during the phase of training, especially for some algorithms that are very sensitive to the size. Thus, it is very beneficial to reduce the size of the vocabulary by selection or extraction of an optimal subset of terms without affecting the quality of categorization.
The main objectives of such a reduction of terms are:

- Facilitate the visualization and the understanding of the categorization problem.
- Define the most applicable terms for the categorization.
- Reduce the space of representation of documents, as well as the necessary storage space.
- Reduce the training time, and therefore to increase the performances of the used algorithms.
- Avoid the phenomena of over-fitting.

3.7.1. Local reduction

Here we define for every category C_i a new set of terms T_i such as: $|T_i| \ll |T_r|$. Then each category C_i possesses its own subset of terms, and therefore, every document d_j will be represented by several vectors V_j different according to the category. [Apté et al., 1994, Lewis and Ringuette, 1994, Schutz et al., 1995, Wiener et al., 1995, Ng et al., 1997, Li and Jain, 1998]. Generally, we put: $10 \leq |T_i| \leq 50$

3.7.2. Global reduction

In global reduction, the new set of terms $\grave{T}$ is chosen according to all the categories. Thus, every document d_j will be represented by only one vector V_j whatever the category. [Yang and Pedersen, 1997, Mladenic and Grobelnik, 1998, Caropreso et al., 2001, Yang and Liu, 1999].

We also note, that all the techniques of term reduction of terms can be applied either locally either globally.

3.8. Dimensionality reduction by feature selection

All words that are irrelevant to the categorization task can be dropped with no harm to the classifier performance and may even result in improvement owing to noise reduction. The preprocessing step that removes the irrelevant words is called feature selection. Most TC systems at least remove the stop words – the function words and in general the common words of the language that usually do not contribute to the semantics of documents and have no real added value. Many systems, however, perform a much more aggressive filtering, removing 90 to 99 percent of all features.

Various FS methods, such as: document frequency (DF), information gain (IG) [Yang and Pedersen, 1997, Sebastiani, 2002], mutual information (MI) [Lewis, 1992a, Moulinier, 1997, Dumais et al., 1998], Chi-2 test (χ^2) [Schutze et al., 1995, Wiener et al., 1995], Bi-Normal Separation (BNS), and weighted log-likelihood ratio (WLLR), have been proposed for the task [Moulinier, 1996, Wiener et al., 1995, Yang and Pedersen, 1997; Sahami, 1999, Nigam et al., 2000; Forman, 2003] and make text classification more efficient and accurate. We note also, that some comparative studies have been established between many of these methods [Yang and Pedersen, 1997, Liu, 2004, Li et al, 2009].

3.8.1. Document Frequency DF

Document frequency is the number of documents in which a term t occurs in a dataset. It is the simplest criterion for term selection and easily scales to a large dataset with linear computation complexity. It is a simple but effective feature selection method for text categorization [Yang and Pedersen, 1997, Liu, 2004, Li et al, 2009].

$$DF(t) = \sum_{i=1}^{m}(A_i) \tag{3.16}$$

Where: m: the total number of documents in the collection.

A_i: an integer that takes 1 if the term t occurs in the document and 0 if not.

Experimental evidence suggests that using only the top 10 percent of the most frequent words does not reduce the performance of classifiers. This seems to contradict the well-known “law” of IR, according to which the terms with low-to-medium document frequency are the most informative. There is no contradiction, however, because the large majority of all words have a very low document frequency, and the top 10 percent do contain all low-to-medium frequency words.

3.8.2. Mutual Information (MI)

This measure is based on the number of time that a word appears in a category. More a word appears in a category; more the mutual information of the word will be high. And on the contrary, more a word appears outside of the category (and more a category appears without the word), less the mutual information will be high [Lewis, 1992, Mouliner, 1997, Dumais *et al.*, 1998]. And then, it is necessary to make an average of the word scores paired to each of the categories. The weakness of this measure is that it is influenced by the frequency of the words. For example, a rare term will be more favored, because it risks less to appear outside of the category.

$$MI(t_k, C_i) = log \frac{P(t_k, C_i)}{P(t_k).P(C_i)} \quad (3.17)$$

And it is estimated as:

$$MI = log \frac{A_i * N_{all}}{(A_i + C_i)(A_i + B_i)} \quad (3.18)$$

Where:

P(t_k) : the probability that a document *d* contains the term t_k.

P(c_i): the probability that a document *d* belongs to the category c_i.

P(t_k, c_i): the probability that a document *d* contains the term t_k and also belongs to the category c_i

A_i: the number of the documents that contain the term t_k and also belong to the category c_i ;

B_i: the number of the documents that contain the term t_k but do not belong to the category c_i;

C_i: the number of the documents that do not contain the term t_k but belong to the category c_i , (i.e., $Ci = N_i - A_i$)

D_i : the number of the documents that neither contain the term t nor belong to the category c_i , (i.e., $D_i = N_{all} - N_i - B_i$)

N_i : the total number of the documents that belong to the category c_i ;

N_{all} : the total number of all documents from the training data.

3.8.3. Information Gain (IG)

Information gain [Yang & Pedersen, 1997] of a term measures the number of bits of information obtained for category prediction by the presence or absence of the term in a document. Let m be the number of classes. The information gain of a term t is defined as: [Sebastiani, 2002]

$$IG(t_k, c_i) = \sum_{c_i \in \{C, \overline{C}\}} \sum_{t \in \{t_k, \overline{t_k}\}} p(t_k, c_i) \,.\, log \frac{P(t_k, c_i)}{P(t_k).P(c_i)} \quad (3.19)$$

$$IG(t, c_i) = -\sum_{c \in \{C \cup \overline{C}\}} \sum_{t \in \{t, \overline{t}\}} p(t, c) \,.\, \log \frac{\mathrm{P(c|t)}}{\mathrm{P(c)}} \quad (3.20)$$

Where:

$P(t_k)$, P(c_i): seen previously

$P(t_k, c_i)$: the probability that a document *d* contains the term t_k and also belongs to the category c_i

P($c_i|t_k$): the probability that a document d belongs to a category c_i, and contains the term t_k.

3.8.4. χ^2 Statistic (Chi-square / Chi-2)

the χ^2 statistic measures the maximal strength of dependence between a term (describer) t_k (present or absent) and a category c_i (present or absent) [Schütze *et al.*, 1995, Wiener *et al.*, 1995]. For this fact, it is calculated on a table *2x2* called table of contingency. this last one is built for each term t_k of the corpus, as well as for each category c_i and contains several calculated values such as: the number of the documents belonging to the category c_i and in which the term t_k is present (value a), the number of documents in which appears t_k but that don't belong to the category c_i (value b), etc. (see table 22).

Table 3.3. Table of contingency

	Term t_k present	**Term t_k absent**	
Category c_i present	a	c	$a + c$
Category c_i absent	b	d	$b + d$
	$a + b$	$c + d$	$N_{all} = a + b + c + d$

Experiments show that this measure (and several other measures) can reduce the dimensionality by a factor of 100 without loss of categorization quality –or even with a small improvement (Yang and Pedersen 1997). It's defined as fellows: [Sebastiani, 2002]

$$\chi^2_{max}(t_k, c_i) = max_{c \in C} \frac{|T_r|.(P(t_k,c_i).p(\overline{t_k},\bar{c}_i) - P(t_k,\ \bar{c}_i).P(\bar{t}_k,c_i))^2}{P(t_k).P(\bar{t}_k).P(c_i).P(\bar{c}_i)} \quad (3.21)$$

$$\chi^2(t_k, c_i) = \frac{N_{all}.(ad-cb)]^2}{(a+c).(b+d).(a+b)(c+d)]} \quad (3.22)$$

$$\chi^2(t_k, c_i) = \frac{N_{all}.[(a\ (N_{all}-N_i-b)-b(N_i-a)]^2}{N_i.(N_{all}-N_i).(a+b).[N_{all}-(a+b)]} \quad (3.23)$$

Where:

$|T_r|$: The collection size

P(t_k) : the probability that a document *d* contains the term t_k ;

P($\bar{t}_k$): the probability that a document *d* does not contain the term t_k;

P(c_i): the probability that a document *d* belongs to the category c_i ;

P($\bar{c}_i$): the probability that a document *d* does not belong to the category c_i ;

P(t_k, c_i): the probability that a document *d* contains the term t_k and also belongs to the category c_i

P($\overline{t_k}, \bar{c}_i$): The probability that a document *d* does not contain the term t_k and does not belong to the category c_i

P($t_k, \bar{c}_i$): the probability that a document *d* contains the term t_k , but does not belong to the category c_i

P($\bar{t}_k, c_i$): The probability that a document *d* does not contain the term t_k , but belongs to the category c_i

To calculate these probabilities statistical information from the training data is needed, and notations about the training data are given as follows:

$\{C_i\}_{i=1}^m$: the set of categories;

a: the number of the documents that contain the term t_k and also belong to the category c_i ;

b: the number of the documents that contain the term t_k but do not belong to the category c_i;

c: the number of the documents that do not contain the term t_k but belong to the category c_i , (i.e., $c = N_i - a$)

d : the number of the documents that neither contain the term t nor belong to the category c_i , (i.e., $d = N_{all} - N_i - b$)

N_i : the total number of the documents that belong to the category c_i ;

N_{all} : the total number of all documents from the training data.

3.8.5. Weighted Log Likelihood Ratio (WLLR)

WLLR method [Nigam et al., 2000] is defined as:

$$WLLR(t_k, c_i) = p(t_k \mid c_i) . \log \frac{P(t_k|c_i)}{P(t_k|\bar{c}_i)} \quad (3.24)$$

And it is estimated as:

$$WLLR(t_k, c_i) = \frac{a}{N_i} . \log \frac{a.(N_{all} - N_i)}{b.N_i)} \quad (3.25)$$

3.9. Dimensionality Reduction by Feature Extraction

Another way of reducing the number of dimensions is to create a new, much smaller set of synthetic features from the original feature set in order to maximize the categorization efficiency [Yang and Pedersen, Forman, 2003, Emmanuel, 2007]. The rationale for using synthetic features rather than naturally occurring words (as the simpler feature filtering method does) is that, owing to polysemy, homonymy, and synonymy, the words may not be the optimal features. By transforming the set of features it may be possible to create document representations that do not suffer from the problems inherent in those properties of natural language. One used technique is the term clustering which addresses the problem of synonymy by grouping together words with a high degree of semantic relatedness. These word groups are then used as features instead of individual words. Experiments conducted by several groups of researchers showed a potential in this technique only when the background information about categories was used for clustering [Baker and McCallum 1998; Slonim and Tishby 2001]. With unsupervised clustering, the results are inferior [Lewis 1992a, 1992b; Li and Jain 1998]. Another systematic approach is the latent semantic indexing approach (LSI) proposed by [Deerwester et al., 1990]. It consists in the decomposition of the matrix representing each document into singular values by the column of the occurrences of terms that compose it. The performance of the LSI also improves if the categories information is used. Several LSI representations, one for each category, outperform a single global LSI representation [Pham et al., 2008]. The experiments also show that LSI usually performs better than the chi-square filtering scheme.

3.10. Knowledge engineering approach to TC

The knowledge engineering approach to TC is focused around manual development of classification rules. A domain expert defines a set of sufficient conditions for a document to be labeled with a given category. The development of the classification rules can be quite labor intensive and tedious.
We mention only a single example of the knowledge engineering approach to the TC – the well-known CONSTRUE system [Hayes et al., 1988; Hayes et al. 1990; Hayes and Weinstein, 1990; Hayes 1992] built by the Carnegie group for Reuters. A typical rule in the CONSTRUE system is as follows:

If DNF (disjunction of conjunctive clauses) formula **then** category **else** ¬category

Such rule may look like the following:

If ((wheat & farm) or
(wheat & commodity) or
(bushels & export) or
(wheat & tonnes) or
(wheat & winter & ¬soft))
then Wheat
else ¬Wheat

The system was reported to produce a 90-percent breakeven between precision and recall on a small subset of the Reuters collection (723 documents). It is unclear whether the particular chosen test collection influenced the results and whether the system would scale up, but such excellent performance has not yet been unattained by machine learning systems. However, the knowledge acquisition bottleneck that plagues such expert systems (it took several man-years to develop and fine-tune the CONSTRUE system for Reuters) makes the ML approach attractive despite possibly somewhat lower quality results.

3.11. Machine learning approach to TC

In the ML approach, the classifier is built automatically by learning the properties of categories from a set of preclassified training documents. In the ML terminology, the learning process is an instance of supervised learning because the process is guided by applying the known true category assignment function on the training set. There are many approaches to classifier learning; some of them are variants of more general ML algorithms, and others have been created specifically for categorization. Four main issues need to be considered when using machine learning techniques to develop an application based on text categorization. First, we need to decide on the categories that will be used to classify the instances. Second, we need to provide a training set for each of the categories. As a rule of thumb, about 30 examples are needed for each category. Third, we need to decide on the features that represent each of the instances. Usually, it is better to generate as many features as possible because most of the algorithms will be able to focus just on the relevant features. Finally, we need to decide on the algorithm to be used for the categorization.
Several learning algorithms have been used in the field of TC, including: Bayesian networks [Lewis, 1998, Lewis and Ringuette, 1994, McCallum et Nigam, 1998a, Sahami, 1998, Androutsopoulos et al., 2000, Chai et al., 2002], kNN method [Yang and Chute, 1994, Yang and Li, 1999], Rocchio method [Rochio, 1971], Decision trees [Quilan, 1986, Quilan, 1987, Quilan, 1990, Quilan, 1993, Fuhr et al, 1991, Quilan, 1996, Lewis and Ringuette, 1994, Apté *et al.*, 1998, Li et Jain ,1998, Cohen et Singer ,1999, Zighed and Rakotomalala, 2000, Govindarajan, 2007], regression Methods [Hull, 1994, Zhang et al., 2003], Support Vector machines [Vapnik, 1995, Joachims, 1998, Joachims, 1999, Joachims, 2000, Joachims, 2001, Dumais *et al.*, 1998, He et al, 2000, Platt, 1998, Pilaszy, 2005], Decision rules [Cohen, 1995a, Cohen, 1995b, Cohen and Hirch, 1998, Li and Yamanishi, 2002], Neural networks [Wiener, 1993, Wiener *et al.*, 1995, Schütze *et al.*, 1995, Ng *et al.*, 2000], Adaboost algorithm [Schapire *et al.*, 1998, Iyer *et al.*, 2000, Shapire and Singer, 2000, Schapire, 2002, Liu et al, 2002].

3.12. Using unlabeled data to improve classification

All of the ML classifiers require fairly large training collections of preclassified documents. The task of manually labeling a large number of documents, although much less costly than manually creating a classification knowledge base, is still usually quite a chore. On the other hand, unlabeled documents usually exist in abundance, and any amount of them can be acquired with little cost. Therefore, the

ability to improve the classifier performance by augmenting a relatively small number of labeled documents with a large number of unlabeled ones is very useful for applications. The two common ways of incorporating knowledge from unlabeled documents are expectation maximization (EM) and co-training. EM works with probabilistic generative classifiers such as Naive Bayes. The idea is to find the most probable model given both labeled and unlabeled documents. The EM algorithm performs the optimization in a simple and appealing way: First, the model is trained over the labeled documents. Then the following steps are iterated until convergence in a local maximum occurs: E-step: the unlabeled documents are classified by the current model. M-step: the model is trained over the combined corpus. In the M-step, the category assignments of the unlabeled documents are assumed to be fractional according to the probabilities produced by the E-step. Co-training works with the documents, for which two views are available, providing two different document representations, both of which are sufficient for classification.

For example, a Web page may have its content as one view and the anchor text appearing in the hyperlinks to the page as another. In the domain of MedLine papers, the abstract may be one view and the whole text another. The co-training is a bootstrapping strategy in which the unlabeled documents classified by means of one of the views are then used for training the classifier using the other view, and vice versa. Both EMand co-training strategies have experimentally shown a significant reduction (up to 60%) in the amount of labeled training data required to produce the same classifier performance.

3.13. Multilingual text categorization

3.13.1. Importance of multilingual categorization

Several reasons are at the origin of the increasing interest of multilingual data treatment during these last years, notably:

1. The massive expansion of Internet and the increasing availability of textual data on it.
2. The increasing number of internauts (internet users) who use languages other than French and English.
3. The new tendency of globalization that deals with various domains, notably: politics, economy, and culture.
4. Apparition of cooperation zones between countries, such as: the European Union EU, the Asia-Pacific Forum APF.
5. The necessity of cooperation and scientific exchange between universities belonging to different countries and use different languages.

1. Availability of multilingual data collections on internet.

The availability of multilingual digitized collections made on the Internet has created for users new needs to find and understand relevant information, whatever the language and the form of storage [Peters and Sheridan, 2001]. But the main problem that it is not always possible for a user to formulate a query in any language.

2. The massive expansion of the Internet.

English is the mother language of about 19,26 % of the world population and yet it is the dominant language in science namely: data collections, articles of scientific journals, conference papers and other various resources on this network. This domination is going backwards to open the way for a multilingual global network.

Table 3.4. Top Ten Languages Used in the Web.
(Source: Internet World stats- www.internetwordstats.com/stats7.htm, November 30, 2015)

Top ten languages in the Internet	World Population for this Language (2015 estimate)	Internet Users by Language	Internet Penetration (% Population)	Users Growth in Internet (2000 - 2015)	Internet Users % of World Total (Participation)
English	1,398,283,969	**872,950,266**	62.4 %	520.2 %	25.9 %
Chinese	1,398,335,970	**704,484,396**	50.4 %	2,080.9 %	20.9 %
Spanish	441,052,395	**256,787,878**	58.2 %	1,312.4 %	7.6 %
Arabic	375,241,253	**168,176,008**	44.8 %	6,592.5 %	5.0 %
Portuguese	263,260,385	**131,903,391**	50.1 %	1,641.1 %	3.9 %
Japanese	126,919,659	**114,963,827**	90.6 %	144.2 %	3.4 %
Russian	146,267,288	**103,147,691**	70.5 %	3,227.3 %	3.1 %
Malay	286,937,168	**98,915,747**	34.5 %	1,626.3 %	2.9 %
French	385,389,434	**97,729,532**	25.4 %	714.9 %	2.9 %
German	95,324,471	**83,738,911**	87.8 %	204.3 %	2.5 %
Top 10 languages	**4,917,011,992**	**2,632,248,147**	**53.5 %**	**787.0 %**	**78.2 %**
Rest of the Languages	2,342,890,251	**734,013,009**	31.3 %	1,042.9 %	21.8 %
World Total	**7,259,902,243**	**3,366,261,156**	**46.4 %**	**832.5 %**	**100.0 %**

To understand Table 3.4, we note the following:

- Internet Penetration is the ratio between the sum of Internet users speaking a language and the total population that speaks that specific language. For example, the internet penetration of English is calculated as follows: 872,950,266/1,398,283,969 = 62.4%.
- Internet Users for a specific language (%) = sum of Internet users speaking this language/total population connected in internet. (Example: for English % = 872,950,266/3,366,261,156 = 25.9%)

According to the statistics published by international organizations such as UNESCO, International Telecommunications Union, U.S. Census Bureau and some official websites such as Internet World Stats, Approximately 3,366,261,156 people which represents 46,4 % of the world population are connected in internet. They are distributed as follows: 872,950,266 (25.9 %) are English users (speak or read English), 704,484,396 (20,9 %) for Chinese, 256,787,878 (7,6%) for Spanish, 168,176,008 (5.0 %) for Arabic, 97,729,532 (2.9 %) for French, and 83,738,911 (2.5 %) for German.

The top ten languages used in the web are respectively: English (25.9%), Chinese (20.9%), Spanish (7.6%), Arabic (5%), Portuguese (3.9%), Japanese (3.4%), Russian (3.1%), Malay (2.9%), French (2.9%), and German (2.5%) with a total of 2,632,248,147 which represent 78.2 of the total connected population. 734,013,009 (21.8%) for the other languages.

We can also conclude from Table 3.4 that the number of internet users for some languages had grown faster than others in the last fifteen years (2000-2015). For example, we find Arabic in the first position with a ratio of 6592.5 %, Russian with a ration of 3227.3 %, Chinese (2080.9 %), etc.

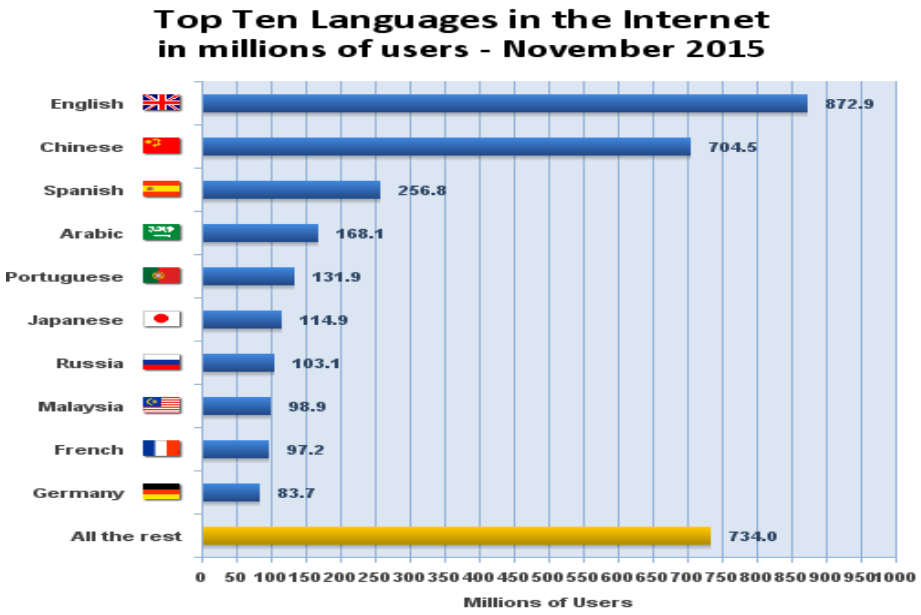


Figure 3.3. Top ten languages in the internet in millions of users (November 2015)

3. Effect of the world globalization.

The globalization of the world is growing. Several areas of political and economic cooperation have been created in recent years. For example, the Enlargement of the European Union is the origin of several research projects on multilingualism such as: EMIR project (European Multilingual Information Retrieval) that supports the languages: English, French, German, Dutch and Russian. MEDLIB is the project that aims to democratize access to cultural heritage of Mediterranne [Jalam, 2003].

4. Flexibility and speed of networks.

The expansion of network infrastructures in particular the internet and the evolution of data transmission speed over these networks, as well as the remote access to very large amounts of data with reasonable costs, all these factors facilitate data duplication by copy. That means data are not rare now as it was before which reduced the value of their grouping for the benefit of the information or the potential knowledge they hold [Jalam, 2003].

3.13.2. Multilingual information retrieval

Few users are able to make retrieval queries in foreign languages. However, the availability of means of automatic translation (the Google translator for example) allow the user to translate the texts of his queries to satisfy their research needs of multilingual information. Several multilingual information retrieval approaches have been proposed, including: approaches based on automatic translation, approaches based on multilingual thesaurus and dictionary-based approaches.

3.13.2.1. Approaches based on automatic translation:

The goal of such information retrieval is to find documents expressed in a target language that can be sometimes different from the language of the formulated query. These documents are similar in the sense of similarity measure of the query expressed in the source language. In the other hand, automatic translation aims to produce a readable and reliable version of a document from a source language to a target language.

The translation of a collection of documents into the query language (the target language) is very expensive, especially for large size collections. Hence, it is best to focus on the translation of queries rather than documents [Peter and Sheridan, 2001]. But queries themselves consist of a set of words with little or no syntactic structure, so, it will be difficult to apply methods of grammatical analysis and language identification. This is the origin of several weaknesses for these approaches.

3.13.2.2. Approaches based on multilingual thesaurus

A controlled vocabulary is a finite set of concepts that must be associated with a collection of texts. Thus, a document will be represented by one or more concepts. A multilingual thesaurus is an extension of the concept of monolingual controlled vocabulary. It can be seen as a set of monolingual thesaurus. To find written information in other languages, users formulate queries in their mother languages, then, the system joins with the multilingual thesaurus established automatically an internal relation between the query terms and concepts of the thesaurus. This approach suffers from some limitations relating to the controlled vocabulary which remains expensive to build and maintain, difficult to update, and requires additional efforts to train users who should use such systems [Peter and Sheridan, 2001].

3.13.2.3. Approaches based on the use of dictionaries.

The use of computerized multilingual dictionaries is available as a solution for multilingual information retrieval. These tools are increasingly available online. Some researchers used to translate queries and have achieved acceptable results but remain far from those obtained by the monolingual research. [Peter and Sheridan, 2001] postpone three reasons for this: (1) general dictionaries are generally poor of specialized vocabularies. (2) Errors due to the translation of compound words (railway, social security). (3) The problem of ambiguity.

3.13.3. Proposed solutions for multilingual text categorization

[Jalam, 2003] provides a general solution to solve the problem of multilingual text categorization. This solution is based on a predictive model built on a corpus of texts written in undefined language and having as objective to infer on texts in any language. The inference phase includes two essential steps:

1. The automatic detection of the language of text that to categorize.
2. The automatic translation of the text into the language of reference (the prediction model language).

For this, we suppose that we have the following text collections: D_1, D_2, ..., D_k and k languages L_1, L_2, ..., L_k. Each collection D_k contains texts written in the language L_k.

t_{jk} be a text d_j belonging to the collection D_k. t_{jk} may be associated with one or more classes below $c_i \in C$

To implement this solution, the author proposes three possible schemes: [Jalam, 2003]

3.13.3.1. Scheme 1: The trivial scheme:

It is an extension of the usual monolingual categorization scheme, except that the multilingual one consists in learning $|L|$ prediction models (one model for each language). So, the categorization of a new text is done according to the following two steps:

1. For each collection D_K (language L_k), we build a prediction model M_K.
2. For each text to classify, we perform the following tasks:
   a. Identify its language.
   b. Apply predictive model that corresponds to the language.
   c. Translate the text into the target language.

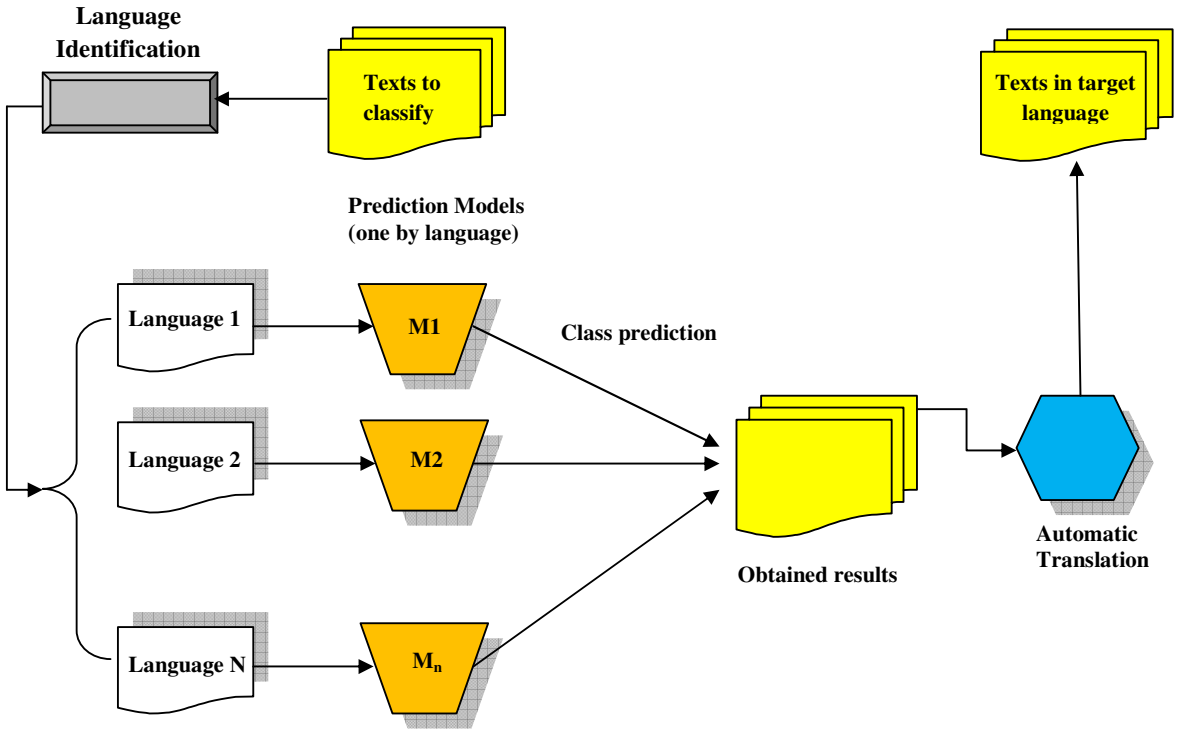


Figure 3.4. The trivial scheme (The extension of the monolingual categorization scheme [Jalam, 2003])

The advantage of this scheme is that the translation is not involved in learning, and therefore no distortion of information or loss is made at this stage. But this scheme still has its limits. It requires perform learning for each language, this suppose having a sufficient number of labeled texts in each language and in each class. This seems difficult, especially for rare languages on the web.

3.13.3.2. Scheme 2: Using a single language for learning.

This second scheme partially overcomes the two disadvantages seen in the first scheme, because here we use a single prediction model. And the categorization process will be as follows:

1. Get the text to classify.
2. Identify its language.
3. If the text language is supported by the prediction model translator, we translate the text into the learning language. (prediction model language).
4. Apply the prediction model to classify the text.

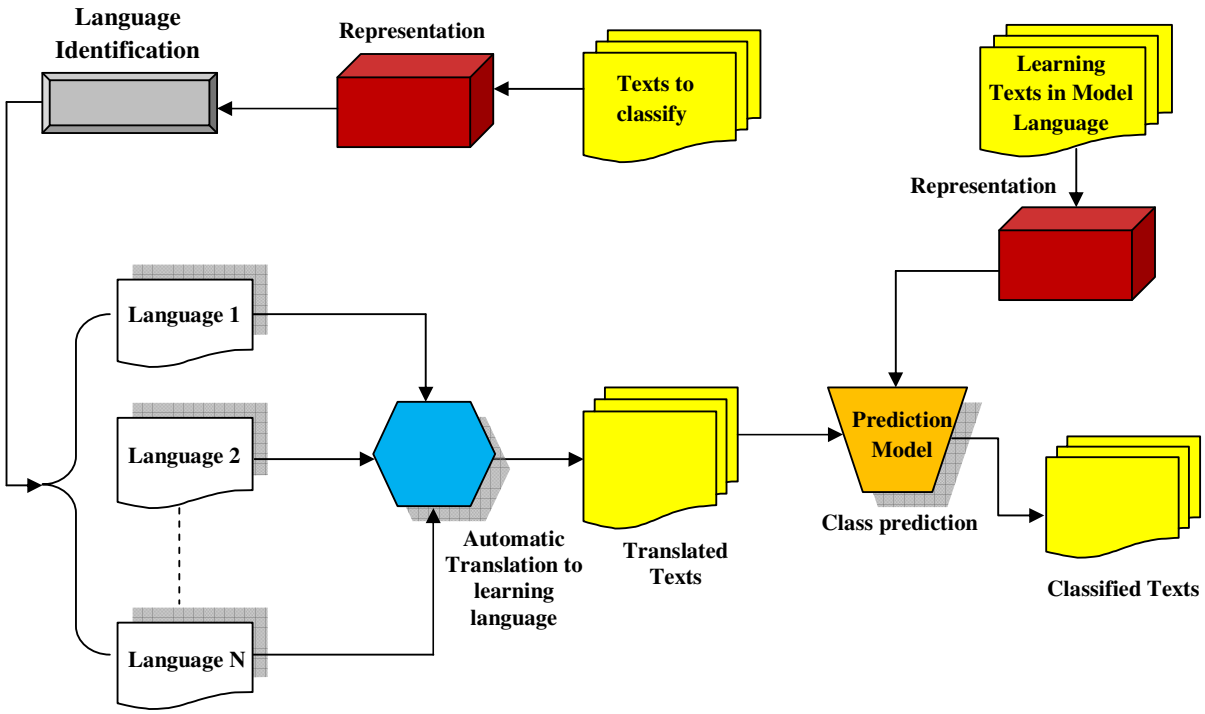


Figure 3.5. Using a Single Learning Language and a Single Prediction Model [Jalam, 2003].

3.13.3.3. Scheme 3: Mix the training sets.

In this scheme, the translation occurs in two phases: learning and classification.

Suppose we have a set *D* of labeled texts written in different languages, we proceed to the following steps:

1. Translate the texts of the set *D* into a unique learning language L_{Learn}.
2. Group all translated texts in a unique set $\grave{D}$.
3. Proceed to the phase of labels' learning (Classes' learning), the model will be learned from these translated texts and will be ready to classify a new text.
4. Identify the language of the new text to classify.
5. If the language is supported by the predictive model translator, we translate the text into the model language L_{Learn}.
6. Apply the prediction model to classify the new text.

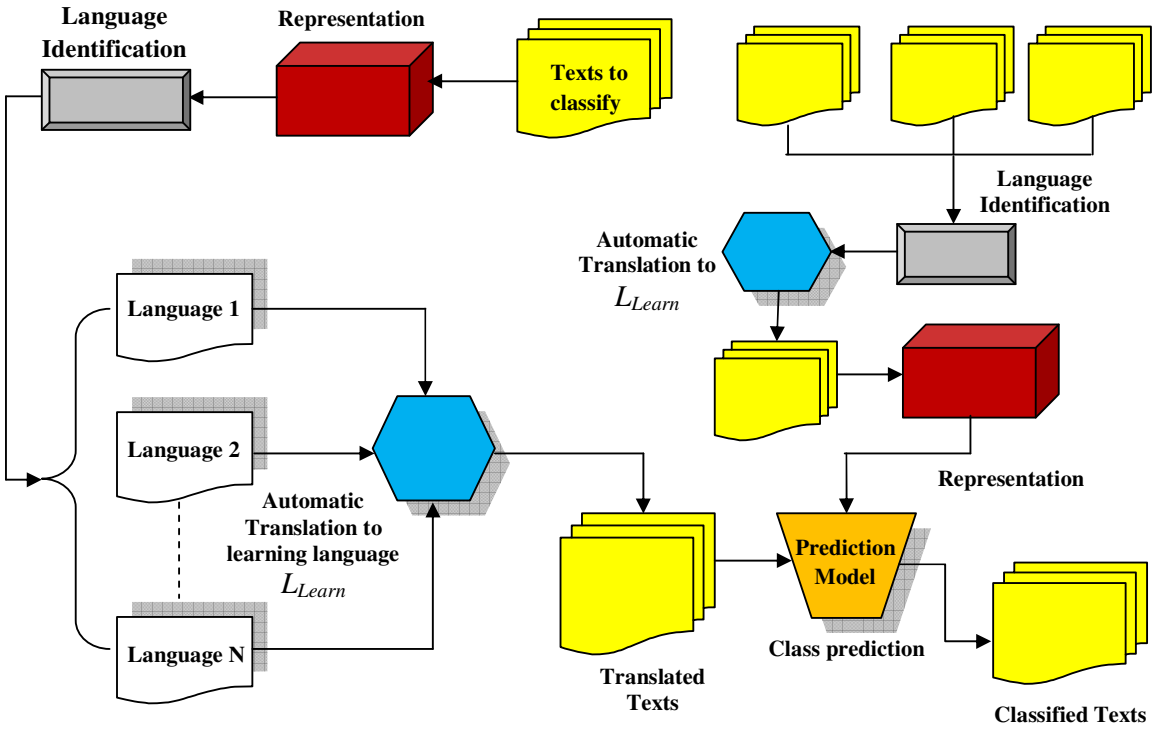


Figure 3.6. Mix the training sets and use a single learning language [Jalam, 2003]

Finally, we note that we have tested the three schemes seen previously in several works of master, they gave good results.

It remains to note that the performance of the prediction model based on one of the three previous schemes is directly influenced by the quality of the language identification of the text and its translation to the target language (the prediction model language).

3.13.4. The main phases of multilingual text categorization

3.13.4.1. Language Identification: consists in assigning a text to a language in which it is written. This identification has become extremely important as textual data in various languages are growing their way in the global network. Several significant elements may be considered during the identification of the language, including: the presence of some characters [Mustonen, 1965, Souter et al, 1994], the presence of some words [Souter et al, 1994, Giguet, 1998], or even the frequency of n-grams [Beesly, 1988, Cavnar and Trenkle, 1994, Dunning, 1994, Grefenstette, 1995]. More details about this subject are explained in Chapter 6.

3.13.4.2. Automatic Translation: This is an important phase that appears in schemes 2 and 3. We note here that it is not necessary to have a perfect translator which never exists in reality, because, the goal here is not the strict semantic translation of the text, but only translating keywords that allow a good thematic classification of the text. The Systran system developed by the US Army is one of the systems adopted for this task.

3.13.4.3. Text categorization: it is a very important task that consists in assigning labels (categories/classes) to a given text based on a set of previous labeled texts (by learning). The interest of such categorization approach is to organize knowledge so that they can perform some specific treatments including; information retrieval or efficient information extraction. The need for automatic classification were felt both on the internet (search engines) and within companies (classification of internal documents, dispatches of news agencies, etc ...). We can distinguish in the field of automatic classification two approaches: unsupervised classification for which groups of documents are formed automatically by the machine during treatment according to similarity criteria and supervised classification (categorization) in which these groups are defined in advance by an expert.

3.14. Summary

Text categorization is the process of finding the correct topic (class/category/label) of a text based on a given set of predefined categories (topics/subjects) and a collection of labeled texts. It is constituted of two big steps: the learning step and the categorization step. Each step is composed of several tasks such as: preprocessing, vector representation, terms coding, features selection, predictive model building, applying the predictive model on the text to categorize.

In this chapter, we have treated many important topics related to the TC field including: definition of TC problem, its applications, some related difficulties, representation methods, coding models, features reduction and features selection, approaches of TC.

We have also dedicated a large part of this chapter for the multilingual text categorization. For which we treated many important elements such as: its importance, the most known approaches of multilingual information retrieval, some proposed solutions (schemes), the main phases of multilingual text categorization process.

CHAPTER 4

Machine Learning Algorithms for Text Categorization

Chapter Content

4.1. Preface

The field of machine learning ML is concerned with the question of how to construct computer programs that automatically improve with experience. In the last years, many successful applications based on machine learning algorithms have been developed, ranging from data-mining applications such as: detection of fraudulent credit card transactions, cross-selling, churn analysis, risk management, sales forecast, to text mining applications such as: document categorization, document clustering, information retrieval, information extraction, to autonomous vehicles that learn to drive on public highways. At the same time, there have been important advances in the theory and algorithms that form the foundations of ML field.

In the ML approach to text categorization, the classifier is built automatically by learning the properties of categories from a set of preclassified training documents (the training set). In the ML terminology, the learning process is an instance of supervised learning because the process is guided by applying the known true category assignment function on the training set. The unsupervised version of the classification task is called clustering. There are many approaches to classifier learning; some of them are variants of more general ML algorithms, and others have been created specifically for categorization. [Feldman and Sanger, 2007]

Our main goal through this chapter is to present an overview of the key ML algorithms that are mostly used in the field of text mining, in particular, in text categorization subfield. We will explain the principle of each algorithm, giving its advantages and inconvenient, and for some of them we present a short pseudo-code and some illustrative examples.

4.2. The text classification problem

In text classification, we define a description of a document $d \in X$, where X is the document space; and a fixed set of classes $C=\{c_1, c_2, \ldots, c_J\}$. Classes are also called categories or labels. Typically, the document space X is some type of high-dimensional space, and the classes are human defined for the needs of an application

Given a training set D of labeled documents (d, c), where $(d, c) \in X \times C$. For example:

(d, c)=(*"Beijing joins the World Trade Organization", China)* for the one-sentence document <"*Beijing joins the World Trade Organization"*> and the class <*China*>. [Manning et al, 2008]

Using a learning method or learning algorithm, we then wish to learn a classifier or classification function γ that maps documents to classes:

$$\gamma: X \to C \tag{4.1}$$

This type of learning is called supervised learning because a supervisor (the supervised learning is a human who defines the classes and labels of training documents) serves as a teacher directing the learning process. We denote the supervised learning method by φ and write $\varphi(D) = \gamma$. The learning method φ takes the training set D as input and returns the learned classification function γ.

The learning method φ can be one of the most known methods (algorithms) in machine learning field such as: Naïve Bayes (NB), k-nearest neighbor, SVM, etc. [Sebastiani, 2002]

How to build a classifier γ? The answer to this question is a key to understand the categorization process. The training set provides some typical examples for each class, so that we can learn the classification function γ. Once we have learned γ, we can apply it to the test set (or test data).

It's also important to note that in such application of text categorization, a document can be a member of exactly one class. It's the case of *one-of problems* or *a single label classification*. Unfortunately, this is not the most appropriate case. Another case is when a document can be a member of two or more classes. This type of classification problem is referred to as *any-of problem* or *multilabel categorization*. Whatever the case, our goal in text classification is to obtain high accuracy on test data or new data. We know that it is easy to achieve high accuracy on the training set (e.g., we can simply memorize the labels). But high accuracy on the training set in general does not mean that the classifier will work well on new data in an application. When we use the training set to learn a classifier for test data, we make the assumption that training data and test data are similar or from the same distribution.

4.3. Learning algorithms used in text categorization

4.3.1. Naïve Bayes classifier

4.3.1.1.The multinomial model

The first supervised learning method we introduce is the multinomial Naive Bayes or multinomial NB model [Maron and Kuhns, 1960, Lewis, 1998, Kołcz and Yih, 2007, Manning et al, 2008], a probabilistic learning method. The probability of a document *d* being in class *c* is computed as:

$$P(c|d) \propto P(c) \prod_{1 \le k \le n_d} P(t_k|c) \tag{4.2}$$

Where:

$P(t_k|c)$*:* is the conditional probability of term t_k occurring in a document *d* of class *c*. We interpret $P(t_k|c)$ as a measure of how much evidence t_k contributes that *c* is the correct class.

P(c): is the prior probability of a document *d* occurring in class *c*. If a document's terms do not provide clear evidence for one class versus another, we choose the one that has a higher prior probability.

$<t_1, t_2, \ldots, t_{n_d}>$*:* are the tokens in *d* that are part of the vocabulary we use for classification.

n_d: is the number of such tokens in *d*. For example, $<t_1, t_2, \ldots, t_{n_d}>$ for the one-sentence document <*Beijing and Taipei join the WTO*> might be <Beijing, Taipei, join, WTO>, with $n_d = 4$, if we treat the terms <*and, the*> as stop words.

As we said previously, our goal in text classification is to find the best class for the document. So, the best class in NB classification is the most likely or maximum a posteriori (MAP) class c_{map}:

$$c_{map} = \text{Argmax}_{c \in C} \hat{P}(c|d) = Argmax_{c \in C} P(c) \prod_{1 \le k \le n_d} P(t_k|c) \tag{4.3}$$

We can summarize this equation as follows:

- We calculate the probabilities$P(c)$, $P(\bar{c})$ of each class *c* and $\bar{c}$ in the data set *D*.
- We calculate the conditional probabilities $P(t_k|c)$, $P(t_k|\bar{c})$ of each term t_k.
- We calculate the product R_1 of all the conditional probabilities $P(t_k|c)$, and the product $\bar{R}_1$ of all the conditional probabilities $P(t_k|\bar{c})$.
- We multiply $P(c)$ by R_1 and $P(\bar{c})$ by $\bar{R}_1$ we obtain c_{map} and $\bar{c}_{map}$
- We assign the new document to the class having the higher result (for example we assign the document *d* to the class *c* if $c_{map} > \bar{c}_{map}$).

In equation (4.3) we use $\hat{P}$ for P because we do not know the true values of the parameters $P(c)$ and $P(t_k|c)$, but estimate them from the training set.

In Equation (4.3), we must be careful because many conditional probabilities are multiplied, one for each position $1 \leq k \leq n_d$. This can result in a floating point underflow (gives a very small result$\ll 0$). So, it is better to perform the computation by adding logarithms of probabilities instead of multiplying probabilities. And in this case, the class with the highest log probability score is still the most probable; (based on the property ($log(xy) = log(x) + log(y)$) and the logarithm function is monotonic. Hence, the maximization that is actually done in most implementations of NB is:

$$C_{map} = Argmax_{c \in C}[\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)] \quad (4.4)$$

Where:
The conditional parameter $\log \hat{P}(t_k|c)$ is a weight that indicates how good an indicator t_k is for c.
The prior $\log \hat{P}(c)$ is a weight that indicates the relative frequency of c (More frequent classes are more likely to be the correct class than infrequent classes).
The sum of log prior $log\, \hat{P}(c)$ and term weights $\log \hat{P}(t_k|c)$ is then a measure of how much evidence there is for the document being in the class, and Equation (4.4) selects the class for which we have the most evidence.
We note also:

$$\hat{P}(c) = \frac{N_c}{N} \quad (4.5)$$

Where:
N_c: is the number of documents in class c.
N: is the total number of documents.

Similarly, we estimate the conditional probability $\hat{P}(t_k|c)$ as the relative frequency of term t in documents belonging to class c, and it is calculated as follows:

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}} \quad (4.6)$$

Where:
T_{ct}: is the number of occurrences of t in training documents belonging to the class c, including multiple occurrences of a term in a document. T_{ct} is a count of occurrences in all positions k in the documents in the training set.
$T_{ct'}$: is the number of terms constituting the class c including multiple occurrences of terms.

One problem which can occur when applying the previous equation (4.6) is when we calculate a conditional probability of a rare term t_r and t_r doesn't occur in the training data for a class c_r. In this case we have $\hat{P}(t_r|c_r) = 0$
And consequently, we will have also $c_{map}=0$ because we are multiplying the conditional probabilities for all terms in equation (4.2).
To avoid such situation, we must simply add 1 to each count in (4.6) as given in (4.7)

$$\hat{P}(t|c) = \frac{T_{ct}+1}{\sum_{t' \in V}(T_{ct'}+1)} = \frac{T_{ct}+1}{(\sum_{t' \in V} T_{ct'})+B} \quad (4.7)$$

Where: $B = |V|$ is the number of terms in the vocabulary of the training set.

Now let's give the complete algorithm of the basic NB:

```
TrainMultinomialNB(C, D)
  V ← ExtractVocabulary(D)  /* to build the vocabulary of the training data D
  N ← CountDocs(D)          /* Total number of docs in D
  for each c ∈C do          /* for each class c
    Nc ← CountDocsInClass(D, c) /* number of docs in class c
    prior [c] ← Nc/N   /* Calculate the probability of c
                       /* in the training data D
    Text_c ← ConcatenateTextOfAllDocsInClass(D, c)
          /* Regroup the texts of the class c to find its vocabulary
     for each t ∈V  do   /* for each term t of the total vocabulary V
       T_ct ← CountTokensOfTerm(text_c, t)  /*calculate the frequency of t∈ c in D
     for each t ∈V do   /* for each term t of the total vocabulary V
       condprob[t][c] ← (T_ct+1)/(∑_t'(T_ct'+1))   /* calculate the probability P̂(t_k|c)
  Return V, prior, condprob

ApplyMultinomialNB(C, V, prior, condprob, d)  /* Apply the bayes classifier on
                                              /* the test set or a new document
  W ← ExtractTokensFromDoc(V, d)       /* Find the vocabulary W of the test set
                                       /* or the new document
  for each c ∈C do             /* for each class c
    score[c] ← log prior [c]   /* calculate the probability of c in D
    for each t ∈W do           /* for each term t of the new set
      score[c] += log condprob[t][c]
Return arg max_c∈C score[c]
```

Algorithm 4.1 Naive Bayes algorithm (multinomial model): Training and testing.

Example 4.1: Multinomial model of NB: we have the following data set:

Table 4.1. Data for parameter estimation examples

	Doc Id	Words in document	*c: d∈class{China}* *c̄: d∉class {China}*
Training set	*d1*	Chinese Beijing Chinese	Yes
	d2	Chinese Chinese Shanghai	Yes
	d3	Chinese Macao	Yes
	d4	Tokyo Japan Chinese	No
Test set	*d5*	Chinese Chinese Chinese Tokyo Japan	?

As it's shown in table 3.1 we have two classes c and $\bar{c}$
c: the document d_i belongs to the class $\{China\}$ (mentioned by Yes)
$\bar{c}$: the document d_i doesn't belong to the class $\{China\}$ (mentioned by No)
We have also: $\{d_1, d_2, d_3, d_4\} \in c$
$\{d_4\} \notin c$ i.e,. $\{d_4\} \in \bar{c}$

We need to classify the test document d_5. So, we must follow the following steps:

1. We calculate : $\hat{P}(c) = \frac{N_c}{N} = 3/4$, $\hat{P}(\bar{c}) = \frac{N_c}{N} = 1/4$
2. We find the vocabulary V of the training data set
   $V = \{Chinese, Beijing, Shanghai, Macao, Tokyo, Japan\}$ (we take each term one time)
   $B = |V| = 6$
3. We calculate the number of terms $t_{ct'}$ constituting c and $\bar{c}$
   $t_{ct'}(c) = 8$; $t_{ct'}(\bar{c}) = 3$
4. We calculate t_{ct} the number of occurrences of the term t in the training document that belong to the class c and $\bar{c}$ respectively including multiple occurrences
   $T_{ct}(c, Chinese) = 5$; $T_{ct}(\bar{c}, Chinese) = 1$
   $T_{ct}(c, Beijing) = 1$; $T_{ct}(\bar{c}, Beijing) = 0$
   $T_{ct}(c, Shanghai) = 1$; $T_{ct}(\bar{c}, Shanghai) = 0$
   $T_{ct}(c, Macao) = 1$; $T_{ct}(\bar{c}, Macao) = 0$
   $T_{ct}(c, Tokyo) = 0$; $T_{ct}(\bar{c}, Tokyo) = 1$
   $T_{ct}(c, Japan) = 0$; $T_{ct}(\bar{c}, Japan) = 1$
5. We calculate the conditional probabilities according to the equation (4.7)
   $\hat{P}(Chinese|c) = (5+1)/(8+6) = 6/14 = 3/7$
   $\hat{P}(Chinese|\bar{c}) = (1+1)/(3+6) = 2/9$
   $\hat{P}(Beijing|c) = (1+1)/(8+6) = 2/14 = 1/7$
   $\hat{P}(Beijing|\bar{c}) = (0+1)/(3+6) = 1/9$
   $\hat{P}(Shanghai|c) = (1+1)/(8+6) = 2/14 = 1/7$
   $\hat{P}(Shanghai|\bar{c}) = (0+1)/(3+6) = 1/9$
   $\hat{P}(Macao|c) = (1+1)/(8+6) = 2/14 = 1/7$
   $\hat{P}(Macao|\bar{c}) = (0+1)/(3+6) = 1/9$
   $\hat{P}(Tokyo|c) = (0+1)/(8+6) = 1/14$
   $\hat{P}(Tokyo|\bar{c}) = (1+1)/(3+6) = 2/9$
   $\hat{P}(Japan|c) = (0+1)/(8+6) = 1/14$
   $\hat{P}(Japan|\bar{c}) = (1+1)/(3+6) = 2/9$

   The dominators are (8+6) and (3+6) because the lengths of $text_c$ and $text_{c'}$ are 8 and 3 respectively, and because the constant B in (3.7) is 6 as the vocabulary consists of six terms.
6. We calculate then the scores of the test document $\hat{P}(c|d_5)$, $\hat{P}(\bar{c}|d_5)$:
   $d_5 = \{Chinese\ Chinese\ Chinese\ Tokyo\ Japan\}$ ➔

   $\hat{P}(c|d_5) = p(c) \prod_{1 \le k \le n_d} p(t_k|c) = 3/4 * [\hat{P}(Chinese|c) * \hat{P}(Chinese|c) * \hat{P}(Chinese|c) * \hat{P}(Tokyo|c) * \hat{P}(Japan|c) = 3/4 * [(3/7)^3 * 1/14 * 1/14] \approx 0.0003$

$\hat{P}(\bar{c}|d_5) = p(\bar{c}) \prod_{1 \le k \le n_d} p(t_k|\bar{c})$ = 1/4*[$\hat{P}(Chinese|\bar{c})$* $\hat{P}(Chinese|\bar{c})$* $\hat{P}(Chinese|\bar{c})$ * $\hat{P}(Tokyo|\bar{c})$* $\hat{P}(Japan|\bar{c})$ = 1/4*[$(2/9)^3$ * 2/9*2/9] ≈ 0.0001

Thus, because $\hat{P}(c|d_5) > \hat{P}(\bar{c}|d_5)$ the NB classifier assigns the test document d_5 to the class c {*China*}

4.3.1.2.The Bernoulli model

Another model of NB classifier is the Bernoulli model [McCallum and Nigam, 1998]. It's equivalent to the binary model which generates an indicator for each term of the vocabulary, either 1 indicating presence of the term in the document or 0 indicating absence. The Bernoulli model has the same time complexity as the multinomial model.

The different generation models imply different estimation strategies and different classification rules. For example, the Bernoulli model estimates $\hat{P}(t|c)$ as the fraction of documents of class c that contain term *t*. In contrast, the multinomial model estimates $\hat{P}(t|c)$ as the fraction of tokens in documents of class *c* that contain term *t* (Equation (4.7)).

When classifying a test document, the Bernoulli model uses binary occurrence information, ignoring the number of occurrences. Thus, as a result, the Bernoulli model typically makes many mistakes when classifying long documents. For example, it may assign an entire book to the class *China* because of a single occurrence of the term *China* which occurs in it.

```
TrainBernoulliNB(C, D)
1  V ← ExtractVocabulary(D)
2  N ← CountDocs(D)
3  for each c ∈ C do
   4  N_c ← CountDocsInClass(D, c)
   5  prior [c] ← N_c/N
   6  for each t ∈ V do
      7  N_ct ← CountDocsInClassContainingTerm(D, c, t)
      8  condprob[t][c] ← (N_ct + 1)/(N_c + 2)
9  Return V, prior, condprob
ApplyBernoulliNB(C, V, prior, condprob, d)
1  V_d ← ExtractTermsFromDoc(V, d)
2  for each c ∈ C do
   3  score[c] ← log prior [c]
   4  for each t ∈ V do
      5  if t ∈ V_d then
      6  score[c] += log condprob[t][c]
      7  else score[c] += log(1 − condprob[t][c])
8 Return arg max_c∈C score[c]
```

Algorithm 4.2 NB algorithm (Bernoulli model)

Example 4.2: Bernoulli model
Applying the Bernoulli model to the example given in table 3.1, we obtain the following results:

1. We have the same estimates for the priors as before:
   $\hat{P}(c) = \frac{N_c}{N} = 3/4, \quad \hat{P}(\bar{c}) = \frac{N_c}{N} = ¼$

2. The conditional probabilities are:

$\hat{P}(Chinese|c) = (3 + 1)/(3 + 2) = 4/5$
(1) (2) (3)

(1) The term "*Chinese*" belongs to three documents from the class *c*.
(2) Because there are three documents in the class *c*.
(3) Because the constant *B* in (4.7) is 2 (there are two cases to consider for each term, occurrence and nonoccurrence.

$\hat{P}(Chinese|\bar{c}) = (1 + 1)/(1 + 2) = 2/3$
(1) (2) (3)

(1) The term "*Chinese*" belongs to one document from the class $\bar{c}$.
(2) Because there is one document in the class$\bar{c}$.
(3) Because the constant *B* in (4.7) is 2

$\hat{P}(Beijing|c) = \hat{P}(Shanghai|c) = \hat{P}(Macao|c) = (1+1)/(3+2) = 2/5.$
$\hat{P}(Beijing|\bar{c}) = \hat{P}(Shanghai|\bar{c}) = \hat{P}(Macao|\bar{c}) = (0+1)/(1+2) = 1/3.$
$\hat{P}(Tokyo|c) = \hat{P}(Japan|c) = (0+1)/(3+2) = 1/5.$
$\hat{P}(Tokyo|\bar{c}) = \hat{P}(Japan|\bar{c}) = (1+1)/(1+2) = 2/3.$

3. The scores of the test document for the two classes c, $\bar{c}$ are
   d_5 = {*Chinese Chinese Chinese Tokyo Japan*}

   $\hat{P}(c|d_5) = p(c)*[\hat{P}(Chinese|c)*\hat{P}(Tokyo|c)*\hat{P}(Japan|c)*(1-\hat{P}(Beijing|c))$
   $*(1-\hat{P}(Macao|c)) = 3/4*[(4/5) * 1/5*1/5*(1-2/5)*(1-2/5)*(1-2/5)] \approx 0.005$

   We note that: *{Beijing, Shanghai, Macao}* are nonoccurrence terms for d_5 because they belong to the vocabulary *V*, but don't belong to d_5.

   And analogously:
   $\hat{P}(\bar{c}|d_5) = p(\bar{c})*[\hat{P}(Chinese|\bar{c})*\hat{P}(Tokyo|\bar{c})*\hat{P}(Japan|\bar{c})*(1-\hat{P}(Beijing|\bar{c}))$
   $(1-\hat{P}(Shanghai|\bar{c}))*(1-\hat{P}(Macao|\bar{c}))= 1/4*[(2/3) * 2/3*2/3*(1-1/3)*(1-1/3)*(1-1/3)] \approx 0.022$

$\hat{P}(\bar{c}|d_5) > \hat{P}(c|d_5)$ ➔ The classifier assigns the test document to the class $\bar{c}$ = {*not-China*} which is not the same result obtained with the multinomial model.

4.3.1.3.Time complexity of NB classifier

The complexity of computing the parameters is $|C|x|V|$ because the set of parameters consists of $|C|x|V|$ conditional probabilities to calculate.

The preprocessing necessary for computing the parameters (extracting the vocabulary, counting terms, etc.) can be done in one pass through the training data. The time complexity of this component is therefore $|D|xL_{ave}$. Where $|D|$ is the number of documents in the training data set and L_{ave} is the average length of a document. In addition the multinomial model and the model of Bernoulli, other models for NB are described by several authors such as [Eyheramendy et al., 2003].

4.3.1.4. Linear classifiers

In Naive Bayes, classifiers documents are represented as a sequence of terms or a binary vector of the form $<e_1, \ldots, e_m> \in \{0, 1\}^{|V|}$. Another representation for text classification is the vector space model. It represents each document as a vector with one real-valued component, usually a *Tf–Idf* weight, for each term belonging to the document. Here, we will see many methods that are based on this model, for instance; Rocchio and kNN. We note also that a a large number of text classifiers can be viewed as linear classifiers which partition the space of features into regions separated by linear decision hyperplanes.

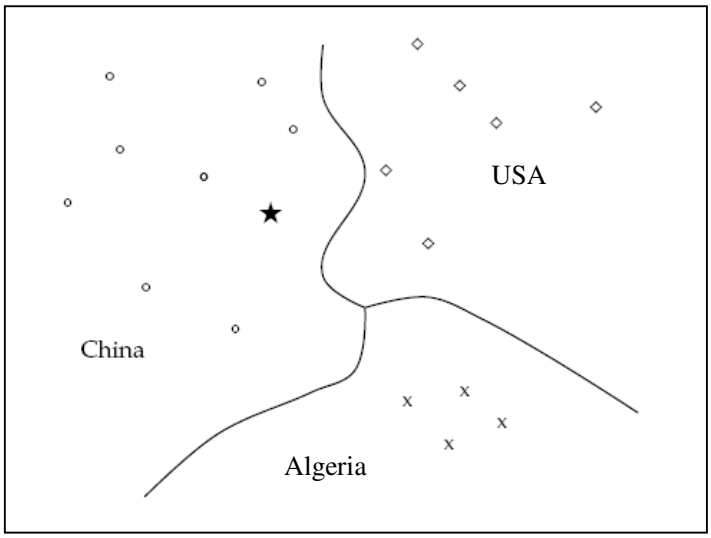


Figure 4.1 Vector space classification into three classes.

Decisions of many vector space classifiers are based on a notion of distance, such as kNN classification. There are several distances measures, the most known are: the Euclidean distance, the cosine similarity and dot product.

4.3.2. **Rocchio classifier**

Rocchio method divides the vector space into regions centered on centroids or prototypes, one for each class, computed prototype as the center of mass of all documents in the class. Rocchio classification is simple and efficient, but inaccurate if classes are not approximately spheres with similar radii. [Rocchio, 1971, Joachims, 1997, Manning et al., 2008]

If we take the figure 4.1 seen above, it shows three classes, *China, USA*, and *Algeria*, in a 2D space. Documents are shown as circles, diamonds, and *Xs*. The decision boundaries in the figure are chosen to separate the three classes, but are arbitrary. To classify a new document, depicted as a star (*) in the

figure, we determine the region it occurs in and assign it the class of that region – *China* in this case. The main work we must do in vector space classification is to define good boundaries between classes because they determine the classification decision. One best way to do this is Rocchio classification, which uses centroids to define the boundaries [Han and Karypis, 2000, Tan and Cheng, 2007]. The centroid of a class c is computed centroid as the vector average or center of mass of its members:

$$\vec{\mu}(c) = \frac{1}{|D_c|}\sum_{d\in D_c}\vec{v}(d) \tag{4.8}$$

Where:
D_c: is the set of documents in D whose class is c: $D_c = \{d : <d, c> \in D\}$.
$\vec{v}(d)$: The normalized vector of d.

The boundary between two classes in Rocchio classification is the set of points with equal distance from the two centroids. For example, $|a1| = |a2|$, $|b1| = |b2|$, *and* $|c1| = |c2|$ in the figure 4.2. This set of points is always a line. The generalization of a line in M-dimensional space is a hyperplane, which we define as the set of points $\vec{x}$ that satisfy:

$$\vec{w}^T.\vec{x} = b \tag{4.9}$$

Where:
$\vec{w}$: is the M-dimensional normal vector of the hyperplane.
B: is a normal vector constant.

This definition of hyperplanes includes lines (any line in 2D can be defined by $w_1x_1 + w_2x_2 = b$) and 2-dimensional planes (any plane in three dimensions (3D) can be defined by $w_1x_1 + w_2x_2 + w_3x_3 = b$).
A line divides a plane in two, a plane divides 3D space in two, and hyperplanes divide higher-dimensional spaces in two.

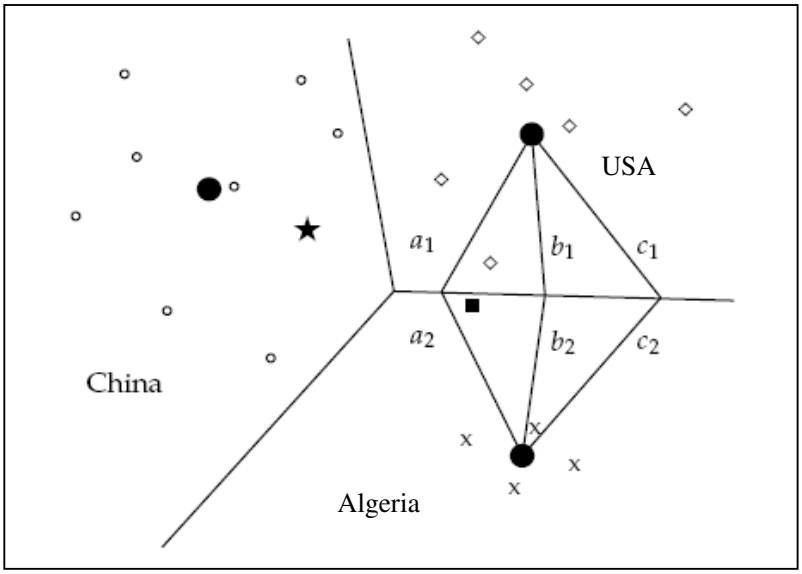


Figure 4.2 Rocchio classification.

Thus, the boundaries of class regions in Rocchio classification are hyperplanes. The classification rule in Rocchio is to classify a point in accordance with the region it falls into. Equivalently, we determine the centroid $\vec{\mu}(c)$ that the point is closest to and then assign it to *c*. For example, the star in Figure 4.2. It is located in the *China* region of the space and Rocchio therefore assigns it to the class *China*.

TRAINROCCHIO(C, D)
1 **for each** $c_j \in C$ **do**
2 $D_j \leftarrow \{d : \langle d, c_j \rangle \in D\}$
3 $\vec{\mu}_j \leftarrow \frac{1}{|D_c|}\sum_{d \in D_c} \vec{v}(d)$
4 **return** $\{\vec{\mu}_1, \ldots, \vec{\mu}_j\}$
APPLYROCCHIO($\{\vec{\mu}_1, \ldots, \vec{\mu}_j\}$, d)
1 **return** Arg $\min_j |\vec{\mu}_j - \vec{v}(d)|$

Algorithm 4.3 Rocchio classification algorithm: Training and testing.

Example 4.3: Rocchio algorithm
We take the document data set seen previously (Example 4.1) and we represent each document by the *TF-IDF* vector representation using the formula:

$$TF - IDF(t,d) = (1 + log_{10}(TF(t,d))) * log_{10}(\frac{4}{df(t)}) \quad \text{if } TF(t,d) > 0 \qquad (4.10)$$

We obtain the following results:

Table 4.2 TF-IDF Vector representation of document data set in Table 4.1.

	Doc. vector	Term weights						
		Chinese	Japan	Tokyo	Macao	Beijing	Shanghai	Class
Training set	$\vec{d}_1$	0	0	0	0	1.0	0	*c*
	$\vec{d}_2$	0	0	0	0	0	1.0	*c*
	$\vec{d}_3$	0	0	0	1.0	0	0	*c*
	$\vec{d}_4$	0	0.71	0.71	0	0	0	$\bar{c}$
Test set	$\vec{d}_5$	0	0.71	0.71	0	0	0	?
Centroids of c and $\bar{c}$	$\vec{\mu}_c$	?	?	?	?	?	?	
	$\vec{\mu}_{\bar{c}}$	?	?	?	?	?	?	

We calculate also the two class centroids as follows:

$\vec{\mu}_c = \frac{1}{|D_c|}\sum_{d \in D_c} \vec{v}(d)$ = 1/3($\vec{d}_1 + \vec{d}_2 + \vec{d}_3$)=1/3[(0, 0, 0, 0, 1, 0) + [(0, 0, 0, 0, 0, 1) + [(0, 0, 0, 1, 0, 0)]
= (0, 0, 0, 1/3, 1/3, 1/3) = (0, 0, 0, 0.33, 0.33, 0.33)

$\vec{\mu}_{\bar{c}} = \frac{1}{|D_{\bar{c}}|}\sum_{d \in D_{\bar{c}}} \vec{v}(d)$ = 1/1($\vec{d}_4$)= 1(0, 0.71, 0.71, 0, 0, 0)= (0, 0.71, 0.71, 0, 0, 0)

The Euclidean distances of the test document $\vec{d}_5$ from the centroids are:

$|\vec{\mu}_c - \vec{d}_5| = |(0, 0, 0, 1/3, 1/3, 1/3) - (0, 0.71, 0.71, 0, 0, 0)| = \sqrt{0^2 + (-0.71)^2 + (-0.71)^2 + \frac{1}{9} + \frac{1}{9} + \frac{1}{9}}$

$= \sqrt{0.5041 + 0.5041 + 0.3333} \approx 1.15$

$|\vec{\mu}_{\bar{c}} - \vec{d}_5| = |(0, 0.71, 0.71, 0, 0, 0) - (0, 0.71, 0.71, 0, 0, 0)| = 0.0$

$|\vec{\mu}_{\bar{c}} - \vec{d}_5| < |\vec{\mu}_c - \vec{d}_5|$ ➔ Rocchio classifier assigns d_5 to $\bar{c}$

The separating hyperplane has the following parameters:

$\vec{w} = (0, -0.71, -0.71, \frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ It is finally: $(\vec{\mu}_c - \vec{d}_5)$

$b = -1/3$

Thus documents in class c when $\vec{w}^T . \vec{d}_i > b$ (d_1, d_2, d_3) and documents in $\bar{c}$ when $\vec{w}^T . \vec{d}_i < b$ (d_4)

Time complexity of Rocchio classifier

Rocchio classification and Naive Bayes have the same linear training time complexity. [Buckley et al., 1994a, Buckley et al., 1994b, Voorhees and Harman, 2005].

4.3.3. *k* nearest neighbor classifier

kNN or *k* nearest neighbor classification determines the decision boundary locally. For 1NN we assign the new document to the class of its closest neighbor. For kNN (k = 2, 3, …), we assign each document to the majority class of its *k* closest neighbors, where *k* is a parameter representing the number of neighbors. kNN requires no explicit training and can use the unprocessed training set directly in classification. It is less efficient than other classification methods in classifying documents. If the training set is large, then kNN can handle nonspherical and other complex classes better than Rocchio. [Yang and Chute, 1994, Yang and Li, 1999, Hastie et al., 2001]

1NN is not very robust. The classification decision of each test document relies on the class of a single training document, which may be incorrectly labeled or atypical. kNN for $k > 1$ is more robust. It assigns documents to the majority class of their *k* closest neighbors, with ties broken randomly.

The parameter *k* in kNN is often chosen based on experience or knowledge about the classification problem at hand. It is desirable for *k* to be odd to make ties less likely. $k = 3$ and $k = 5$ are common choices, but much larger values, between 50 and 100, are also used. An alternative way of setting the parameter is to select the *k* that gives best results on a held-out portion of the training set.

Basic *k*NN Algorithm (*D, C*)
Input : *D*, the set of training documents, the test document, $\mathbf{d_t}$, which is a vector of term weights, and *C*, the set of classes used to label the documents
Output : $c_d \in C$, the class of d_t
foreach document $\mathbf{d_i} \in D$ **do**
| Compute *dist*($\mathbf{d_t}$, $\mathbf{d_i}$), the distance between $\mathbf{d_t}$ and $\mathbf{d_i}$;
end
Select $N \subseteq D$, the set (neighborhood) of *k* closest training objects for d_t;
$c_{dt} = \text{argmax}_{v \in C} \sum_{di \in N} I(v = class(c_{di}))$;
where $I(\cdot)$ is an indicator function that returns the value 1 if its argument is true and 0 otherwise.

Algorithm 4.4 The basic kNN algorithm

Example 4.4: kNN algorithm

We take always the above example, and try to find the closet class of the test document d_5 using kNN classifier ($k = 1$ one neighbor).

We calculate the Euclidean distances of d_5 from the four training documents (d_1, d_2, d_3, d_4), we obtain:

$$|\vec{d}_1 - \vec{d}_5| = |(0, 0, 0, 0, 1, 0) - (0, 0.71, 0.71, 0, 0, 0)|$$
$$= \sqrt{(0-0)^2 + (0-0.71)^2 + (0-0.71)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2}$$
$$= \sqrt{(0.5041) + (0.5041) + 1} = \sqrt{2.0082} \approx 1.4171$$

Analogously:

$|\vec{d}_2 - \vec{d}_5| = |\vec{d}_3 - \vec{d}_5| \approx 1.4171$

$|\vec{d}_4 - \vec{d}_5| = 0.0$ it is the minimal distance ➔ the nearest neighbor of d_5 is d_4 and 1NN assigns d_5 to d_4's class, $\bar{c}$

4.3.3.1. Similarity measures used with kNN algorithm

Several similarity measures (distances) can be used to find the nearest neighbor, among them we can note the following:

1. Inner product (dot product):

$$Inner(\overrightarrow{d_j}, \overrightarrow{d_k}) = \overrightarrow{d_j} \cdot \overrightarrow{d_k} = \sum_{i=1}^{|V|} w_{ij}.w_{ik} \tag{4.11}$$

2. Cosine distance (similarity):

$$Cosine(\overrightarrow{d_j}, \overrightarrow{d_k}) = \frac{\overrightarrow{d_j} \cdot \overrightarrow{d_k}}{\|\overrightarrow{d_j}\|\|\overrightarrow{d_k}\|} = \frac{\sum_{i=1}^{|V|} w_{ij}.w_{ik}}{\sqrt{\sum_{i=1}^{|V|}(w_{ij})^2}\sqrt{\sum_{i=1}^{|V|}(w_{ik})^2}} \tag{4.12}$$

3. Dice distance:

$$Dice(\overrightarrow{d_j}, \overrightarrow{d_k}) = \frac{2*(\overrightarrow{d_j} \cdot \overrightarrow{d_k})}{\|\overrightarrow{d_j}\|^2 \|\overrightarrow{d_k}\|^2} = \frac{2*\sum_{i=1}^{|V|} w_{ij}.w_{ik}}{\sum_{i=1}^{|V|}(w_{ij})^2 + \sum_{i=1}^{|V|}(w_{ik})^2} \tag{4.13}$$

4. Jaccard distance:

$$Jaccard(\overrightarrow{d_j}, \overrightarrow{d_k}) = \frac{\overrightarrow{d_j} \cdot \overrightarrow{d_k}}{\|\overrightarrow{d_j}\|^2 \|\overrightarrow{d_k}\|^2 - \overrightarrow{d_j} \cdot \overrightarrow{d_k}} = \frac{\sum_{i=1}^{|V|} w_{ij}.w_{ik}}{\sum_{i=1}^{|V|}(w_{ij})^2 + \sum_{i=1}^{|V|}(w_{ik})^2 - \sum_{i=1}^{|V|} w_{ij}.w_{ik}} \tag{4.14}$$

5. Overlap distance:

$$Overlap(\overrightarrow{d_j}, \overrightarrow{d_k}) = \frac{\overrightarrow{d_j} \cdot \overrightarrow{d_k}}{Min[\|\overrightarrow{d_j}\|^2, \|\overrightarrow{d_k}\|^2]} = \frac{\sum_{i=1}^{|V|} w_{ij}.w_{ik}}{Min[\sum_{i=1}^{|V|}(w_{ij})^2, \sum_{i=1}^{|V|}(w_{ik})^2]} \tag{4.15}$$

Where:

$\overrightarrow{d_j}, \overrightarrow{d_k}$: Vectors representing documents d_j, d_k in vector space.

w_{ij}, w_{ik}: weights of the term i in documents d_j and d_k successively.

$|V|$: size of the term vocabulary.

$||d_j||$, $||d_k||$: vectors lengths.

6. ***Other distances:*** Minkowski, Manhattan, Chebychev, etc.

4.3.3.2. Probabilistic kNN

There is a probabilistic version of kNN classification algorithm in which we can estimate the probability of membership in class *c* as the proportion of the *k* nearest neighbors in *c*. Figure 4.3 gives an example for $k = 3$. Probability estimates for class membership of the star are: $\hat{P}(circle\ class|star) = 1/3$, $\hat{P}(Xclass|star) = 2/3$, and $\hat{P}(diamond\ class|star) = 0$ (the denominator 3 is the number of neighbors *k*). Thus, 3NN estimate ($\hat{P}(circle\ class|star)$=1/3) and the 1NN estimate $\hat{P}(circle\ class|star) = 1$). i.e., differ with 3NN preferring the *X* class and 1NN preferring the circle class. [Creecy et al., 1992, Indyk, 2004]

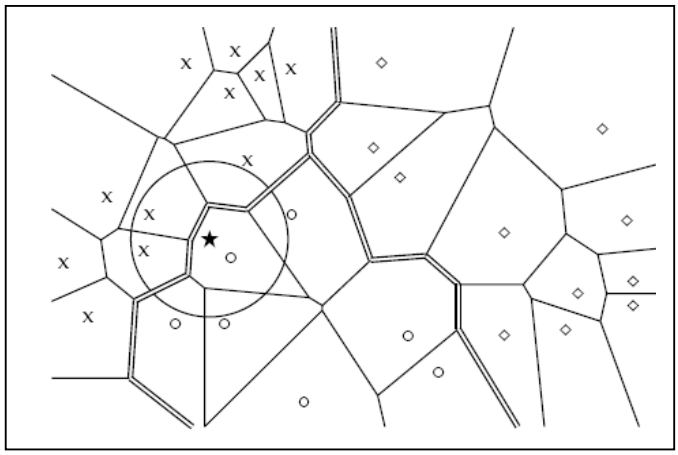


Figure 4.3. The probabilistic version of kNN

```
TRAIN-KNN(C, D)
1  D' ← Preprocess(D)
2  k ← Select-k(C, D')
3  return D', k

APPLY-KNN(C, D, k, d)
1  S_k ← ComputeNearestNeighbors(D', k, d)
2  for each c_j ∈ C do
3     p_j ← |S_k ∩ c_j |/k
4  return arg max_j  p_j
```

Algorithm 4.5 A pseudo code of the probabilistic version of kNN (with preprocessing)

4.3.3.3. Performance of kNN classifier

kNN classifier has some properties that are quite different from most other classifiers. Training a kNN classifier simply consists of determining *k* and preprocessing documents. If we find a value for *k* and do not preprocess documents, then kNN requires no training at all. In practice, we have to perform preprocessing steps like tokenization. In kNN classification, we do not perform any estimation of parameters as we do in Rocchio classification (centroids) or in Naive Bayes (priors and conditional probabilities). kNN simply memorizes all examples in the training set and then compares the test document to them by calculating a distance value. For this reason, kNN is also called *memory-based learning*. It is usually desirable memory-based learning to have a large training data as possible. But large training sets come with a severe efficiency penalty in classification. There are fast kNN algorithms for small dimensionality *M*. There are also approximations for large *M* that give error bounds for specific efficiency gains [Andoni et al., 2006]. But, they have not been tested for extensive

text classification applications. So, it is not clear if they can achieve much better efficiency without a significant loss of accuracy. kNN's effectiveness is close to that of the most accurate learning methods in text classification. The error of 1NN is asymptotically bounded by twice the Bayes error rate (as the training set increases). i.e., if the optimal classifier has an error rate of x, then 1NN has an asymptotic error rate of $2x$. This is due to the effect of noise which can affect the test document and the closest training document.

4.3.4. Support Vector Machine classifier (SVM classifier)

The support vector machine (SVM) algorithm is very fast and effective for text classification problems [Vapnik, 1995, Vapnik, 1998]. In geometrical terms, a binary SVM classifier can be seen as a hyperplane in the feature space separating the points that represent the positive instances of the category from the points that represent the negative instances. The classifying hyperplane is chosen during training as the unique hyperplane that separates the known positive instances from the known negative instances with the maximal margin. The margin is the distance from the hyperplane to the nearest point from the positive and negative sets. It is interesting to note that SVM hyperplanes are fully determined by a relatively small subset of the training instances, which are called the support vectors. The rest of the training data have no influence on the trained classifier. In this respect, the SVM algorithm appears to be unique among the different categorization algorithms. The SVM classifier has an important advantage in its theoretically justified approach to the overfitting problem, which allows it to perform well irrespective of the dimensionality of the feature space. Also, it needs no parameter adjustment because there is a theoretically motivated "default" choice of parameters that has also been shown experimentally to provide the best performance. We will initially motivate and develop SVMs for the case of two-class data sets that are separable by a linear classifier, and then extend the model to non separable data, multiclass problems, and nonlinear models, and also present some additional discussion of SVM performance. [Burges, 1998, Joachims, 1999, Joachims, 2002a]

4.3.4.1. The linearly separable case

For two-class, separable training data sets, such as the one in Figure 4.4, there are lots of possible linear separators. Intuitively, a decision boundary drawn in the middle of the space between data items of the two classes seems better than one which approaches very close to examples of one or both classes. [Joachims, 1998, Dumais et al., 1998, Joachims, 1999, Joachims, 2000, Joachims, 2001]

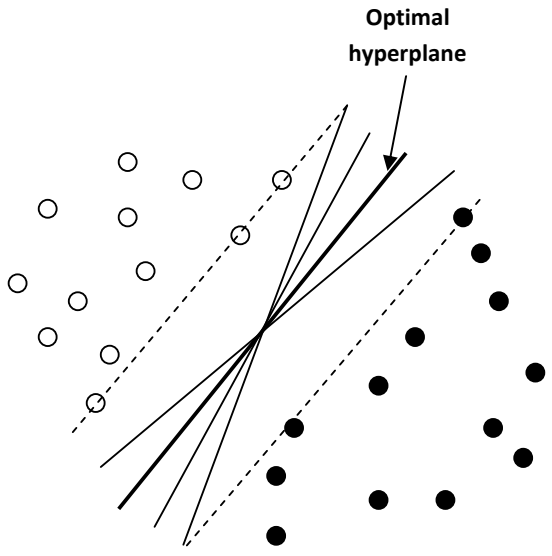


Figure 4.4. An infinity of hyperplanes that separating two linearly separable classes.

The SVMs look for a decision surface or hyperplane that is maximally far away from any data example. This distance from the decision hyperplane to the closest data point is the margin of the classifier. And it's fully specified by a subset of data which is the set of support vectors (the nearest points to the separator).

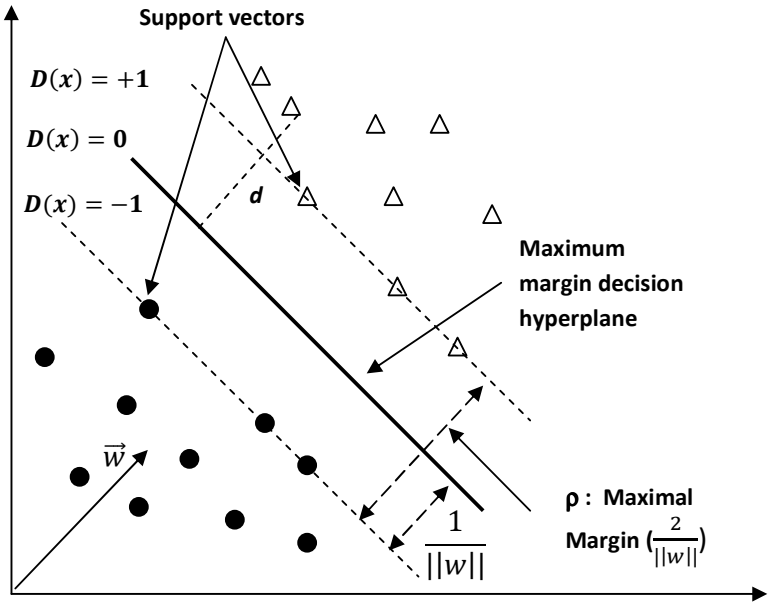


Figure 4.5. Principle of SVM method.

Let us formalize an SVM with algebra. Suppose we have a set of training data points $D = \{(\vec{x}_i, y_i)\}$, where each member of D is a pair of point $\vec{x}_i$ and its class label $y_i \in \{-1, +1\}$.
A decision hyperplane separating the data points according to the two classes $\{-1, +1\}$ can be defined by two parameters: a normal vector $\vec{w}$ also called a weight vector which is perpendicular to the hyperplane, and an intercept term b, where $b/||w||$ is the distance from the hyperplane to the origin, and $||w||$ is the Euclidean norm of $\vec{w}$ ($||w|| = \vec{w}^T.\vec{w}$). this hyperplane verify the following equation:

$$\vec{w}^T.\vec{x} + b = 0 \tag{4.16}$$

And thus, the linear classifier (classification function) is given by:

$$f(\vec{x}) = sign(\vec{w}^T.\vec{x} + b) \tag{4.17}$$

That means, if we want to classify a point $\vec{x}$ we have the following inequation:

$$\begin{cases} if\ \vec{w}^T.\vec{x} + b \geq +1 \Rightarrow y_i = +1 \\ if\ \vec{w}^T.\vec{x} + b \leq -1 \Rightarrow y_i = -1 \end{cases} \tag{4.18}$$

Where $\vec{w}^T.\vec{x}$ is the dot product calculated as follows:

$$\vec{w}^T.\vec{x} = \sum_{j=1}^{n} w_j.x_j \tag{4.19}$$

We can also calculate the distance between any point x_i and the separating hyperplane as follows:

$$d(x_i) = \frac{|\vec{w}^T.\vec{x_i} + b|}{||\vec{w}||} \tag{4.20}$$

And consequently, we have

$$\begin{cases} d(x_i) > +1 \Rightarrow x_i \in \{+1\} \\ d(x_i) < -1 \Rightarrow x_i \in \{-1\} \\ d(x_i) = +1 \Rightarrow x_i \; belongs \; to \; support \; vector \; line \; of \; c1\{+1\} \\ d(x_i) = -1 \Rightarrow x_i belongs \; to \; support \; vector \; line \; of \; c2\{-1\} \\ d(x_i) = 0 \Rightarrow x_i belongs \; to \; the \; separating \; hyperplane \end{cases} \tag{4.21}$$

We recall that the goal of SVMs is to find an optimal hyperplane whose distance from all examples is maximal. Why? Suppose that we have an example which is not well described. In this case, a small variation in the position of the example doesn't update its classification, if its distance to the hyperplane is large.

Thus, we want to find $\vec{w}$ and b such that:

$$\begin{cases} d = \frac{2}{||\vec{w}||} \; is \; maximized \\ \qquad\qquad and \\ for \; all \; (\vec{x_i}, y_i) \in D, y_i.(\vec{w}^T.\vec{x_i} + b) \geq 1 \end{cases} \tag{4.22}$$

Maximizing $\frac{2}{||\vec{w}||}$ is the same as minimizing $\left|\left|\vec{w}\right|\right|/2$ or $1/2\left|\left|\vec{w}\right|\right|^2$

So, find $\vec{w}$ and b such that:

$$\begin{cases} 1/2\vec{w}^T\vec{w} \; is \; minimized \\ \qquad\qquad and \\ for \; all \; (\vec{x_i}, y_i) \in D, y_i.(\vec{w}^T.\vec{x_i} + b) \geq 1 \end{cases} \tag{4.23}$$

We are now optimizing a quadratic problem QP subject to linear constraints which is a standard, well-known class of mathematical optimization problems, and many algorithms exist to solve them.

The solution involves constructing a dual problem where a Lagrange multiplier α_i is associated with each example x_i. Where x_i is a support vector point ($x_i \in$ SV=$\{ x_i \}$ $i=1, 2, \ldots, n_s$ $ns \leq n$) which are the nearest points for the hyperplane.

The problem becomes, find $\alpha_1, \alpha_2, \ldots, \alpha_{ns}$ such that:

$$\begin{cases} \alpha_i - 1/2 \sum_i^{ns} \sum_j^{ns} \alpha_i \alpha_j y_i y_j \vec{x_i}^T x_j \; is \; maximized \\ \alpha_i y_i = 0 \\ \alpha_i \geq 0 \; for \; all \; 1 \leq i \leq ns \end{cases} \tag{4.24}$$

The solution is then of the form:

$$\begin{cases} \hat{\vec{w}} = \sum_{i=1}^{ns} \hat{\alpha}_i y_i \vec{x_i} \\ \hat{b} = y_s - \sum_{i=1}^{ns} \hat{\alpha}_i y_i \vec{x_i}^T x_s \\ d(x) = \hat{\vec{w}}^T.x + \hat{b} \end{cases} \tag{4.25}$$

Where:
$\wedge$: Estimated value.
ns: number of support vectors.
(x_s , y_s): a support vector point.
α_i: A Lagrange multiplier associated to the support vector point x_i.

Most of the α_i are nonzero. Each nonzero α_i indicates a support vector point x_i.
Consequently, the classification function becomes:

$$f(x) = sign(\sum_{i=1}^{ns} \hat{\alpha}_i y_i \vec{x_i}^T x + \hat{b}) \quad (4.26)$$

So, to classify a new example *z*, we must calculate:

$$d(z) = \vec{w}^T . \vec{z} + b = \sum_{i=1}^{ns} \hat{\alpha}_i y_i \vec{x_i}^T z + b \quad (4.27)$$

Don't forget that:

$$\begin{cases} if \;\; d(z) > 0 \Rightarrow z \in \{+1\} \\ if \;\; d(z) < 0 \Rightarrow z \in \{-1\} \end{cases} \quad (4.28)$$

To summarize, we begin with a training data set. This data set helps to define the best separating hyperplane, and we feed data through a quadratic optimization problem to find this hyperplane.
Given a new point $\vec{x}$ to classify, the classification function $f(\vec{x})$ in (4.17) or (4.18) is used to determine the class to assign to the point.

Example 4.5: How to build an SVM classifier

Let the data set represented geometrically as follows:

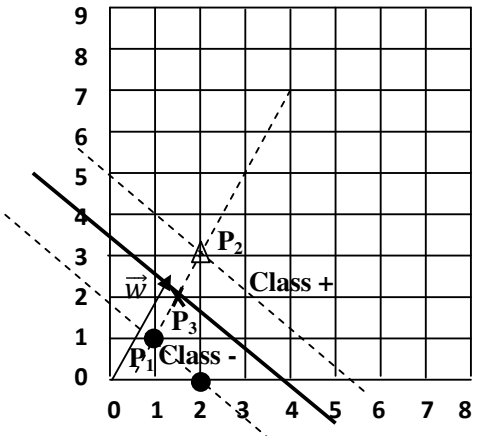


Figure 4.6. Building an SVM for a small data set

How to build an SVM classifier over this data set?
For an example with a very small data set like this, we can Work geometrically to find the hyperplane separating the two classes of points.
Such hyperplane is defined with 2 parameters:
The weight vector $\vec{w}$ and the intercept *b*

$\vec{w}$ will be perpendicular to the hyperplane, and so, parallel to the shortest line connecting points of the two classes.
It is the line between $p_1(1, 1)$ and $p_2(2,3)$ giving a weight Vector $\vec{w}$ *(1, 2)*

The optimal decision surface (hyperplane) is orthogonal to This line, and intersects it at the halfway point $p_3(1.5, 2)$
So, the SVM decision boundary is:

$$y = x_1 + 2x_2 - 5.5 \quad (4.29)$$

We can also work algebraically, with the standard constraint that:

$$sign[y_i.(\vec{w}^T.\vec{x_i} + b)] \geq 1 \quad (4.30)$$

We want to minimize $\|\vec{w}\|$, that happens when the constraint is satisfied with equality by the two support vectors points $p_1(1,1)$ and $p_2(2,3)$.
Further, we know that the solution is $\vec{w}$ $(a, 2a)$ for some a

So, we have that:

$$\begin{cases} a + 2a = -1 \ \ for\ SV_1\ (1,1) \\ 2a + 6a = +1 \ \ for\ SV_2\ (2,3) \end{cases} \quad (4.31)$$

By solving (4.31), we obtain: $a=2/5;$ $b=-11/5$
Thus, the optimal hyperplane is given by $\vec{w}$ $(2/5, 4/5)$ and $b=-11/5$
The margin $\rho = \frac{2}{||\vec{w}||} = 2/\sqrt{(2/5)^2 + (4/5)^2} = 2/\sqrt{20/25} = 2/[2\sqrt{5}/5] = 5/\sqrt{5} = \sqrt{5} \approx 2.236$
All the obtained results can be confirmed geometrically.

Let us see where we can classify the two points $z_1(3, 2)$, $z_2(1, 1/2)$ applying the decision function $sign[y_i.(\vec{w}^T.\vec{x_i} + b)]$.

$$\begin{cases} \left(\frac{2}{5},\frac{4}{5}\right) * (3,2) - \frac{11}{5} = \left(\frac{6}{5} + \frac{8}{5}\right) - \frac{11}{5} = \frac{3}{5} > 0 \Rightarrow z_1(3,2) \in \{+1\} \\ \left(\frac{2}{5},\frac{4}{5}\right) * (1,1/2) - \frac{11}{5} = \left(\frac{2}{5} + \frac{2}{5}\right) - \frac{11}{5} = -\frac{7}{5} < 0 \Rightarrow z_2(1,1/2) \in \{-1\} \end{cases} \quad (4.32)$$

4.3.4.2. Nonlinearly separable case and noisy data

a) Large margin classification for noisy data

Sometimes the data are linearly separable. But in the general case they are not, and even if they are linearly separable, we might prefer finding a good solution that better separates data while ignoring a few weird noise documents (examples inside or on the wrong side of margin). To implement this, we introduce slack variables ξ_i (error variables). And the formulation of the SVM optimization problem with slack variables becomes: [Cristianini and Shawe-Taylor, 2000, Schuolkopf and Smola, 2001]

Find $\vec{w}$ and b and $\xi_i \geq 0$ such that:

$$\begin{cases} \frac{1}{2\vec{w}^T\vec{w}} + C\sum_i \xi_i \ \ is\ minimized \\ \qquad\qquad and \\ for\ all\ (\vec{x_i}, y_i) \in D, y_i.(\vec{w}^T.\vec{x_i} + b) \geq 1 - \xi_i \end{cases} \quad (4.33)$$

Where:
C: a constant that limits the margin failure.
ξ_i: slack variable that allows a margin failure.

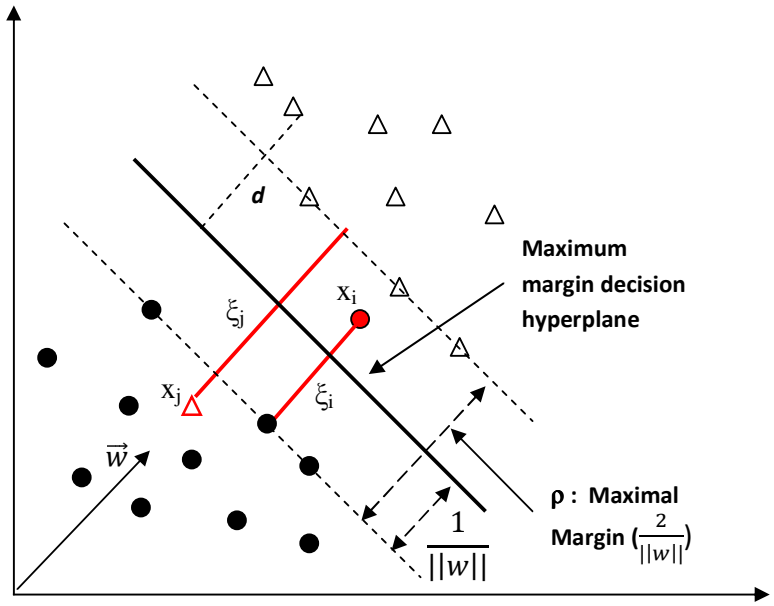


Figure 4.7 Large margin classification with slack variables.

As seen before, we have:
If $\xi_i = 0$ ➔ no error on points x_i
Else:

$$\begin{cases} if\ \ \vec{w}^T.\vec{x} + b \geq 1 - \xi i \Rightarrow y_i = +1 \\ if\ \ \vec{w}^T.\vec{x} + b \leq -1 + \xi i \Rightarrow y_i = -1 \\ \xi i \geq 0 \end{cases} \quad (4.34)$$

The dual problem becomes
Find $\alpha_1, \alpha_2, \ldots, \alpha_{ns}$ such that:

$$\begin{cases} \sum \alpha_i - 1/2 \sum_i \sum_j \alpha_i \alpha_j y_i y_j \vec{x}_i^{\,T} x_j \ is\ maximized\ and \\ \sum_i \alpha_i y_i = 0; 0 \leq \alpha_i \leq C \quad for\ all\ \ 1 \leq i \leq ns \end{cases} \quad (4.35)$$

We see that the slack variables ξ_i and their correspondents Lagrange multipliers don't appear in the dual problem. But we use the constant C bounding the possible values of Lagrange multipliers for the support vector data points. We recall also that the x_i with nonzero α_i will be the support vector points. Thus the solution of the dual problem will be of the form:

$$\begin{cases} \vec{\hat{w}} = \sum_{i=1}^{ns} \hat{\alpha}_i y_i \vec{x_i} \\ \hat{b} = y_k(1 - \xi_k) - \ \vec{w}^T \vec{x}_k \quad for\ k = \ argmax_k \ \alpha_k \\ d(x) = \vec{\hat{w}}^T.x_k + \hat{b} \end{cases} \quad (4.36)$$

b) Multiclass SVM

SVMs are two-class classifiers. Therefore, to do multiclass classification with SVMs we proceed to one of the following methods: [Crammer and Singer, 2001, Tsochantaridis et al., 2005]

1. ***One-Versus-All (One-Against-All/OVA) technique:*** the most common technique in practice and consists in building $|C|$ one-versus-rest classifiers, and to choose the class that classifies the test datum with greatest margin.
2. ***One-Versus-One (One-Against-One/OVO) technique:*** consists in building a set of one-versus-one classifiers, and to choose the class that is selected by the most classifiers. This involves building $|C|(|C| - 1)/2$ classifiers, but the time for training classifiers may decrease, because the training data set for each classifier is much smaller.

Example 4.6: One-Versus-One Technique

Let the classes C_1, C_2, C_3

To build a multiclass SVM, we must transform our problem to a binary SVM classification. Thus, we build 3(3-1)/2 = 3 binary classifiers; $SVM1(C_1, C_2)$, $SVM2(C_1, C_3)$, $SVM3(C_2, C_3)$. And if the obtained results of binary classifiers were: $SVM1(C_1, C_2)=C_1$, $SVM2(C_1, C_3)=C_3$, $SVM3(C_2, C_3)=C_3$. To decide to which class our point will belong, we use the majority voting (e.g., we choose the class which receive more binary decisions as the correct class of the multiclass decision). So the multiclass decision in this case is C_3

c) Nonlinear SVM

We have presented so far how we build SVMs for data sets that are linearly separable (even with a few exceptions or some noise). But what happen if the given data set doesn't allow classification using a linear classifier? One way to solve this problem is to map the given data onto a higher dimensional space and then to use a linear classifier in the new higher dimensional space. Of course, we must preserve relevant dimensions of relatedness between data points, so that the resultant classifier should still generalize well. [Pilaszy, 2005, Chen et al., 2005, Joachims, 2006a, Joachims, 2006b]

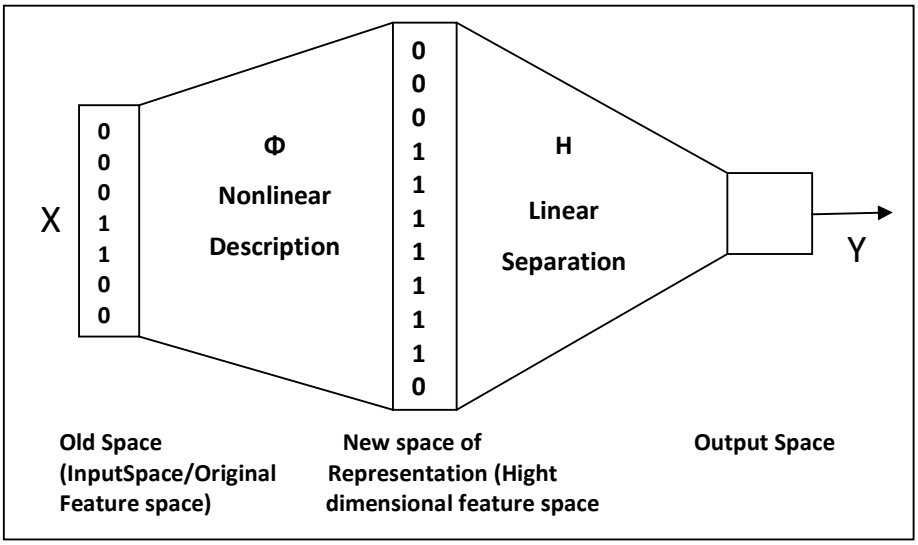


Figure 4.8. How to formulate a nonlinear SVM problem

SVMs, and also other linear classifiers, provide an easy technique to establish this mapping to a higher dimensional space. This technique is called kernel trick. [Aizerman et al., 1964, Cristianini and Shawe-Taylor, 2000, Shawe-Taylor and Cristianini, 2004]

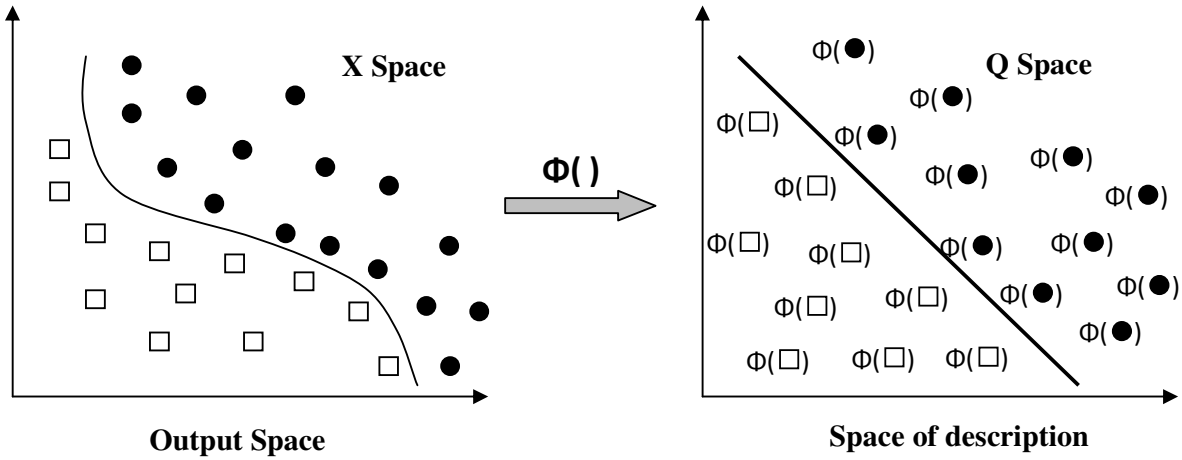


Figure 4.9. Using a kernel trick to map nonlinearly separable data

Let $(\vec{x}_i, \vec{x}_j)$) = $\vec{x}_i^{\,T} x_j$ (dot product)

So, $$f(\vec{x}) = sign(\sum_{i=1}^{ns} \alpha_i y_i \, k(\vec{x_i}, \vec{x}) + b) \quad (4.37)$$

To map every data point into a high dimensional space via a transformation function Φ such that:

$$\Phi : \vec{x} \longrightarrow \Phi(\vec{x}) \quad (4.38)$$

Then the dot product becomes:

$$\Phi(\vec{x_i})^T . \Phi(\vec{x_j}) \quad (4.39)$$

We could simply compute the quantity:

$$k(\vec{x_i}, \vec{x}) = \Phi(\vec{x_i})^T . \Phi(\vec{x_j}) \quad (4.40)$$

And then use the function's value in equation (4.37)
A kernel function is a function that corresponds to a dot product in some expanded feature spaces.

Example 4.7: Definition of a kernel function
Let the function $k(\vec{u}, \vec{v}) = (1 + \vec{u}^T v)^2$, with $\vec{u} = (u_1, u_2)$, $\vec{v} = (v_1, v_2)$

We want to demonstrate that $k(\vec{u}, \vec{v})$ is a kernel function. i.e., $k(\vec{u}, \vec{v}) = \Phi(\vec{u})^T . \Phi(\vec{v})$ for some Φ
Φ is a function of transformation defined as follows:

$\Phi(\vec{u}) = \Phi(u_1, u_2) = (1 \quad u_1^{\,2} \quad \sqrt{2}u_1u_2 \quad u_2^{\,2} \quad \sqrt{2}u_1 \quad \sqrt{2}u_2)$
$\Phi(\vec{v}) = \Phi(v_1, v_2) = (1 \quad v_1^{\,2} \quad \sqrt{2}v_1v_2 \quad v_2^{\,2} \quad \sqrt{2}v_1 \quad \sqrt{2}v_2)$

We have on one hand $k(\vec{u}, \vec{v}) = (1 + \vec{u}^T v)^2 = (1 + u_1v_1 + u_2v_2)^2$

On the other hand, we have :

$\Phi(\vec{u})^T.\Phi(\vec{v})= \Phi(u_1,u_2).\ \Phi(v_1,v_2) = (1\ \ {u_1}^2\ \ \sqrt{2}u_1u_2\ \ {u_2}^2\ \ \sqrt{2}u_1\ \ \sqrt{2}u_2)^{\mathrm{T}}.(1\ \ {v_1}^2\ \ \sqrt{2}v_1v_2\ \ {v_2}^2\ \ \sqrt{2}v_1\ \ \sqrt{2}v_2)$ (dot product)

$= (1 + u_1v_1 + u_2v_2)^2$

$= k(\vec{u},\vec{v})$

Thus, $k(\vec{u},\vec{v})$ is a kernel function.

d) What kinds of functions are valid kernel functions?

1. A kernel function must satisfy Mercer's condition: for any function g($\vec{x}$), such that $\int g(x)^2\ \mathrm{d}\vec{x}$ is finite, we must have $\int g(\vec{x},\vec{y})\ \mathrm{g}(\vec{x})\mathrm{g}(\vec{y})\mathrm{d}\vec{x}\mathrm{d}\vec{y} \geq 0$
2. A kernel function must be:
   - Symmetric
   - Continuous
   - Have a positive definite matrix
3. If a kernel function doesn't satisfy Mercer's condition, than the corresponding QP may have no solution.

e) The new optimizing problem

The dual problem when using a kernel function becomes:

$$\begin{cases} Max\ \sum \alpha_i - \ 1/2 \sum_i \sum_j \alpha_i\alpha_j y_i y_j \Phi(\vec{x_i})^T.\Phi(\vec{x_j}) = \\ \sum_i \alpha_i y_i = 0; \\ 0 \leq \alpha_i \leq C \quad for\ all\ \ 1 \leq i \leq ns \end{cases} \tag{4.41}$$

We must satisfying the Karush-Kuhn-Tucker KKT condition

$$\begin{cases} \alpha_i = 0\ \Rightarrow y_i.f(\vec{x}_j) \geq 1 \\ 0 \leq \alpha_i \leq C\ \Rightarrow\ \ y_i.f(\vec{x}_j) = 1 \\ \alpha_i = C\ \ \Rightarrow\ y_i.f(\vec{x}_j) \leq 1 \end{cases} \tag{4.42}$$

The decision function is given by:

$$f(x) = sign(\textstyle\sum_{i=1}^{ns} \hat{\alpha}_i y_i \Phi(\overrightarrow{x_i})^T.\Phi(\vec{x}) + \hat{b}) \tag{4.43}$$

The solution is:

$$\begin{cases} \vec{\hat{w}} = \sum_{i=1}^{ns} \hat{\alpha}_i y_i \Phi(\overrightarrow{x_i}) \\ \hat{b} = y_s - \ \sum_{i=1}^{ns} \hat{\alpha}_i y_i\ \Phi(\overrightarrow{x_i})^T x_s \\ d(x) = \vec{\hat{w}}^T.\Phi(\vec{x}) + \hat{b} \end{cases} \tag{4.44}$$

f) Examples of kernel functions

1. ***Polynomial kernels:*** are of the form:

$$k(\vec{x},\vec{y}) = (1 + \vec{x}^T y)^d. \quad (4.45)$$

The case *d=1* gives the linear kernel, the case *d=2* gives a quadratic kernel which is very used.

2. ***The radial basis kernel function:*** is a Gaussian distribution, calculated as:

$$k(\vec{x},\vec{y}) = e^{-(\vec{x}-\vec{y})^2/(2\sigma^2)} \quad (4.46)$$

3. ***The sigmoid kernel with parameters k, θ***

$$k(\vec{x},\vec{y}) = Tangh(k + \vec{x}^T y) \quad (4.47)$$

Kernel We note that the sigmoid doesn't satisfy the condition of Mercer ∀ k, θ

4. ***The Laplace kernel***

$$k(\vec{x},\vec{y}) = e^{-(\vec{x}-\vec{y})/(1/\sigma)} \quad (4.48)$$

5. ***Other kernels***

Example 4.8: Some kernel functions

Let the following data training set: 47 data points (22+, 25-). This data set can be represented by two kernel functions: polynomial kernel (with degree *d=2, 5, 8* and *C=10000*) and sigmoid kernel (with σ*=2, 5, 10, 20* and *C=10000*). We note that there are some critical points (4+, 3-).

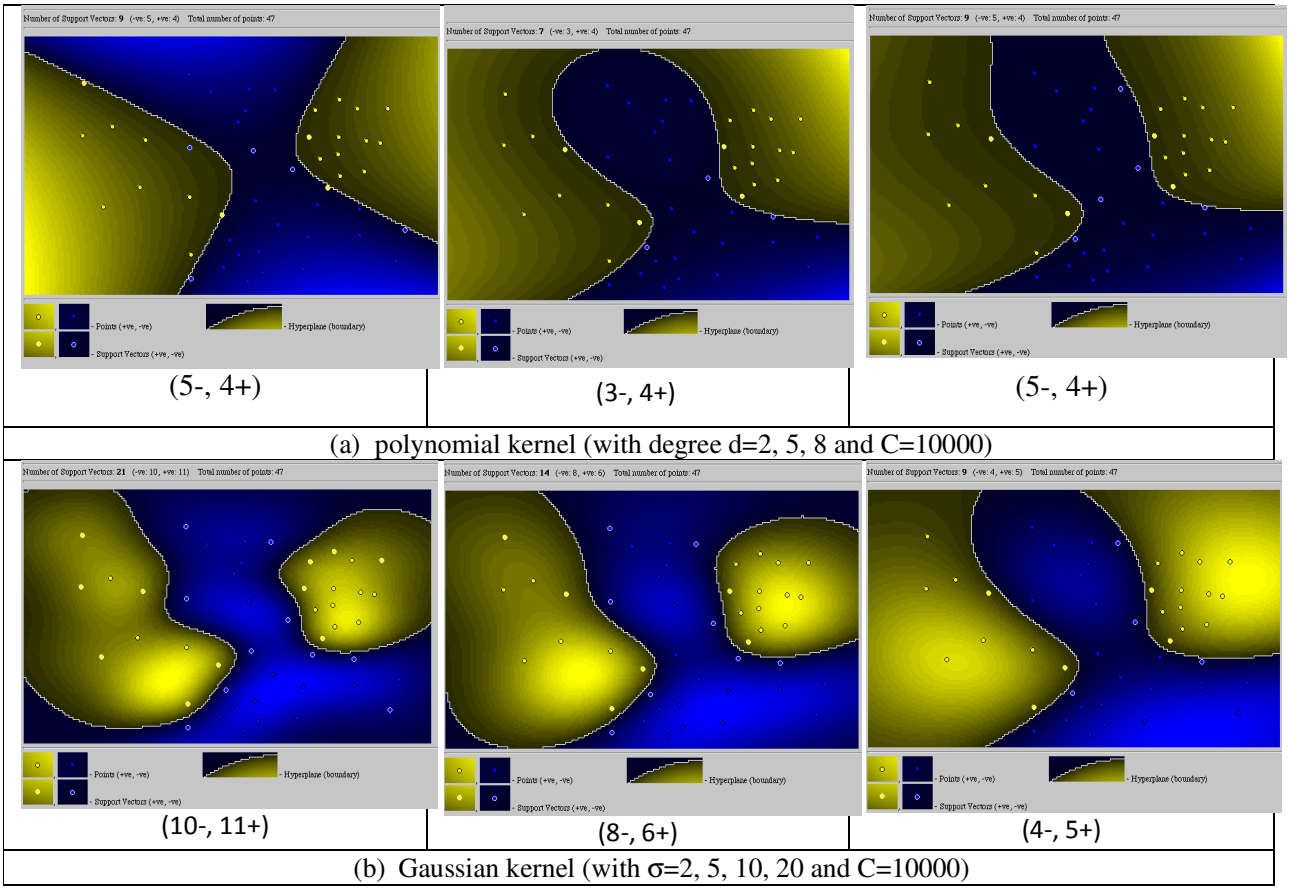


Figure 4.10. Examples of kernel functions

4.3.4.3. Software Implementations

1. ***LibSVM [16]:*** is one of the most famous software about the implementation of SVM algorithms. LibSVM provides not only compiler languages used in the Windows system, but also C++ and Java source codes which are easy to improve, revise, and apply in other operating systems. Specially, LibSVM has relatively fewer tunable parameters involved in SVM algorithms than other software and provides lots of default parameters to solve real application problems effectively. [Chang and Lin, 2011] (https://www.csie.ntu.edu.tw/~cjlin/libsvm/)

2. ***SVMlight [17]:*** is another implementation in C language. It adopts an efficient set selection technique based on steepest feasible descent, and two effective computational policies "Shrinking" and "Caching" of kernel evaluations. SVMlight mainly includes two C programs: SVM learn, used for learning training samples and training the corresponding classifier, and SVM classifiy, used for classifying testing samples. The software also provides two efficient estimation methods for assessing the generalization performance: XiAlpha-estimates, computed at essentially no computational expense but conservatively biased, and Leave-one-out testing, almost unbiased. [Joachims, 1999a, Joachims, 1999c, Joachims, 2002a, Joachims, 2002b, Joachims, 2002c]. (http://svmlight.joachims.org/).

Furthermore, there are lots of complete machine learning toolboxes, including SVM algorithms, such as Torch (in C++), Spider (in MATLAB), and Weka (in Java), which are all available at http://www.kernel-machines.org.

4.3.5. Decision Tree classifiers.

4.3.5.1. Algorithm presentation

The majority of classifiers seen previously have a common problem that they cannot be easily understood by humans. The symbolic classifiers such as the decision tree classifier do not suffer from this problem.

A decision tree (DT) classifier is a tree in which the internal nodes are labeled by the features (words or terms in the case of text categorization), the edges leaving a node are labeled by tests on the feature's value, and the leaves are labeled by categories. [Quilan, 1986, Quilan, 1987, Quilan, 1990, Quilan, 1993 Fuhr et al, 1991, Quilan, 1996, Lewis and Ringuette, 1994, Apté et al., 1998].

A DT classifier categorizes a document by starting at the root of the tree and moving successively downward via the branches whose conditions are satisfied by the document until a leaf node is reached. The document is then assigned to the category that labels the leaf node.

Most of the DT classifiers use a binary document representation, and thus the trees are binary. Most of the DT-based systems use some form of general procedure for a DT induction such as ID3, C4.5, and CART. Typically, the tree is built recursively by picking a feature *f* at each step and dividing the training collection into two sub-collections, one containing *f* and another not containing *f*, until only documents of a single category remain – at which point a leaf node is generated. The choice of a feature at each step is made by some information-theoretic measure such as: information gain or entropy. However, the trees generated in such a way are prone to overfit the training collection, and so most methods also include pruning – that is, removing the too specific branches.

Original applications of decision trees were in domains with nominal valued or categorical data but today they span a multitude of domains with numeric, symbolic, and mixed-type attributes. Examples include clinical decision making, manufacturing, document analysis, bioinformatics, spatial data modeling (geographic information systems), and practically any domain where decision boundaries between classes can be captured in terms of tree-like decompositions or regions identified by rules.

The performance of a DT classifier is mixed but is inferior to the top-ranking classifiers. Thus it is rarely used alone in tasks for which the human understanding of the classifier is not essential. DT classifiers, however, are often used as a baseline for comparison with other classifiers and as members of classifier committees.

Example 4.9: How to build a decision tree

We have the dataset shown in table 4.3.

Table 4.3. Dataset used to create a Decision Tree

Day	Outlook	Temperature	Humidity	Windy	Play Golf?
1	Sunny	85	85	False	No
2	Sunny	80	90	True	No
3	Overcast	83	78	False	Yes
4	Rainy	70	96	False	Yes
5	Rainy	68	80	False	Yes
6	Rainy	65	70	True	No
7	Overcast	64	65	True	Yes
8	Sunny	72	95	False	No
9	Sunny	69	70	False	Yes
10	Rainy	75	80	False	Yes
11	Sunny	75	70	True	Yes
12	Overcast	72	90	True	Yes
13	Overcast	81	75	False	Yes
14	rainy	71	80	True	No

Where rows denote specific days, attributes denote weather conditions on the given day, and the class denotes whether the conditions are conducive to playing golf or no (the last column). Thus, each row denotes an instance of the dataset, described by many attributes such as: Outlook (valued random variable), Temperature (continuous-valued), Humidity (continuous-valued), Windy (binary), and the class is the Boolean variable PlayGolf?. All of the data in Table 4.3 form the "training data," our goal is to build a model using this dataset and apply it on other new instances to predict the value for the class variable. [Xu and Kumar, 2009]
Figure 4.11 illustrates the induced tree using Table 4.3 as training data (and the default options).

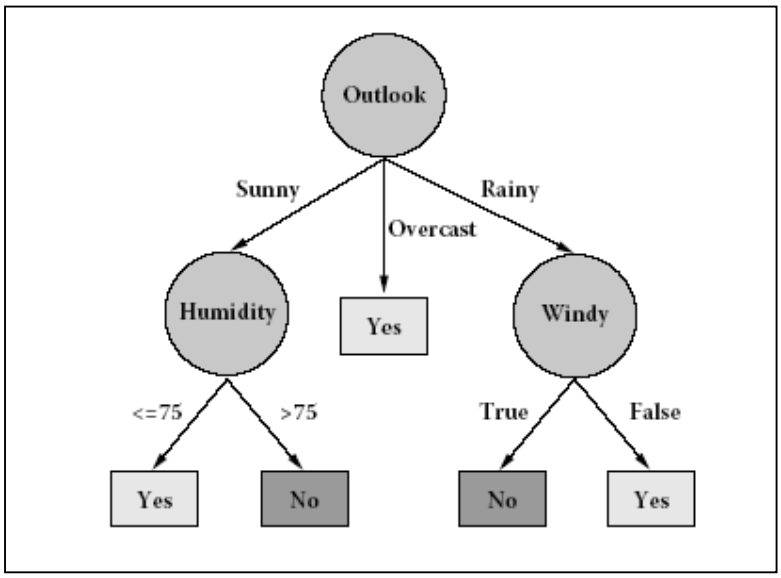


Figure 4.11 Decision tree induced from the dataset of Table 4.3.

A new instance will be classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the sub-tree rooted at the new node.
For example, the decision tree built in figure 4.11 classifies Friday afternoon according to whether they are suitable for playing golf. This instance is described as follows:

$$Friday <Outlook = Sunny, Temperature = 83, Humidity = 90, Windy = True>$$

It would be sorted down the leftmost branch of this decision tree and would therefore be classified as a negative instance (i.e., the tree predicts that *PlayGolf =No*).
In general, decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances. For example, the decision tree shown in Figure 4.11 corresponds to the expression:

$$(Outlook = Sunny \wedge Humidity \leq 75)$$
$$\vee (Outlook = Overcast)$$
$$\vee (Outlook = Rain \wedge Windy = False)$$

4.3.5.2. When we use decision tree learning?

Decision tree learning is generally best suited to problems with the following characteristics:

1. Instances are represented by attribute-value pairs. i.e., they are described by a fixed set of attributes (e.g., *Outlook, Temperature, Humidity,…*) and their values (e.g., *Sunny, Hot,…*). The easiest situation for decision tree learning is when each attribute takes on a small number of disjoint possible values (e.g., *Hot, Mild, Cold*). However, extensions to the basic algorithm allow handling real-valued attributes as well (e.g., representing Temperature numerically).
2. The target function has discrete output values (e.g., boolean values: *Yes* or *No* for the target attribute *PlayGolf*). Decision tree methods easily extend to learning functions with more than two possible output values and even real-valued outputs
3. Disjunctive descriptions may be required. As noted above, decision trees naturally represent disjunctive expressions.
4. The training data may contain errors. Decision tree learning methods are robust to errors, both errors in classifications of the training examples and errors in the attribute values that describe these examples.
5. The training data may contain missing attribute values. Decision tree methods can be used even when some training examples have unknown values (e.g., if the Humidity of the day is known for only some of the training examples).

Most algorithms that have been developed for learning decision trees are variations on the same core algorithm that employs a top-down, greedy search through the space of possible decision trees. Among these algorithms, we find the ID3 algorithm [Quinlan, 1986] and its successor C4.5 [Quinlan 1993]. ID3 is the basic algorithm for decision tree learning. But there is also a number of extensions to this basic algorithm, including extensions incorporated into C4.5 and other more recent algorithms.

4.3.5.3.ID3 algorithm

The basic algorithm for decision tree learning, ID3, learns decision trees by constructing them top-down, beginning with the question "*which attribute should be tested at the root of the tree?*". To answer this question, each instance attribute is evaluated using a statistical test (mutual information, information gain, etc) to determine how well it alone classifies the training examples. The best attribute is selected and used as the test at the root node of the tree. A descendant of the root node is then created for each possible value of this attribute, and the training examples are divided and sorted to the appropriate descendant. This process is then repeated using the training examples associated with each descendant node to select the best attribute to test in the following step (at that point in the tree). This builds an acceptable decision tree, in which the algorithm never backtracks to reconsider earlier choices. [Mitchell, 1997]

a) Which Attribute Is the Best Classifier?

The central choice in the ID3 algorithm is selecting which attribute to test at each node in the tree. We would like to select the attribute that is most useful for classifying examples. So, *what is a good quantitative measure of the worth of an attribute?* We will define a statistical property, called information gain, that measures how well a given attribute separates the training examples according to their target classification. ID3 uses this information gain measure to select among the candidate attributes at each step while growing the tree.

1. ***The entropy measure E:*** In order to define information gain precisely, we begin by defining a measure commonly used in information theory, called entropy, that characterizes the (im)purity of an arbitrary collection of examples. Given a collection *S*, containing positive and negative examples of some target concept, the entropy of *S* relative to this boolean classification is:

$$Entropy(S) = -p_+ log_2 p_+ - p_- log_2 p_- \quad (4.49)$$

Where:
P_+: is the proportion of positive examples in *S*.
p_-: is the proportion of negative examples in *S*.
We define 0 log 0 to be 0.

Example 4.10: Choice of the best classifier attribute

Let *S* a collection of 14 examples of some Boolean concept, including 9 positive and 5 negative examples. We use the notation *S* = [9+, 5-] to summarize such a sample of data. What is the entropy of *S* relative to this boolean classification?

Entropy([9+, 5-]) = -(9/14)*Log_2(9/14) – (5/14)*Log_2(5/14)
= - (0.642857)*[ln(0.642857)/ln(2)] – (0.357142857)*[ln(0.357142857)/ln(2)]
= - (0.642857)*[-0.441832/0.693147] – (0.357142857)*[- 1.02961942/0.693147]
= (- 0.642857)* (– 0.637429) - (0.357142857)*[- 1.4854271818]
= + 0.409775695 + 0.5305097
≈ 0.940

Remarks

- The entropy is 0 if all members of *S* belong to the same class.
- The entropy is 1 when the collection contains an equal number of positive and negative examples.
- If the collection contains unequal numbers of positive and negative examples, the entropy is between **0** and 1.

In equation (4.49), we have discussed entropy in the special case where the target classification is boolean. More generally, if the target attribute can take on *c* different values, then the entropy of *S* relative to this c-wise classification is defined as:

$$Entropy(S) = \sum_{i=1}^{c}(- p_i log_2 p_i) \quad (4.50)$$

Where:
p_i : is the proportion of *S* belonging to class *i*.

2. ***Information gain measures IG:*** Given entropy as a measure of the impurity in a collection of training examples, we can define a new measure of the effectiveness of an attribute in classifying the training data. This measure is called information gain. It is simply the expected reduction in entropy caused by partitioning the examples according to this attribute. More precisely, the information gain, *Gain(S, A)* of an attribute *A*, relative to a collection of examples *S*, is defined as:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)}^{nb.v} \frac{|S_v|}{|S|} Entropy(S_v) \ ) \quad (4.51)$$

Where:
S: the set of examples in input.
A: the used attribute.
Values(A): is the set of all possible values for attribute A.
S_v: is the subset of *S* for which attribute *A* has value *v* (i.e., $S_v = \{s \in S / A(s) = v\}$).
Entropy(s): is the entropy of the original collection *S*.
$Entropy(S_v)$: is the expected value of the entropy after *S* is partitioned using attribute A. The expected entropy described by this second term is simply the sum of the entropies of each subset S_v weighted by the fraction of examples $\frac{|S_v|}{|S|}$ that belong to S_v.
Gain(S, A) is therefore the expected reduction in entropy caused by knowing the value of attribute A. Or, Gain(S, A) is the information provided about the target function value, given the value of some other attribute A.

Example 4.11: The gain information of an attribute.

Suppose *S* is a collection of training-example days described by attributes, *Wind* is one of these attributes, having two values *False* or *True*.
As it was seen before, *S* is a collection containing 14 examples, [9+, 5-]. Among these 14 examples, suppose 6 of the positive and 2 of the negative examples have *Wind = False*, and the remainder have *Wind = True*.
The information gain due to sorting the original 14 examples by the attribute *Wind* may then be calculated as:
S=[9+, 5-]; A=Wind; Values(A=Windy)={False, True}; S_{False} *=[6+, 2-];* S_{True}*=[3+, 3-]*

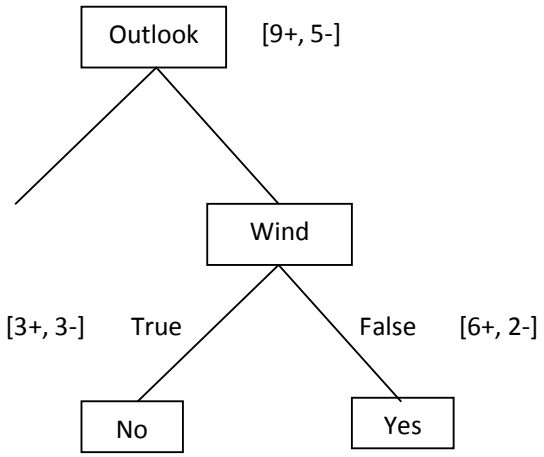


Figure 4.12. Calculation of the gain Information of *Windy* attribute.

⇨ $Gain(S, A=Wind) = Entropy(S) - \sum_{v \in \{False,\ True\}} \frac{|S_v|}{|S|} Entropy(S_v)$

$= Entropy(S) - [\frac{|S_{v=False}|}{|S|} Entropy(S_{v=False}) + \frac{|S_{v=True}|}{|S|} Entropy(S_{v=True})]$

We calculate first:

$Entropy(S_{v=False}) = -p_+ log_2 p_+ - p_- log_2 p_-$ = - 6/8 * [ln(6/8)/ln(2)] – 2/8*[ln(2/8)/ln(2)]

= -6/8*(-0.4150) – 2/8*(-2)= +0.31125+0.5 = 0.81125

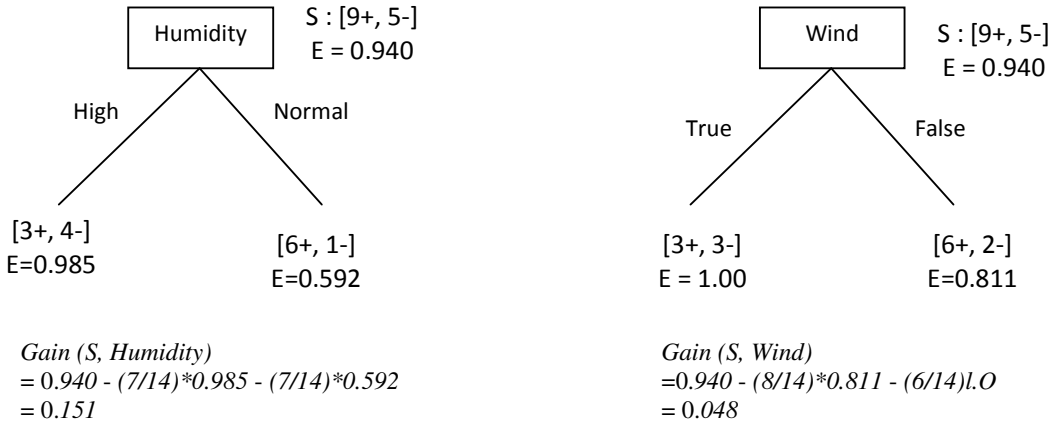
$Entropy(S_{v=True}) = -p_+ log_2 p_+ - p_- log_2 p_-$ = -3/6*[ln(3/6)/ln(2)] – 3/6*[ln(3/6)/ln(2)]

= -0.5*(-1) – 0.5*(-1) = +0.5+0.5 = 1.00

⇨ *Gain(S, A=Wind) =0.940 – [(8/14*0.81125) + (6/14*1.00)]*

=0.940 – [0.46357 + 0.42857] = 0.940 – 0.89214 = 0.04786 ≈0.048

Thus, Information gain IG is precisely the measure used by ID3 to select the best attribute at each step in growing the tree. The use of IG to evaluate the relevance of attributes is summarized in Figure 4.13. In this figure the information gain of two different attributes, Humidity and Wind, is computed in order to determine which is the better attribute for classifying the training examples shown in Table 4.4. of example 4.11



Gain (S, Humidity)
*= 0.940 - (7/14)*0.985 - (7/14)*0.592*
= 0.151

Gain (S, Wind)
*=0.940 - (8/14)*0.811 - (6/14)l.O*
= 0.048

Figure 4.13. Selection of the best attribute using Gain Information (*Humidity* is the best)

Example 4.12: Creation of DT with ID3 algorithm

Consider the learning task represented by the training examples of Table 4.4. The target attribute *PlayTennis*, which can have values (*Yes* or *No*) for different Saturday mornings, is to be predicted based on other attributes of the morning in question.

Table 4.4. Training examples for the target concept ***PlayTennis.***

Day	Outlook	Temperature	Humidity	Windy	PlayTennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	rainy	Mild	High	Strong	No

b) How to create the decision tree using ID3 algorithm? [Mitchell, 1997]

To create the DT using ID3 algorithm, we must know which attribute should be tested first in the tree? For this purpose, we calculate the IG for each candidate attribute (i.e., *Outlook, Temperature, Humidity, Wind*) as it was shown before, then selects the one with highest IG. After calculation, we obtained the following values:

Gain(S, Outlook) = 0.246
Gain(S, Humidity) = 0.151
Gain(S, Wind) = 0.048
Gain(S, Temperature) = 0.029

So, the Outlook is the selected attribute, because it provides the best prediction of the target attribute *PlayTennis*, over the training examples. It's selected as the decision attribute for the root node, and branches are created below the root for each of its possible values (i.e., *Sunny, Overcast, and Rain*). (See Figure 4.13). At this point, the training examples are sorted (partitioned) to each new descendant node. We note also that every example for which *Outlook = Overcast* is a positive example of *PlayTennis.* So, this node of the tree becomes a leaf node with the classification *PlayTennis = Yes*. In contrast, the descendants corresponding to *Outlook = Sunny* and *Outlook = Rain* have nonzero entropy, and the decision tree will be further induced below these nodes.

The process of selecting a new attribute and partitioning the training examples is now repeated for each nonterminal descendant node, but, this time using only the training examples associated with that node. Attributes that have been incorporated higher in the tree are excluded in the current step, so that any given attribute can appear at most once along any path through the tree. This process continues for each new leaf node until either of two conditions is met:

(1) Every attribute has already been included along this path through the tree, or

(2) The training examples associated with this leaf node all have the same target attribute value (i.e., their entropy is zero). Figure 4.14 illustrates the computations of information gain for the next step in growing the decision tree. The final decision tree learned by ID3 from the 14 training examples of Table 4.4 is shown in Figure 4.15

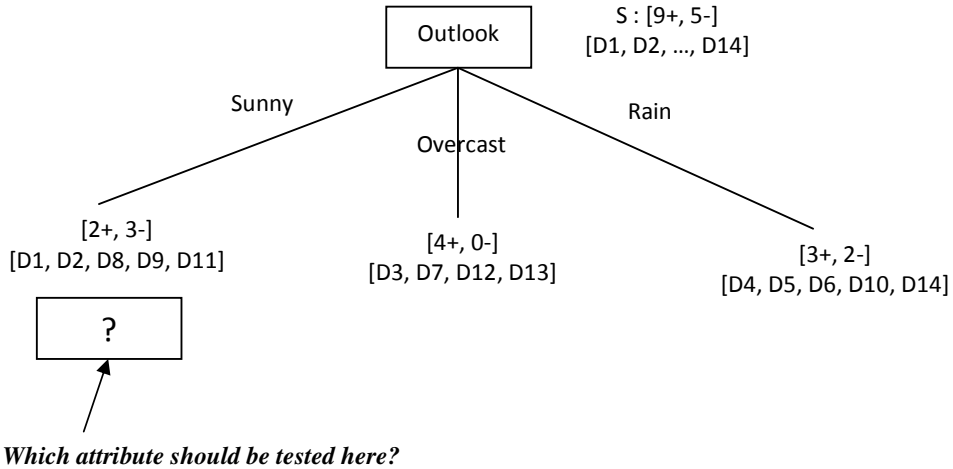


Figure 4.14. The partially learned decision tree resulting from the first step of ID3.

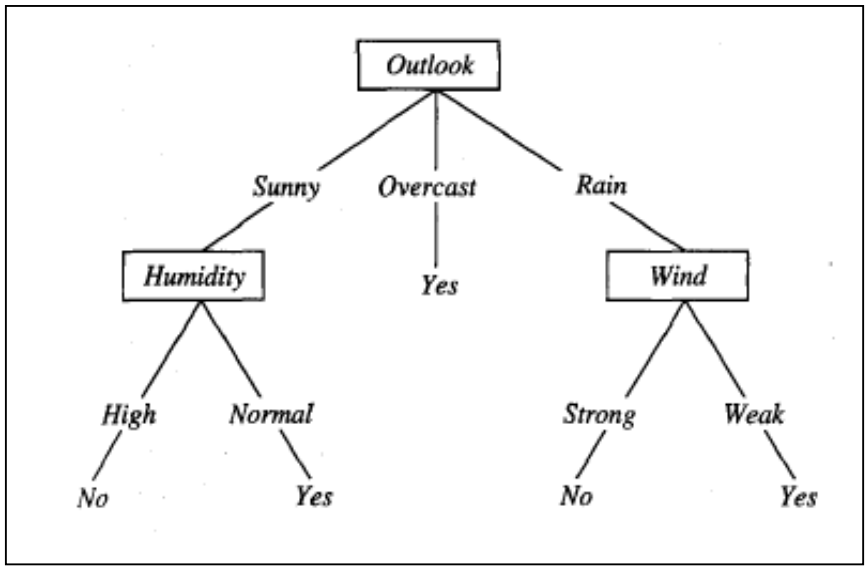


Figure 4.15. A full decision tree for the target attribute *PlayTennis induced using ID3 algorithm.*

ID3(Examples, Targetattribute, Attributes)
//Examples are the training examples.
// Targetattribute is the attribute whose value is to be predicted by the tree.
//Attributes is a list of other attributes that may be tested by the learned decision tree. Returns a
//decision tree that correctly classiJies the given Examples.

Create a ***Root*** node for the tree
I f all ***Examples*** are positive ***Then*** Return the single-node tree ***Root,*** with label = +
I f all ***Examples*** are negative ***Then*** Return the single-node tree ***Root,*** with label = -
If Attributes is empty ***Then*** Return the single-node tree ***Root,*** with label = most common value of ***Targetattribute*** in ***Examples***
Otherwise
Begin
A ← the attribute from ***Attributes*** that best* classifies ***Examples***
//The best attribute is the one with highest information gain
The decision attribute for ***Root*** ← ***A***

For each possible value, v_i, of ***A do***
Add a new tree branch below ***Root,*** corresponding to the test ***A*** = $\boldsymbol{v_i}$
Let $\boldsymbol{Examples_{vi}}$ be the subset of ***Examples*** that have value $\boldsymbol{v_i}$ for ***A***
If $\boldsymbol{Examples_{vi}}$ is empty ***Then***
Below this new branch add a leaf node with label = most common value of ***Target attribute*** in ***Examples***
Else
Below this new branch add the subtree
$\boldsymbol{ID3(Examples_{vi}, Targetattribute, Attributes - \{A\}))}$
End
Return ***Root***

Algorithm 4.6. ID3 algorithm

4.3.5.4. C4.5 algorithm

C4.5 [Quilan, 1996] is a suite of algorithms (C4.5, C4.5-no-pruning, C4.5-rules) for classification problems in machine learning and data mining. It was designed by J. Ross Quinlan, and named like this because it is a descendant of the ID3 approach [Quilan, 1986]. It is considered as a supervised learning method. That means, given an attribute-valued dataset in where instances are described by collections of attributes and belong to one of a set of mutually exclusive classes, C4.5 learns a model (a mapping) from attribute values to classes that can be applied to classify new, unseen instances. Today, C4.5 is superseded by the C5.0 system, a commercial product offered by Rulequest Research, Inc.

Example 4.13: Creation of DT with C 4.5 algorithm

Here, we take the dataset described in Table 4.3, and try to create a DT using C4.5 algorithm.

a) How C4.5 works?

All tree induction methods begin with a root node that represents the entire, given dataset and recursively split the data into smaller subsets by testing for a given attribute at each node. The subtrees denote the partitions of the original dataset that satisfy specified attribute value tests. This process typically continues until the subsets are "pure," that is, all instances in the subset fall in the same class, at which time the tree growing is terminated. The detailed algorithm of C4.5 is given as follows:

Algorithm 1.1 C4.5(D)
Input: an attribute-valued dataset *D*
1: Tree = {}
2: **if** *D* is "pure" OR other stopping criteria met **then**
3: terminate
4: **end if**
5: **for all** attribute $a \in D$ **do**
6: Compute information-theoretic criteria if we split on *a*
7: **end for**
8: a_{best} = Best attribute according to above computed criteria
9: Tree = Create a decision node that tests a_{best} in the root
10: D_v = Induced sub-datasets from *D* based on a_{best}
11: **for all** D_v **do**
12: $Tree_v = C4.5(D_v)$
13: Attach Tree*v* to the corresponding branch of Tree
14: **end for**
15: **return** Tree

Algorithm 4.7. C4.5 algorithm

Figure 4.16 illustrates the tree induced by C4.5

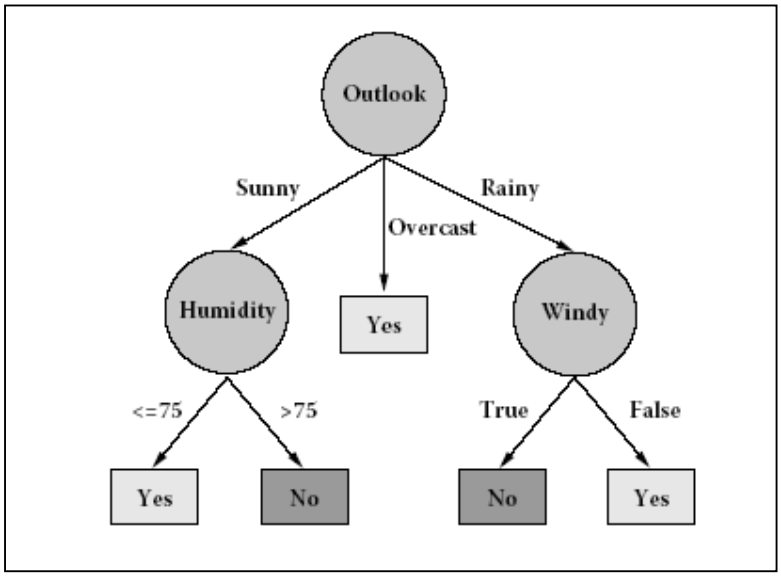


Figure 4.16 Decision tree induced by C4.5 for the dataset of table 4.3.

b) How are tests chosen?

C4.5 uses information-theoretic criteria such as gain (reduction in entropy of the class distribution due to applying a test) and gain ratio (a way to correct for the tendency of gain to favor tests with many outcomes). The default criterion is gain ratio. At each point in the tree-growing, the test with the best criteria is greedily chosen.

c) How is tree-growing terminated?

A branch from a node is declared to lead to a leaf if all instances that are covered by that branch are pure. Another way in which tree-growing is terminated is if the number of instances falls below a specified threshold.

d) How are class labels assigned to the leaves?

The majority class of the instances assigned to the leaf is taken to be the class prediction of that subbranch of the tree.

4.3.6. Decision Rule Classifiers

Decision rule (DR) classifiers are also symbolic method like decision trees. The rules used here are like the disjunctive normal form (DNF) rules but are built from the training collection using inductive rule learning. Typically, the rule learning methods attempt to select the best rule from the set of all possible covering rules (i.e., rules that correctly classify all training examples) according to some optimality criterion. DNF rules are often built in a bottom-up fashion. The initial most specific classifier is built from the training set by viewing each training document as a clause of the form:

$$t_1 \wedge t_2 \wedge \ldots \wedge t_n \rightarrow c \qquad (4.52)$$

Where: t_i: are the features of the document and c its category.

The learner then applies a series of generalizations (e.g., by removing terms from the clauses and thus maximizing the compactness of the rules while keeping the covering property. At the end of the process, a pruning step similar to the DT pruning is applied that trades covering for more generality.
One of the prominent examples of this family of algorithms is RIPPER (Repeated Incremental Pruning to Produce Error Reduction) [Cohen, 1995a; Cohen, 1995b; Cohen and Singer, 1996]. Ripper builds a rule set by first adding new rules until all positive category instances are covered and then adding conditions to the rules until no negative instance is covered. One of the attractive features of RIPPER is its ability to bias the performance toward higher precision or higher recall as determined by the setting of the loss ratio parameter, which measures the relative cost of "false negative" and "false positive" errors.

4.3.7. Regression Methods

Regression is a technique for approximating a real-valued function using the knowledge of its values on a set of points. It can be applied to TC, which is the problem of approximating the category assignment function. For this method to work, the assignment function must be considered a member of a suitable family of continuous real-valued functions. Then the regression techniques can be applied to generate the (real-valued) classifier.
One regression method is the linear least-square fit (LLSF), which was first applied to TC in [Yang and Chute, 1994]. In this method, the category assignment function is viewed as a $|C| \times |F|$ matrix, which describes some linear transformation from the feature space to the space of all possible category assignments. To build a classifier, we create a matrix that best accounts for the training data. The LLSF model computes the matrix by minimizing the error on the training collection according to the formula:

$$M = arg\ min_M ||MD - O||_F \tag{4.53}$$

Where:
D: is the $|F| \times |TrainingCollection|$ matrix of the training document representations.
O: is the $|C| \times |TrainingCollection|$ matrix of the true category assignments.
$\|\cdot\|_F$: is the Frobenius norm.

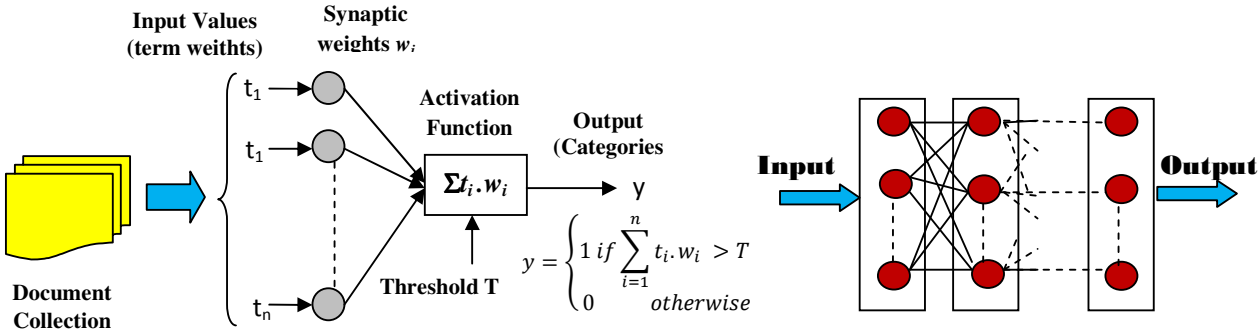
$$||A||_F = \sqrt{\sum A_{ij}{}^2} \tag{4.54}$$

The matrix M can be computed by performing singular value decomposition on the training data. The matrix element m_{ij} represents the degree of association between the i th feature and the j th category.

4.3.8. Artificial Neural Networks

Artificial neural network (ANN) can be built to perform text categorization. Usually, the input nodes of the network receive the feature values, the output nodes produce the categorization status values, and the link weights represent dependence relations. For classifying a document, its feature weights are loaded into the input nodes; the activation of the nodes is propagated forward through the network, and the final values on output nodes determine the categorization decisions.
The neural networks are trained by back propagation, whereby the training documents are loaded into the input nodes. If a misclassification occurs, the error is propagated back through the network, modifying the link weights in order to minimize the error.

The simplest kind of a neural network is a perceptron. It has only two layers – the input and the output nodes. Such network is equivalent to a linear classifier. More complex networks contain one or more hidden layers between the input and output layers. However, the experiments have shown very small – or no – improvement of nonlinear networks over their linear counterparts in the text categorization task [Schutze et al., 1995,;Wiener et al., 1995, Ng et al., 2000].



(a) ANN with threshold T (b) Multilayer ANN

Figure 4.17 Examples of neuronal networks used in text categorization

4.3.9. Classifier Committees: Bagging and Boosting

The idea behind the use of committees of classifiers is issued from the intuition that a team of experts, when combining their knowledge, may produce better results than a single expert alone. In the bagging method of building committees, the individual classifiers are trained in parallel on the same training collection. These classifiers must differ significantly from each other – either in their document representation or in their learning methods. The bagging method is usually chosen in TC.

For instance, if there are *k* different classifiers. To build a single committee classifier, one method consists in combining their results. The simplest way in this case is the *majority vote* in which a category is assigned to a document if at least *(k+1)/2* classifiers classify it in this category (*k* must be an odd number). Another way suited for continuous output, is the *weighted linear combination*, in which the final CSV is given by a weighted sum of the CSVs of the *k* classifiers. The weights can be estimated on a validation dataset. Other ways of combining classifiers are also possible.

Boosting is another method of building classifier committees and improving the quality of categorization by combining several classifiers (decision tree, neuronal network, kNN, other kind of machine learning algorithms). Unlike the bagging method, in boosting the classifiers are trained sequentially (one after the other). Before training the *i*th classifier, the training set is reweighed with greater weight given to the documents that were misclassified (badly classified) by the previous classifiers. The AdaBoost algorithm [Freund and Shapire, 1997, Freund and Shapire, 1999] is the best known example of this approach. It answers to an interesting question that is, whether two complexity classes, *weakly learnable* and *strongly learnable* problems, are equal. If the answer to the question is positive, a weak learner that performs just slightly better than random guess can be "*boosted*" into an arbitrarily accurate strong learner. [Schapire, 1999] found that the answer to the question is "yes," and gave a proof by construction, which is the first boosting algorithm. Freund and Schapire [Freund and Shapire, 1997] then proposed an adaptive boosting algorithm, named AdaBoost, for which they won the Godel Prize. AdaBoost and its variants have been applied to diverse domains with great success, owing to their solid theoretical foundation, accurate prediction, and great simplicity (Schapire said: "*it*

needs only "just 10 lines of code"). It has become a buzzword in computer vision and many other application areas. Formally, It is defined as follows:

Let X be the feature space, and let $D= \{(d_1, c_1), (d_2, c_3), \ldots, (d_m, c_m)\}$ be the training labeled data, where $di \in X$ are the training document representations and $c_i \in \{+1,-1\}$the category assignment (binary). We note also that the label of a test instance is unknown and thus to be predicted. We assume both training and test instances are drawn independently and identically from an underlying distribution D. After training on a training data set D, a learning algorithm L will output a hypothesis h, which is a mapping from X to Y, or called as a classifier. The learning process can be regarded as picking the best hypothesis from a hypothesis space.

A General Boosting Procedure

As we have explained previously, boosting is actually a family of algorithms, among which the AdaBoost algorithm is the most influential one. So, it may be easier to explain the general boosting procedure. [Schapire et al., 1998, Shapire and Singer, 2000, Schapire, 2002, Liu et al, 2002].

Suppose we have a binary classification problem, i.e., we are trying to classify instances as *positive* and *negative*. Usually we assume that there exists an unknown target concept (a classifier), which correctly does this task of binary classification. Suppose we are unlucky and only have a weak classifier at hand, which is only slightly better than random guess on the underlying instance distribution D, say, it has an accuracy of 49% . Let's denote this weak classifier as h_1. It is obvious that h_1 is not what we want, and we will try to improve it. A natural idea is to correct the mistakes made by h_1. For this purpose, we can try to derive a new distribution D' from D, which makes the mistakes of h_1 more evident, for example, it focuses more on the instances wrongly classified by h_1. We can train a classifier h_2 from D'. Again, suppose we are unlucky and h_2 is also a weak classifier. Since D' was derived from D, if D' satisfies some condition, h_2 will be able to achieve a better performance than h_1 on some places in D where h_1 does not work well, without scarifying the places where h_1 performs well. Thus, by combining h_1 and h_2 in an appropriate way, the combined classifier will be able to achieve better performance than that achieved by h_1. By repeating the above process, we can expect to get a combined classifier which has very small (ideally, zero) 0/1-loss on D (small error rate).

Input: Instance distribution D;
Base learning algorithm L;
Number of learning rounds T *(iterations)*.
Process:
1. $D1 = D$. % Initialize distribution
2. **for** $t = 1, \cdots, T$:
3. $h_t = L(D_t)$; % Train a weak learner from distribution D_t
4. $\varepsilon_t = \Pr_{\boldsymbol{x} \sim D_{t,y}} \boldsymbol{I}[h_t(\boldsymbol{x}) \neq y]$; % Measure the error of h_t
5. $D_{t+1} = AdjustDistribution\ (D_t, \epsilon_t)$
6. **end**
Output: $H(\boldsymbol{x}) = CombineOutputs(\{h_t(\boldsymbol{x})\})$

Algorithm 4.8 A general boosting procedure.

Briefly, boosting works by training a set of classifiers sequentially and combining them for prediction, where the later classifiers focus more on the mistakes of the earlier classifiers.

Input: Data set D = {(x_1, y_1), (x_2, y_2), . . . , (x_m, y_m)}; with $y_i \in \{-1, +1\}$
Base learning algorithm L;
Number of learning rounds T.
The set of hypothesis H={h_1, h_2, …, h_k}

Process:

1. for i=1, …m
2. $D_1(x_i) = 1/m$. % Initialize the weight distribution associated to each Sample in the data set
3. Set a number of iteration T
4. for t = 1, ⋯ ,T: (OR $E > \varepsilon_t$)
5. $h_t = L(D, D_t)$; % Train a weak learner h_t over D using distribution D_t
   % h_t : D={(x_1, y_1), (x_2, y_2), . . .} ----→ y{-1, +1}
6. $\varepsilon_t = P_D[h_t(\boldsymbol{x_i}) \neq y_i] = \sum_{ht\,(\boldsymbol{xi}) \neq yi} D_t(i)$; % Measure the error ε_t of each % weak hypothesis h_t
7. if $\varepsilon_t > 0.5$ then **break**
8. $\alpha_t = \frac{1}{2} * \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$; % For each h_t Determine the weight α_t which % represents the importance given to this hypothesis % in the final combination ($\alpha_t > 0$ if $\varepsilon_t < 0.5$) % more ε_t is weak more α_t have an important value
9. for i=0, m

$$D_{t+1}(x_i) = \frac{D_t(xi)}{Z_t} * \begin{cases} \exp(-\alpha_t) & if\ h_t(\boldsymbol{x_i}) = \text{y correctly classified by } h_t(\boldsymbol{x}) \\ \exp(+\alpha_t) & if\ h_t(\boldsymbol{x_i}) \neq \text{y incorrectly classified by } h_t(\boldsymbol{x}) \end{cases}$$

$\frac{D_t(i) * exp(-\alpha_t y_i h_t(\boldsymbol{x_i}))}{Z_t}$ % Update the distribution, where Z_t is a normalization % factor which enables D_{t+1} to be distribution. % we assign the strong weights to the misclassified examples

10. **End**

Output: find a final strong hypothesis (classifier) $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t\, h_t(\boldsymbol{x_i})\right)$
And its error $Err(H(x)) \leq [2 * \sqrt{(\varepsilon_t(1 - \varepsilon_t)}$

Algorithm 4.9 The AdaBoost algorithm.

Another thing must be mentioned here, that a weak learner is some algorithm that is able to produce a weak hypothesis (classifier) $h : X \rightarrow \{\pm 1\}$ given the training data D together with a weight distribution W upon it. The "goodness" of a hypothesis is measured by its error.

4.4. Improving classifier performance

A general result in machine learning is that you can always get a small jump in classification accuracy by combining multiple classifiers. There is now a large literature on techniques such as voting, bagging, and boosting multiple classifiers. Nevertheless, ultimately a hybrid (automatic/manual) solution may be needed to achieve sufficient classification accuracy. A well-known approach in such situations is to run a classifier first, and to accept all its high-confidence decisions, in parallel to put low confidence decisions in a queue for manual review. This process also automatically leads to the production of new training data that can be used in future versions of classifiers.

4.5. Evaluation of text classifiers

Because the text categorization problem is not sufficiently well-defined, the performance of classifiers can be evaluated only experimentally. Any TC experiment requires a document collection labeled with a set of categories. This collection is divided into two parts: the training and the test document sets. The training set, as the name suggests, is used for training the classifier, and the test set is the one on which the performance measures are calculated. Usually, the test set is the smaller of the two (10% – 30%). It is very important not to use the test set in any way during the classifier training and fine-tuning. When there is a need to optimize some classifier parameters experimentally, the training set is further divided into two parts – the training set proper and a validation set, which is used for the parameter optimizations.

A commonly used method to smooth out the variations in the corpus is the *n-fold cross-validation*. In this method, the whole document collection is divided into *n* equal parts, and then the training-and-testing process is run *n* times, each time using a different part of the collection as the test set. Then the results for *n* folds are averaged.

4.6. Performance Measures

4.6.1. Recall, Precision and F-measure

The most common performance measures are the classic IR measures of *recall* and *precision*. A recall for a category is defined as the percentage of correctly classified documents among all documents belonging to that category, and precision is the percentage of correctly classified documents among all documents that were assigned to the category by the classifier.

Many classifiers allow trading recall for precision or vice versa by raising or lowering parameter settings or the output threshold. For such classifiers there is a convenient measure, called *the breakeven point*, which is the value of recall and precision at the point on the recall-versus-precision curve where they are equal.

Alternatively, there is the F1 measure, equal to *2/(1/recall + 1/precision)*, which combines the two measures in an ad hoc way.

To illustrate these measures, we refer to the table of contingence defined as bellow:

Table 4.5. Table of contingence used to evaluate classifier performance

		Expert	
		Docs belonging to c_i (a+c)	***Docs don't belong to c_i (b+d)***
Classifier	***Docs assigned by the classifier to c_i (a+b)***	*a*	*b*
	Docs rejected by the classifier from c_i (c+d)	*c*	*d*

Where:

a: the number of documents correctly assigned by the classifier to the classe c_i $(\in c_i)$

b: the number of documents incorrectly assigned by the classifier to the classe c_i $(\in \bar{c}_i)$

c: the number of documents incorrectly rejected by the classfier from the class c_i $(\in c_i)$

d: the number of documents correctly rejected by the classfier from the class c_i $(\in \bar{c}_i.)$.

$$Recall\ R = \frac{nb.Docs.Correctly\ assigned\ by\ classifier\ to\ c_i}{All.Docs.belonging\ to\ c_i\ (total\ of\ correct\ assignations\ to\ c_i)} = \frac{a}{a+c} \quad (4.55)$$

$$Precision\ P = \frac{nb.Docs.Correctly\ assigned\ by\ classifier\ to\ c_i}{All.Docs.Assigned\ by\ the\ classifier\ to\ c_i\ (total\ of\ assign.done\ by\ the\ classifier\ to\ c_i)} = \frac{a}{a+b} \quad (4.56)$$

$$Accurracy\_Rate = \frac{nb.Docs.Correctly\ assigned\ by\ the\ classifier\ to\ c_i\ and\ \bar{c}_i}{Total\ number\ of\ documents} = \frac{a+d}{a+b+c+d} \quad (4.57)$$

$$Error\_Rate = \frac{nb.Docs.inCorrectly\ assigned\ by\ the\ classifier\ to\ c_i\ and\ \bar{c}_i}{Total\ number\ of\ documents} = \frac{b+c}{a+b+c+d} \quad (4.58)$$

$$= 1 - Accurracy\_rate$$

$$Error\_Rate = 1 - Accurracy\_rate \quad (4.59)$$

$$F\_mesure = \frac{[(1+\beta^2)*(Precision*Rappel)]}{[\beta^2*(Precision+Rappel)]} \quad (4.60)$$

With ($\beta^2 = 1$) (in the most of case)

$$F\_mesure = \frac{[2*(Precision*Rappel)]}{[(Precision+Rappel)]} \quad (4.61)$$

N.B: Given a set of groups C and a collection of documents D

1. To compute the recall *R* and the precision *P* we need to answer two questions:
   - Are all documents assigned to a group relevant to this group? ➔ Precision of categorization
   - Are all relevant documents to a group have been assigned to this group? ➔ Recall (completeness)
2. If the precision is low, users will be dissatisfied; as it will waste time reading information that do not interest him. If the recall is low, users will not have access to information that he wished to have.
3. A perfect classifier must have *P=1 and R=1*, but these two requirements are often contradictory and very high Precision can be achieved with a recall and vice versa.
4. Two exceptional cases for *P* and *R* are possible: If no document is selected, the precision *P=1* and the rappel *P=0.* if all the documents are selected, the rappel *R=1* and the precision *P= (a+c)/(a+b+c+d)* which is a low value called group density.

4.6.2. Noise and Silence

a) Noise: is a ratio of the number of documents assigned by the classifier to a group, but which are not relevant to this group and the total number of documents.

$$Noise = 1 - Precision \quad (4.62)$$

b) Silence: is a ratio of relevant documents not assigned by the classifier and the total number of documents.

$$Silence = 1 - Recall \quad (4.63)$$

4.6.3. Micro and Macro Average

a) Micro-average: consists in calculating performance measures (recall, precision, F-measure) globally. i.e., do the sum of contingency tables associated to different categories to obtain an overall table. Then calculate measures for this new table.

$$P^* = \frac{\sum_{i=1}^{|C|} a_i}{\sum_{i=1}^{|C|}(a_i+b_i)} \tag{4.64}$$

$$R^* = \frac{\sum_{i=1}^{|C|} a_i}{\sum_{i=1}^{|C|}(a_i+c_i)} \tag{4.65}$$

b) Macro-average: consists in calculating performance (precision, Recall, F-measure) for each category. And then calculate the overall performance of the classifier by averaging the various individual measures.

$$\bar{P} = \frac{\sum_{i=1}^{|C|} P_i}{|C|} \tag{4.66}$$

$$\bar{R} = \frac{\sum_{i=1}^{|C|} R_i}{|C|} \tag{4.67}$$

Example 4.14: Precision-Recall curve

In this example, we draw the precision-recall curve for 11 recall levels.

i	$P(R_i)$ (%)	R_i (%)
0	100	0
1	100	10
2	100	20
3	100	30
4	80	40
5	80	50
6	71	60
7	70	70
8	70	80
9	62	90
10	62	100

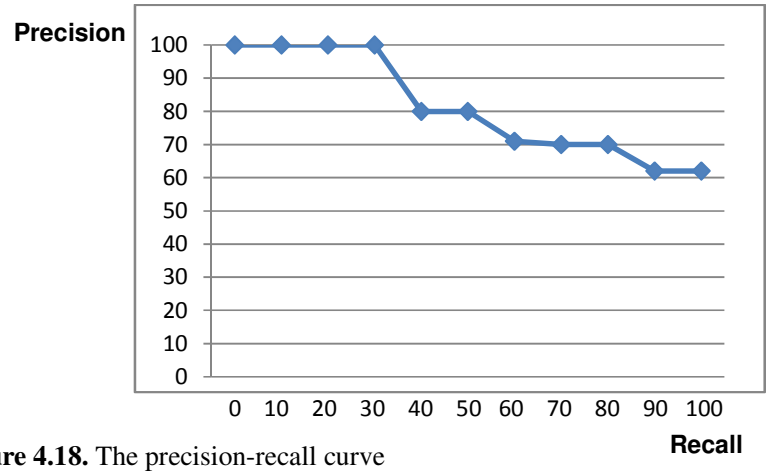


Figure 4.18. The precision-recall curve

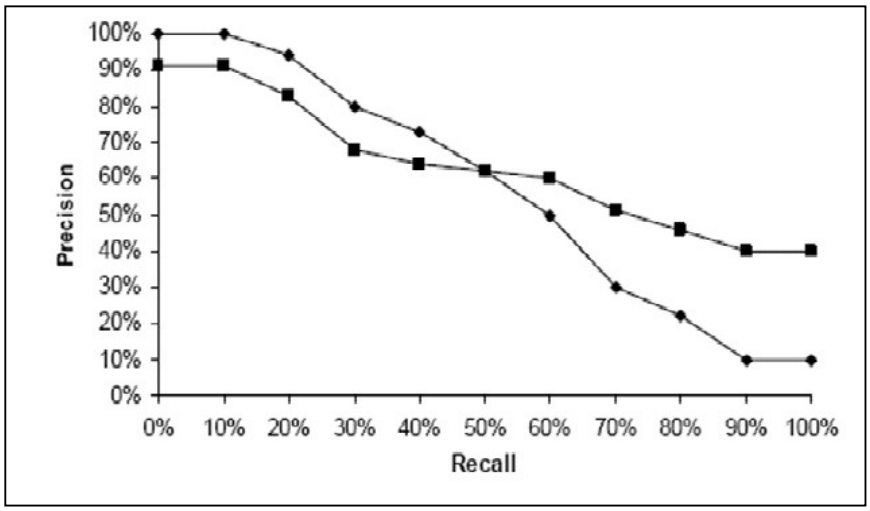


Figure 4.19. Comparison of two learning algorithms based on their Precision-Recall

4.7. Benchmark Collections

The most known publicly available collection is the *Reuters* set of news stories (Reuters21578, 90 categories), classified under economics-related categories. This collection accounts for most of the experimental work in TC so far. Unfortunately, this does not mean that the results produced by different researchers are directly comparable because of subtle differences in the experimental conditions.

In order for the results of two experiments to be directly comparable, the following conditions must be met:

1. The experiments must be performed on exactly the same collection (the same documents and same categories) using the same split between training and test sets.
2. The same performance measure must be chosen.
3. If a particular part of a system is compared, all other parts must be exactly the same. For instance, when comparing learning algorithms, the document representations must be the same, and when comparing the dimension reduction methods, the learning algorithms must be fixed together with their parameters.

These conditions are very difficult to meet – especially the last one. Thus, in practice, the only reliable comparisons are those done by the same researcher. Other frequently used benchmark collections are the *OHSUMED collection* of titles and abstracts of papers from medical journals categorized with MESH thesaurus terms, *20Newsgroups collection* of messages posted to newsgroups with the newsgroups themselves as categories, and the *TREC-AP collection* of newswire stories.

4.8. Comparison among Classifiers

Given the lack of a reliable way to compare classifiers across researchers, it is possible to draw only very general conclusions in reference to the question. Which classifier is the best? [Feldman and Sanger, 2007]

- According to most researchers, the top performers are SVM, AdaBoost, kNN, and Regression methods. Insufficient statistical evidence has been compiled to determine the best of these

methods. Efficiency considerations, implementation complexity, and other application-related issues may assist in selecting from among these classifiers for specific problems.

- Rocchio and Naive Bayes have the worst performance among the ML classifiers, but both are often used as baseline classifiers. Also, NB is very useful as a member of classifier committees.
- There are mixed results regarding the neural networks and decision tree classifiers. Some of the experiments have demonstrated rather poor performance, whereas in other experiments they performed nearly as well as SVM.

4.9. Summary

As we have seen in chapter 3 in knowledge engineering approach to TC, classification rules are developed manually by a domain expert who defines sufficient conditions for a document to be labeled with a given category. In machine learning approach, the classifier is built automatically by learning the properties of categories from a set of preclassified documents called training set. We talk here about a supervised learning because the process is completely guided by applying the known correct category assignment function on the training set. There are several ML algorithms which are used in classification task. In this chapter, we presented the most known ones, especially those used in TC field such as: Naïve Bayes algorithm, Rocchio, kNN, SVM, Decision trees, neuronal networks, regression methods, decision rules, Adaboost algorithm. We tried to explain the principle of each algorithm, its basic concepts, its pseudo-code, and the limits of its use. For some algorithms we gave examples for more clarity.

CHAPTER 5 Categorization of XML Semistructured Documents

Chapter content

5.1. Introduction

The development of web, as well as the electronic document gave birth of new formats of documents called semi-structured documents, such as HTML and XML documents. The advantage of these new formats, that they are able to describe both the structure of the document and its contents. This makes the information related to the document richer than a simple description of content. This rich description is very useful to: access the document, optimize its storage, or even query its content [Wisniewski et al., 2005]

Adding to all this, conventional methods of information research and document classification were primarily designed and developed to handle flat documents, i.e., they don't take into account the information about the structure of the document, hence, this structure can play a particular role in any kind of processing performed on the document.

This chapter briefly presents semi-structured documents XML, giving a global overview on XML language and the semantic of its tag, the notion of XML mining and its relation with other types of mining (data mining, text mining, Web Mining), presenting some approaches of XML documents categorization.

5.2. The web and HTML documents

The Web provides a set of structured multimedia documents. These documents are structured in two ways:

- Through the use of HTML format, which organizes the information elements of the same page.
- Through the use of hyperlinks which organize pages, each one with the other.

Several specific methods have been developed for the web processing, as well as in the field of document search [Cutler, 1997, Huang, 2000]. For example, a summary is available, for classification or extraction [Cohen, 1998, Cohen, 1999, Hsu, 1998, Muslea, 1999]. We note also, that the search engine Google was the first industrial application which takes into account the information contained in the links of web pages.

5.3. XML Semistructured documents

5.3.1. From a flat document to a structured document: a flat document is any text with one of the two types of marking: punctuation (',', ';', '.', '?', '!', …), and presentation marking (spacing, line break, page break, notes, etc...). These two types of marking are required to bring meaning to read or written text. It is also called an unstructured document. Unlike the flat documents, structured documents have a strict structure based on descriptive tagging. i.e., they resemble to the structure of a database, where the data is organized in tables consisting of columns and rows containing typical data.
A semistructured document can be placed between flat and structured documents, because it is not totally structured, so it is not considered as a strictly structured document, and it is not totally free as it is the case of a word document, but it is partially structured. This is the case of HTML, XML, XHTML, etc.

XML language allows us to produce documents in both structured and semi-structured format. It has become a standard tool of representation in the field of electronic documents. First, because it has a specific marking to be inserted in the text. This provides additional semantic elements concerning the structure and the format of the document. On the other hand, because it has a flexible, irregular and incomplete nature in its structure and its format. In addition, it allows to store and exchange data more easily than databases. [Ledmi, 2010]

5.3.2. XML language

XML (eXtensible Markup Language) is a standard developed by W3C. It defines a generic syntax used to mark data with a simple markup readable by users. The structure of an XML document is defined by elements containing attributes, as well as text and other information. These elements cannot overlap. The choice of element names as well as the attribute names and the organization of these elements is left to the author. This is why XML is considered generic.

5.3.3. XML Document

XML documents are the successors of HTML documents. They appeared together in the areas of RI and databases. An XML document may be represented as a tree of elements consisting of a root (a single root), branches and leaves. This tree gives the document a more strict structure.

```
<Library>
  <Book, id="1">
    <Title>The old man and the sea</Title>
    <Author>Ernest Hemingway</Author>
    <Kind>a novel</Kind>
    <PubDate>1952</PubDate>
    <Editor>Charles Scribner's Sons</Editor>
    <EditInfo>
      <OriginalTitle> The old man and the sea<OriginalTitle>
      <Language>American English</Language>
      <PageNum>191</PageNum>
      <Charcters>Santiago, Manolin</Charcters>
      <Setting>Cuba</Setting>
      <Awards>Pulitzer Prize for Fiction (1953), Premio Bancarella (1953), National Book Award Finalist
              for Fiction (1953), Nobel Price (1954) </Awards>
    </EditInfo>
  </Book>
  <Book, id="2">
    ……
  </Book>
</Library>
```

Figure 5.1 An example of XML document

We note the following terminology:

- *<Library>* and *</Library>* are tags (mark respectively: the beginning and the end of an element).
- *Id="1"* is an attribute having the value 1
- "*The old man and the sea*", "1952", … are textual elements.
- The element *<EditInfo>* is the parent of elements: *<OriginalTitle>, <Language>, <PageNum>*, … which are children.
- *<Book>* is the ancestor of elements: *<Title>, <Author>, <Kind>*, …which are considered as descendants.
- The element *<Library>* is the root because it represents the ancestor of all the other elements.

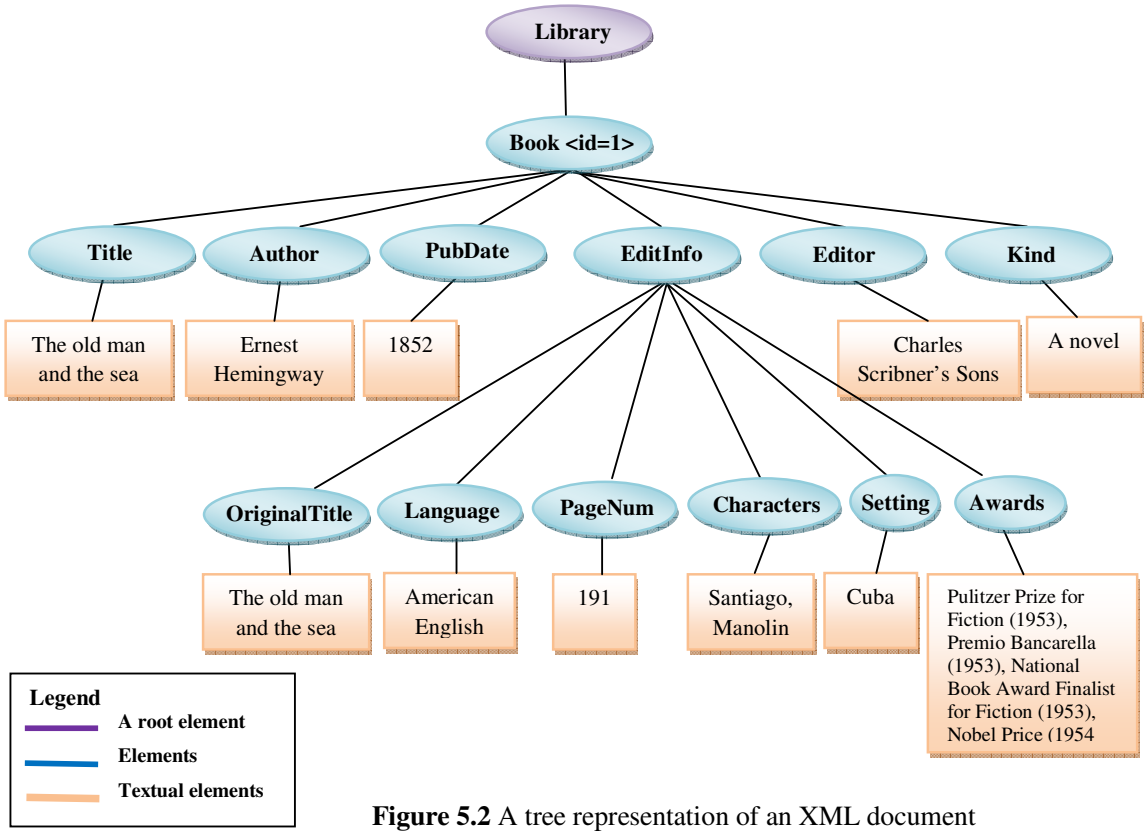


Figure 5.2 A tree representation of an XML document
(A simplified DOM representation)

5.3.3.1. The DTD (Document Type Definition)

The DTD (Document Type Definition) is a generic description of a class of XML documents. It imposes constraints regarding: the used tags, relations between these tags (inclusion, order, etc....). Two documents with the same DTD generally have the same structure, as they can have significant differences in their structure. We also said that a document respecting a given DTD is valid according to this DTD, and to be well represented, it must respect some simple rules:

- There must be only one root tag.
- Any open tag must be closed (the <tag> must be associated to <\tag>)
- Tag names must start with a letter or '_', the other characters can be: numbers, letters, "_", "." or "-".
- Tag names must not start with the string "xml" by convention,
- All tags are lowercase.
- When an item is empty, the tags can be simplified <tag> <tag> is identical to <tag>

```
<! ELEMENT Library ( Book *) >
<! ELEMENT Book (title , Author , PubDate ?, Editor ?, EditInfo ?, Kind?) >
<! ATTLIST Book id CDATA # REQUIRED >
<! ELEMENT Title (# PCDATA )>
<! ELEMENT Author (# PCDATA )>
<! ELEMENT PubDate (# PCDATA )>
<! ELEMENT Editor(# PCDATA )>
<! ELEMENT Kind (# PCDATA )>
<! ELEMENT EditInfo (OriginalTitle, language, PageNum, Characters, Setting, Awards) >
<! ELEMENT OriginalTitle (# PCDATA )>
<! ELEMENT Language (# PCDATA )>
<! ELEMENT PageNum (# PCDATA )>
<! ELEMENT Characters (# PCDATA )>
<! ELEMENT Setting (# PCDATA )>
<! ELEMENT Awards (# PCDATA )>
```

Figure 5.3. An example of DTD

5.3.3.2. XML DOM (XML Document Object Model): is the standard for accessing and processing XML documents. The DOM represents elements, attributes, and text within elements as nodes in a tree as it was in shown Figure 5.1. With a DOM API, we can process an XML document by starting at the root element and then descending down the tree from parents to children.

5.3.3.3. XPATH: XPath context is a standard for enumerating paths in an XML document collection. The XPath expression node selects all nodes related to that node. Successive elements of a path are separated by slashes. For instance, *<Book/EditInfo>* selects all *<EditInfo>* elements whose parent is *<Book>*. Double slashes indicate that an arbitrary number of elements can intervene on a path. For example, *<Library//EditInfo>* selects all *<EditInfo>* elements attached to <Library> element. An initial slash starts the path at the root element: *<Library/Book/Author>* selects the book author. *</Language/Kind>* selects no element. *<Title#"old">* selects all the titles containing the string "*old*".

5.3.3.4. Types of XML documents

There are two types of XML documents corresponding to two different needs, used especially for information retrieval and information extraction [Fuhr and Großjohann, 2001]:

a) ***XML documents oriented texts:*** like flat documents. i.e., the main purpose is to represent a text for reading. The markup is used here to provide information about the document

structure (chapters, sections, paragraphs, etc.) or about the form (bold, italic,... etc). As example of such documents, we can note: articles, manuals, books, static web pages.

```
<Book type = " Novel ">
   <title > Around the world in 80 days </ title >
   <author > Jules Verne </ author >
   <Editor >Hatzel</Editor >
   <chapter n="I">
      <chapter_Title >
          In which Phileas Fogg and Passepartout accept each other, one as master, the other as a maid.
      </ chapter_Title >
          In 1872, the house numbered 7 in Saville-Row, Burlington Gardens -House in which Sheridan
          died in 18th century - was lived by Phileas Fogg, Esq, one of the most singular and the most
          noticed member of the Reform-Club in London, …etc.
         ...
   </chapter >
</Book >
```

Figure 5.4. An example of textual XML document representing a novel

b) ***XML documents oriented data:*** are closer to databases used for representing and exchanging structured data sets (flight schedules, product catalogs, bibliographies, etc).

```
<bibitem type =" article " key =" SEB_02 ">
<title > Machine learning in automated text categorization </ title >
<author > Fabrizio Sebastiani </ author >
<year >2002 </ year >
<journal > ACM Computing Surveys </ address >
<pages > 1-47 </ pages >
<keywords >
   <item > Machine learning </ item >
   <item > text categorization </ item >
   <item > Fabrizio Sebastiani </ item >
</ keywords >
</ bibitem >
```

Figure 5.5. Representation of bibliographic element using an XML document

5.3.4. Semantic of tags

The adding of structure to an XML document offers some freedom in the organization of the text, including: the physical order of XML document elements may differ from the actual order of the text read by humans. [Lini et al., 2001] proposed three types of tags which can be used in the classification of XML structured documents. These types are:

5.3.4.1. Hard tags: are the most common in an XML document. They contribute to the structure of the document. As example of these tags, we can find: titles, chapters, subsections.

```
...
<title > Machine learning in automated text categorization </ title >
<author > Fabrizio Sebastiani </author >
...
```

Figure 5.6. An example of hard tags

5.3.4.2. Soft tags: define significant parts of the text, as the effects of form or corrections, but they are soft when we read the text.

```
...
The Title of the novel of <Bold > Ernest Hemingway </Bold > is <Italic>The old man and
the sea</italic>
...
```

Figure 5.7. An example of soft tags

5.3.4.3.Jump tags: are used to represent specific elements such as: notes margins, bibliographic references. They are detached from the text.

```
...
The XML language <note > eXtensible Markup Language <\ note > has become a standard
mode of representation for electronic documents .
...
```

Figure 5.8. An example of jump tags

5.4. XML Mining

Firstly appeared in [Lee et al., 2001], it is an extension of the data mining concept but oriented to XML documents. It was the result of cumulative efforts made in various disciplines such as: data mining, text mining, web mining, etc.

However, the old techniques of these disciplines are not valid, which requires the development of new approaches that are appropriate to XML documents [Jeang et al., 2004].

Table 5.1. Comparison between Data Mining, Text Mining, Web Mining and XML Mining

Discipline	**Type of data**	**Type of problems**	**Applications**
Data Mining	Vector/Numerical/categorical data	Clustering, classification, regression, prediction	- Pattern recognition. - Fraud detection. - Bioinformatics.
Text mining	Textual data	-Vector Space Model VSM - NLP	- Text Categorization - Clustering - Information retrieval - Information extraction - Text Summarization - NLP -LangI identification - Author identification
Web Mining	Texts – Multimedia	- Web content mining - Web structure mining - Web usage mining	- RI - Topology analysis - Usage forms analysis
XML Mining	Structured/Semistructured texts	- Knowledge formalisms - Structural similarity	- Structured retrieval - Structured text categorization - Ontology mapping. - Schema matching. - Composition of web services.

5.5. Structured information retrieval

The field of Structured Information Retrieval (SIR) is the extension of the classical field RI oriented to structured documents and particularly to semi-structured documents XML. This is an emerging field that has developed during the last years, particularly with the appearance of the INEX initiative and several Workshops in different conferences of RI (e.g., SIGIR conference). It is interested in adapting the classical RI models for simultaneous consideration of the structure and the content as well as, the definition and the evaluation of new problems specific to semistructured data. Studies done on this subject show that the usual methods of representation particularly vector methods are not suitable for XML data representation. Hence, the need to adapt the classic models of RI with the new formats. It is also important to note how these new problems are emerged simultaneously with the emergence of this type of data. Among these problems, we find the problems that arise from the phenomenon of structural heterogeneity [Denoyer, 2004].

5.5.1. Problems related to the representation: we have seen in Chapters 2 and 3 several representation methods for flat documents. These representations are not transferable to structured documents: in fact, the problem of representation and indexing structured documents for IR remains an open problem, particularly for the construction of fast retrieval systems. [Grust, 2002] proposes indexing XML documents to speed up the system response time for XPath queries. This approach was later extended to the specific problem of structured information retrieval in [Olteanu, 2002]. Several other solutions will be presented throughout this chapter.

5.5.2. The need to adapt the old models: As we noted in the previous section, information related to the document structure is not considered in classical models of RI and TC. However, it is a very important information having a strong semantics. E.g., for document categorization task, the title words can tell us on the subject of the document which is not the case for the page number or the name of the author, which are less informative for identifying the topic. So, it is very beneficial to consider the information of structure by RI and TC models because it provides additional information available directly through the new formats. The adaptation of current models with new document formats is justified by: [Denoyer, 2004]

1. The conventional models of RI and TC are not able to process semi-structured documents, but only short and uniform documents.
2. Semi-structured documents are often larger because the need for structuring occurs when the volume of data increases as in the case of web documents.
3. Suitable models must bear: the size, the complexity, and the heterogeneity of content.
4. The notion of document is totally different in the structured case. For example, the assum--ption of independence between documents became wrong with the appearance of hyper--links.
5. The structural information can help in improving the performance of models. Because it is informative for the following tasks: classification, information retrieval, information extraction, clustering.

6. The vector model used by conventional models is not flexible for consideration of tree structures (XML documents) which requires redefining new model families able to consider these new representations of documents (trees, graphs, sequences, etc.) .
7. The appearance of semi-structured data leads to the appearance of new problems and new needs for which conventional models don't bring any response.
8. The introduction of structural information in document coding gives new information units (Websites, web pages, hyperlinks).

5.5.3. The INEX initiative and proposed solutions: this initiative initially allowed the creation of the first corpus of XML documents, queries interrogating these documents and a specific procedure for the evaluation of information retrieval systems (INEX 2002). For queries, INEX proposes two types:

5.5.3.1. The CO Queries (Content Only): have the same form as normal queries of RI based on key words, and eventually united by Boolean operators (e.g., "*computer science web categorization"*, "*hardwareANDsoftwareORsystemANDinformation*". The essential difference with flat corpora lies in the definition of the information that we want to return. INEX defines a relevant information unit as the structural unit which covers both the application and that is as concise as possible.

5.5.3.2. CAS queries (Content and Structure): are specific queries of structured documents that express an information request combining information content and structure at the same time (e.g., "*I want documents that a paragraph talking about computers, whose title contains the word "Computer graphics", and which was published between 2000 and 2004* ").

5.5.3.3. Evaluation of structured RI system: is not simple as in RI system on flat documents. Because structured documents are large and can treat various topics. Therefore, it is not easy to know whether a document that only one of its pieces responds to the request must be considered relevant. It may make more sense in this case to say that only a part of the document is relevant (a paragraph) and not the entire document (total relevance Vs partial relevance). For this, INEX Company offers a new measure of relevance which is firstly defined in two dimensions, unlike the case of simple RI where the evaluation is done on one dimension: "relevant or irrelevant."

a) ***The component coverage dimension:*** dimension evaluates whether the element retrieved is structurally correct, that is neither too low nor too high in the tree. We distinguish four cases: exacte coverage (E), too small coverage (S), too large coverage (L), no coverage (N).

b) The topical relevance dimension: also has four levels: highly relevant (3), fairly relevant (1), relevant (2), not relevant (0).

The following table summarizes the possible values taken by each raised dimensions.

Table 5.2. Possible values taken by the two dimensions of the new measure of relevance proposed by INEX

Dimension	Possible values
Topical Relevance dimension	0: not relevant
	1: fairly relevant
	2: relevant
	3: highly relevant
Component Coverage dimension	N: "no coverage": the information sought is not a topic of the component.
	L: "too large": the information sought is present in the component but is not the main topic.
	S: "too small": the information sought is the main topic of the component, but the component is not a meaningful (self-contained) unit of information.
	E: "Exact": the information sought is the main topic of the content and the component is a meaningful unit of information.

That means that the evaluation criteria: recall, precision are not valid to evaluate the relevance of structured documents. For this purpose, the INEX community shows that the problems and their evaluation methods evolve along with the data and structure evolve.

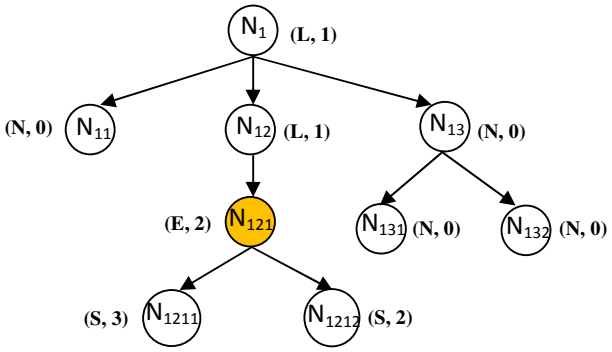


Figure 5. 9. Evaluation of relevance of an XML document using the INEX measures.

Figure 5.9 shows that the node which should ideally be returned by the system is the node N_{121} which corresponds to a relevant and accurate information (E, 2).
The nodes N_{1211}, N_{1212} are too specific ((S, 3), (S, 3)) despite their relevance.
The nodes N_1, N_{12} are too large with low relevance (L, 1).
The nodes N_{11}, N_{13}, N_{131}, N_{132} are completely irrelevant (N, 0).

In theory, there are sixteen combinations of coverage and relevance, but many cannot occur. For example, a non relevant component cannot have exact coverage. So, the combination (N, 3) is not possible.

5.5.4. Problem of semi-structured documents heterogeneity

The Access to structured documents raises the problem of semantic heterogeneity of structure. For example, two editors of scientific journals do not use the same description while these same journals can treat the same area. This creates a great diversity and variability in the

structure of documents. This phenomenon comes from the fact that XML format allows the user the freedom to design its own representation model. For example, the address of a person can be represented either in a highly structured way or unstructured way as it's shown in the figure below.

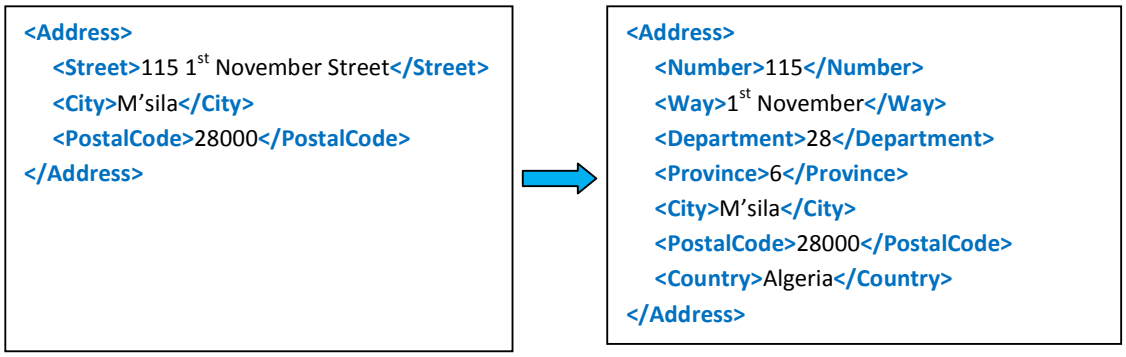
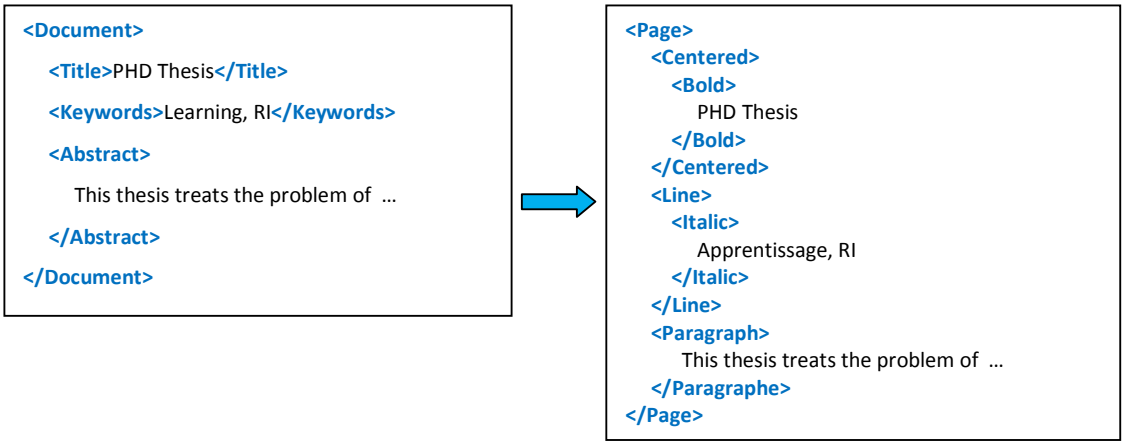


Figure 5.10. Two structural representations for the same information "Address"

A second problem is that the structure of a document is not always informative (partially or fully) to a defined task as the thematic classification or printing.



(a) Informative for classification task (b) Informative for printing task

Figure 5.11. two structural representation for the same XML document

5.5.5. Querying and heterogeneity.

Suppose that a user wants to query a corpus of structured documents. He will use a CAS query (Content and Structure). For example, he writes the following query: "*I want a restaurant located in the city of M'sila, in the 5th region where I can eat fish.*"

Consider that there are three XML documents D_1, D_2, D_3 describing restaurants. Find the answer to the query seems trivial for D_1 which has a structure where the needed information is perfectly located. Here, we must only translate our query to an XPath query to get the right result. For D_2, D_3 the task is more difficult because we must locate the desired information in parts where the content is unstructured.

```
<Restaurant>
  <Name>BaniHammed </Name>
  <Adresse>
    <City>M'sila</City>
    <Region>5</Region>
   <Street>Col. Chaabani</Street>
    <Num>127</Num>
  </Address>
  <Plate>Couscous</Plate>
  <Plate>Fish</Plate>
</Restaurant>
```

```
<Restaurant>
<Name>Royal</Name>
<Address>
   65 Martyrs Street, Msila, 5th
   region, Algeria
</Address>
<Plates>
    Fish soup, Chicken plate,
    Rabbit plate.
</Plates>
</Restaurant>
```

```
<Restaurant>
<Name>The desert</Name>
<Description>
 This pretty restaurant located
 Near the house of
 culture, 19th Independence
 Boulevard, lost in the 5th
 region of M'sila, propose a
 traditional kitchen, notably of
 fresh pasta with 3 cheeses
</Description>
</Restaurant>
```

Figure 5.12. Three XML documents describing restaurants with different structures.

To overcome this kind of problems, two types of solutions are possible:

1. Transform the set of documents in the corpus in a unified format called mediation format readable by the user making the query, for example the format of the document D_1. The documents D_2, D_3 will be transformed and queries will also be defined using this format.
2. Transform the query so that it "fits" with each examined document. This solution is possible when information is decentralized (distributed) and when it is not possible to convert documents into a unified format of mediation. Both strategies were used in databases.

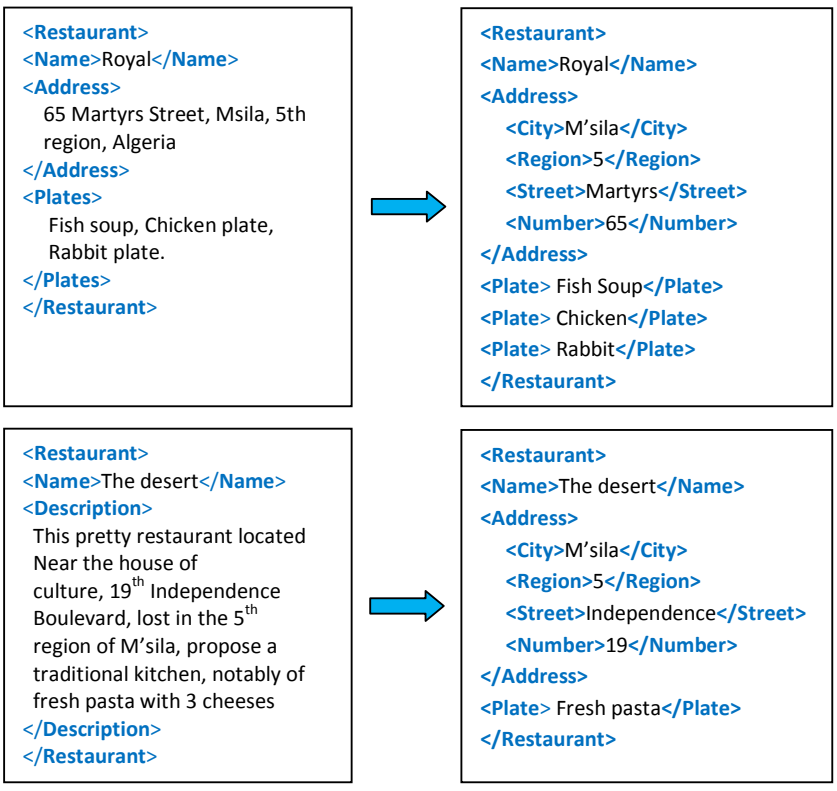


Figure 5.13. Restructuring of XML documents in a unified format (mediation format)

5.5.6. Conversion of document formats

Sometimes a company's documents are stored in different digital formats (.Doc, .pdf, .xls, etc.). Therefore, it would be necessary in this case to have a single format by performing a format conversion. This problem is very wide. It also includes the notion of mediation format seen previously.

5.6. Categorization of XML Semistructured documents

As we have seen with flat documents, XML semi-structured documents can be the subject of a categorization task depending on their topic. There exist two classes of approaches: approaches that take into account the content and the structure simultaneously and approaches based on the analysis of the structure.

5.6.1. Approaches based on the structure and the content

5.6.1.1.[Yang and Zhang, 2007] approach: [Yang and Zhang, 2007] proposed a new vector representation more suited to structured case. It is an extension of the VSM model (Vector Space Model). This new representation is called SLVM (Structured Links Vector Model). The basic idea is to incorporate in the classic vector model the structure of XML documents and then apply a learning algorithm often SVM on these vectors in the classification phase. The new model was first tested on Wikipedia 2007 corpus.

SLVM represents an XML document as VSM vector matrix, where each VSM vector is specific to an XML element (a tag).

Thus, the matrix M_d corresponding to document *d* has the following form:

$$M_d = \begin{bmatrix} m_d(1,1) & m_d(2,1) \cdots & m_d(m,1) \\ m_d(1,2) & m_d(2,2) & m_d(m,2) \\ \vdots & \ddots & \vdots \\ m_d(1,n) & m_d(2,n) \cdots & m_d(m,n) \end{bmatrix} \quad (5.1)$$

Where:

m: the number of elements (tags) of the document *d*.

n: the number of terms appear within the current tag.

$M_d(i, j)$ is given by:

$$M_d(i,j) = TF(w_j, e_i | doc_d).Idf(w_j) \quad (5.2)$$

Where:

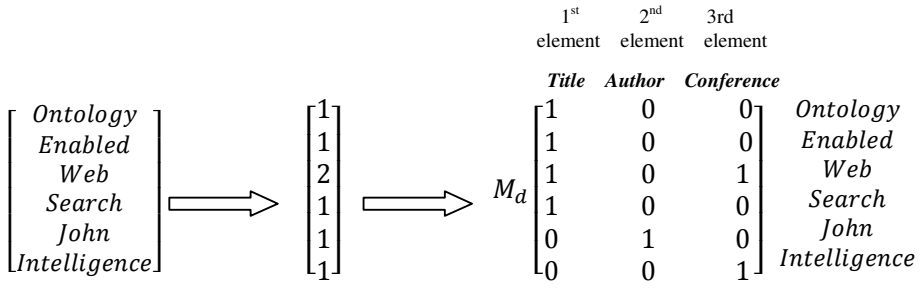
$TF(w_j, e_i | doc_d)$: is the term frequency *wj* within the element e_i of the document *d*.

This can be normalized by:

$$\widehat{M}_d(i,j) = \frac{M_d(i,j)}{\sum_{i=1}^{n} M_d(i,j)} \quad (5.3)$$

Example 5.1: SLVM Model
Let the following XML document:

```
<article >
<title > Ontology Enabled Web Search </ title >
<author > John </ author >
<conference > Web I n t e l l i g e n c e </ conference >
</ article >
```



(a) Vocabulary ***(b) Associated frequency vector*** ***(c) Document matrix***

Then, the normalized matrix is:

$$\widehat{M}_d = \begin{matrix} \\ 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \\ 0 \\ 0 \end{matrix} \begin{matrix} \textit{Title} \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{matrix} \begin{matrix} \textit{Author} \\ 0 \\ 0 \\ 1/2 \\ 0 \\ 0 \\ 1/2 \end{matrix} \begin{matrix} \textit{Conference} \\ \\ \\ \\ \\ \\ \\ \end{matrix}$$

(d) The normalized matrix of an XML document

Figure 5.14. Representation of an XML document using SLVM model

After representing the XML document, the classification is made by applying the SVM method between two XML documents as follows:

$$K(X_x, X_y) = Sim(D_x, D_y) = \sum_{i=1}^{n} M_{x_i}^{n\,T} * M_s * M_{y_i}^{n} \qquad (5.4)$$

Where:
M_s: an *m x m* matrix called similarity matrix of XML elements.

5.6.1.2. [de compos et al, 2007] approach: is based on the probabilistic model. It considers that certain structural attributes of an XML document can be translated into flat text. Several solutions are proposed to add structural information, as well as two probabilistic models: Naive Bayes model, Gate model.

These two models were validated on Wikipedia 2007 corpus. Recall that the classical probabilistic approach to categorize flat texts can be presented as follows:

- Given a set of classes $C = \{c_1, c_2, \ldots, c_n\}$
- A document d_j to classify

The conditional probability of each class $p(c_i | d_j)$ or the probability that the document d_j belongs to the the class c_i is calculated using Bayes theorem as follows:

$$P(c_i|d_j) = \frac{p(c_i).p(d_j|c_i)}{p(d_j)} \propto P(c_i).p(d_j|c_i) = P(c_i)\prod_{1\le k\le n_d} P(t_k|c_i) \quad (5.5)$$

And the document d_j is assigned to the class having the highest probability

$$c^*(d_j) = \text{Argmax}_{c_i \in C}\ \{P(c_i).p(d_j|c_i)\} = Argmax_{c_i \in C} P(c_i)\prod_{1\le k\le n_d} P(t_k|c_i) \quad (5.6)$$

So, the problem comes back to estimate two probabilities: $P(c_i)$, $p(d_j|c_i)$
For this, we can use one of the models proposed previously, which is the model Naïve Bayes or Naïve Bayes classifier.
In this model, a document is considered as an ordered sequence of words/terms extracted from the same vocabulary. We assume that the occurrences and the positions of these terms in a document are conditionally independent for a given class.

So, $p(d_j|c_i)$ is calculated according to the following equation:

$$P(d_j|c_i) = p(|d_j|)\ \frac{|d_j|!}{\prod_{t_k \in d_j} n_{jk}!} \prod_{t_k \in d_j} P(t_k|c_i)^{n_{jk}} \quad (5.7)$$

Where:
t_k: is a term in d_j
n_{kj}: the frequence of the term k in the document d_j.
$|d_j|$: the total number of terms in the document d_j.

You can see that: $p(|d_j|)\ \frac{|d_j|!}{\prod_{t_k \in d_j} n_{jk}!}$ doesn't depend of the class c_i

Then:

$$P(d_j|c_i) \propto \prod_{t_k \in d_j} P(t_k|c_i)^{n_{jk}} \quad (5.8)$$

The estimation of the probability $P(t_k|c_i)$ is given by Laplace equation:

$$P(t_k|c_i) = \frac{N_{ik}+1}{N_i+|V|} \quad (5.9)$$

Where:
N_{ik}: The frequency of the term t_k in the documents of the class c_i.
N_i: The total number of terms in the documents of the class c_i.
$|V|$: The size of term vocabulary.

The estimation of the prior probability of the class c_i is given by:

$$P(c_i) = \frac{N_{i,doc}}{N_{doc}} \quad (5.10)$$

Where:
$N_{i,doc}$: The total number of documents in the class c_i.
N_{doc}: The total number of documents in the corpus.

Adding structural properties

```
<Book >
<Title > Introduction to information retrieval </ title >
<author > Manning et al </ author >
<content >
   <Chapter>One </ Chapter >
   <Text>The meaning of the term information retrieval (IR) can be …</Text>
Content >
</Book>
```

Figure 5.15. A fragment of an XML document

To represent XML documents taking into account the structural information, several methods have been proposed by authors. These methods can be summarized as follows:

1. "Text Only" Method: is the simplest method, consists in taking only the text (i.e., ignoring any tags).

```
Introduction to information retrieval
Manning et al One The meaning of the term information retrieval (IR) can be
```

Figure 5.16. Representation of the previous XML document with the "Only Text" method

2. "Adding" Method: consists in considering tags of an atomic manner, or in other tags where they are included (i.e., to consider a part of the way from the root element to a certain level of depth).

```
_Book_Title Introduction to information retrieval
 _Book_Author Manning et al <_Book_Content_Chapter One _Chapter _Text The
meaning of the term information retrieval (IR) can be …
```

Figure 5.17. Representation of the previous XML document with the "Adding" method

3. "Tagging" Method: the text elements (terms) are prefixed with tags (Up to a certain depth level).

```
Title_ Introduction Title_to Title_information Title_retrieval
author_ Manning author_ et  author_ al
Chapter_ One  Text_The Text_meaning Text_of Text_the Text_term
Text_information Text_retrieval Text_ (IR) Text_can Text_be
```

Figure 5.18. Representation of the previous XML document with the “Tagging” method

4. "No Text" Method: This method considers only the tags, so it is equivalent to the "Adding" method while eliminating the textual terms.

```
Book_Book_Title
Book_Author_ Book_Content
Book_Content_Chapter
Book_Chapter_ Text_
```

Figure 5.19. Representation of the previous XML document with the “No Text” method

5. "Text Replication" Method: consists in assigning an integer value to each tag, depending on the importance of its informative content for categorization (i.e., more the value is high, more the content is informative). For example: (title 1) (author 0) (chapter 0) (text 2)

```
Introduction to information retrieval
The meaning of the term information retrieval (IR) can be …
```

Figure 5.20. Representation of the previous XML document with the “Text Replication” method

5.6.2. Approaches based on the structure only

5.6.2.1. [Zaki and Aggarwal, 2003] approach: This approach proposes a structural classifier that use rules called "XRules". This classifier is based on the search of structural rules that can help to perform the classification task, starting from the idea that the presence of a particular type of structural model in an XML document is related to its membership to a particular class.

Initially, the approach proposes the representation of each XML document with a labeled and ordered tree, having a root and denoted by: $T = (V, B)$, Where:

V: is the set of labeled nodes.

B: is the set of branches, knowing that a branch $b = (x, y)$ is a pair of ordered nodes where x is a parent of y.

The classification task is performed in two main phases:

1. ***The learning phase:*** its role is to find the set of structural rules associated with different categories (a model for each category).

Formally, we can define this phase as follows:
Given a set of documents

$$D = D = \bigcup_{i=1}^{k} D_i \quad (5.11)$$

Where: Di : is the set of documents belonging to the class $c_i \in C = \{c_1, c_2, \ldots, c_n\}$.

As it was mentioned before, The objective is to find a set of structural rules:
$R = \{R^1, R^2, \ldots, R^m\}$
Where: every rule is of the form: $T^i \overset{\pi,\delta}{\Rightarrow} C_j^i$ with $\pi \geq \pi_j^{min}$ et $\delta \geq \delta^{min}$

This phase is in turn divided into three steps:

a. ***Discover the frequent structural models*** associated to each class c_i, then produce rules that meet a minimum support level of media π_i^{min} of the class c_i and a global minimum threshold of confidence δ^{min}.
b. ***Order the classification rules*** obtained in the previous phase according to a priority relationship. i.e., give priority to the set of rules in descending order to remove non-predictive rules.
c. ***Determine a special class*** called "*default class*". This class will be the label of the document if any of the rules have been able to predict the class of this document.

2. ***Test phase:*** to predict the category of a new document using the classification model built in the learning phase (all the structural rules). So, it takes as input, the classification model (the set of predictive rules R_i + default class), and all XML documents $\grave{D}$ whose classes are unknown. This phase is in turn made in two main steps:

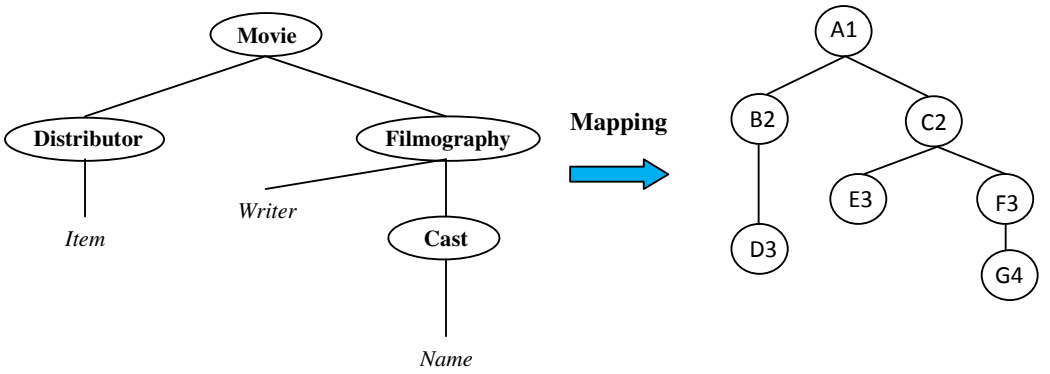
   a. ***Search of rules:*** for each document $d_i \in \grave{D}$, we seek the set of all rules corresponding to this document. This set is called matching rule-set, and denoted by:

$$R(S) = \{R^i : T^i \overset{\delta^i}{\Rightarrow} c^i | T^i\} \quad (5.12)$$

   b. ***Class prediction:*** Combining statistics for all matching rules *R(S)* in order to predict the appropriate class for a given document. if $R(S) = \phi$, then the document belongs to the default class.

5.6.2.2. [Garboni et al., 2005] approach: this approach is based on the calculation of a similarity measure between an XML document and each category. For this, each document is converted into labels sequences of its nodes. Similarly, each class is transformed into a set of sequences. The measure of similarity between an XML document d_j and a category c_i is thus calculated as the longest common subsequence between the sequence of the document and the sequences of different categories. The approach has been validated on the corpus "Movie".

Before making such transformation, the tree nodes associated to the XML document must first be labeled. Each label is associated to its depth in the tree. An exploration of the tree in depth gives the corresponding order. This phase is called the "reduction" phase (see figure below).



Reduction Result: *<(A1)(B2)(D3)(C2)(E3)(F3)(G4)>*

Figure 5.21. Transformation of an XML document tree into sequence

Once the necessary transformations are made and the sought sequences are extracted, we proceed to the extraction of the model that helps to extract frequent sequences. The model extraction involves three basic steps:

1. Cleaning step: consists in the removal of labels (tags) which are irrelevant for classification operations. For example, a label which is very common in the entire collection can be considered irrelevant, since it does not help in the separation between documents (non-distinctive label).

2. Data mining step: After eliminating frequent labels in step (1), we try for each category, to transform all XML documents in sequence as it was explained above. Then for each set of sequences corresponding to the different categories, some data mining tasks are executed to extract sequential patterns. This give us for each category c_i, a set of frequent sequences SP_i which characterize The associated category c_i.

3. Attribution of the document to an appropriate class: consists in calculating the similarity based on the longest common subsequence between a document d_i and the different categories c_j .

$$sim(d_i, c_j) = \frac{\sum_{k=1}^{|c_j|} \frac{|LCS(D_i, SP_{C_j}(k))|}{SP_{C_j}(k)}}{|c_j|} \qquad (5.13)$$

Where:

$|LCS(D_i, SP_{C_j}(k))|$: is the length of the longest common subsequence between the document d_i and the kth sequential model of c_j.

The document d_i is then assigned to the class with which the similarity value is maximum.

5.7. Summary

XML is an extensible markup language which is not limited to a predefined markup as it is the case of its predecessors HTML and SGML. an XML document is an universal file encoded in Unicode, self-descriptive, scalable, self-checking, and semantically rich containing semi-structured information that can be described using a graph or tree (a DTD schema, XML schema, etc.) making them easier to access and handling.

With an XML document, we can store documents having generally large sizes such as: a book, a user guide, multimedia information, an electronic message, etc.

In this chapter, we presented a global overview of XML document, notably: its structure, characteristics, principle components, semantic of tags. We also discussed the different problem of representation and the need of adapting the old models which are not suitable with its new format. In a large part of this chapter, we treated a critical problem well-known in the field which is the categorization of XML documents. Effectively, we exposed the problem and we presented the most known approaches including: the content only approaches, and the content and structure approaches.

Contribution Part

Chapter 6: Identifying the Language of a Text in a Collection of Multilingual Documents

Chapter 7: Contextual Categorization using a New Panoply of Similarity Metrics

Chapter 8: Arabic Text Categorization: An Improved Stemming Algorithm to Increase the Quality of Categorization

CHAPTER 6

Identifying the Language of a Text in a Collection of Multilingual Documents

Chapter content

6.1. Introduction

Needs of users in information keep growing because of the massive expansion of Internet and the increasing availability of textual data on this network. The multilingual documentary research is one of the solutions proposed to satisfy these needs. The main objective of this research is to allow the user to formulate a query in a language and extract the corresponding documents in other languages. This process requires a very important phase that consists in identifying the language of every document before extracting the required information. The language identification can be defined as the assignment of a document in a given language. Several features of the language can be considered in this case, notably: the presence of some characters [Mustonen, 1965, Souter, 1994], presence of some words [Souter, 1994, Giguet, 1998], n-grams frequencies [Beesly, 1988, Cavenar and Trenkle, 1994, [Dunning, 1994, Grefentette, 1995, Milne and al., 2012, Zamperi and Gebre, 2012].

6.2. State of the art

6.2.1. Language Identification Approaches

In general, four approaches exist:

6.2.1.1. The linguistic approach: It is based on the presence of some strings in the text specific to its language. This permits to recognize the text language easily [Mustonen, 1965, Souter, 1994]. For example, the characteristic strings of English are (they, ery,…), for French (eux, aux, …). Unfortunately, this approach comprises several problems [Dunning, 1994], namely: these strings can be rare or absent in the text. A text in a language can also use strings of other languages.

6.2.1.2. The lexical approach: it is based on the use of a lexicon for every language [Souter, 1994, Giguet, 1998]. E.g., we compare words of the new text with a fixed list of words for every language. The language whose lexicon contains all or most words of the text is the effective language of the text. Several problems can arise here: no language lexicon is complete, the presence of mistakes which can disrupt results gotten by this approach.

6.2.1.3. The grammatical approach: it is based on the presence of some grammatical words related to the language [Grefentette, 1995] such as: prepositions, determiners, adverbs, auxiliaries, etc.. representing approximately 50% of texts in most languages [Giguet, 1998]. For example, in English we can find words like (the, they, are, ...) in French (le, la, des, …). This approach is faster and more efficient than the two previous approaches, but it also embodies several weaknesses, such as: the texts should be segmented into words which is difficult for some languages, grammatical words are often removed during a preprocessing performed on the text [Sahami, 1999], the approach gives poor results in the case of short texts because of the absence of these grammatical words, it is also difficult to apply with other types of sequences (DNA sequences).

6.2.1.4. The statistical approach: this approach does not require prior language skills but probability calculations. Its principle is to capture some formal regularities (words, n-grams) of languages using statistical model from a representative corpus for each language, to associate a frequency of occurrence, and then calculate the probability that a text belongs to a language based on the appearance of these regularities [Geiger et al., 2012]. Unfortunately, segmenting a text into words degrades the performance of this approach because a short text does not necessarily contain the most frequent words of the language. In addition, for some languages like Chinese, it is difficult to divide the text into words. For this purpose, the use of n-grams-based approach [Shannon, 1948, Miline et al., 2012, Zampeiri and Gebre, 2012] seems to be a widespread solution.

6.2.2. Principle of Text Segmentation into N-grams of Characters

An n-gram is a consecutive sequence of n characters [Cavenar and Trenkle, 1994] that cannot be ordered. For a document, the set of n-grams that can be generated is a collection of photos that we can obtain by moving a window of n characters with one character on the body of the text. For example, the 3-grams of the phrase "hello sir" are : hel, ell, llo, lo_, o_s, _si, sir [Miller et al., 1999, Biskri and Delisle, 2001, Jalam and Teytaud, 2002]. The n-grams profile of a document is the list of the most frequent n-grams in reverse order of their frequencies.

6.2.3. Advantages of Text Segmentation into N-grams

The approach of text segmentation into characteristic N-grams has several advantages, in particular:

- Avoid the use of lemmatization and stemming phase on the text which requires an algorithmic and linguistic effort.
- Tolerant to spelling, typing, and OCR mistakes.
- This approach operates independently of languages (language independent).
- Segmentation into words is difficult for some languages, e.g., in Arabic names and additional subjects are in some cases attached to verbs and the string is thus a sentence like: (katabtouhou) (I wrote it) [Biskri and Delisle, 2001, Jalam and Teytaud, 2002].

6.2.4. Methods Based on N-grams for Language Identification

Language identification systems using n-grams are based on almost the same pattern [Jalam and Teytaud, 2002]: the phase of automatic acquisition of the language, in which we select a representative corpus, we generate a characteristic profile that will be taken as a reference, after we calculate the frequencies of various n-grams (with $n = 3, 4, 5$). The diagnostic phase in which we build for each new text its n-grams profile then we seek the reference profile which is the most similar. Several methods are proposed to measure this similarity.

6.2.4.1. Nearest Neighbors Methods: Several algorithms of language identification of texts are based on the concept of distance or similarity. The basic idea is to find the text of the training set that is nearest in distance to the new text to classify, and to assign to it the same language. You can increase the number of k texts nearest to the new text if necessary. In this case, the language of the new text is the same as the majority of their k nearest neighbors (the majority language). The challenge for these approaches is how to define a distance measurement. Practically, we use several pseudo-distances measurement, including:

The Distance of Beesly [Beesly, 1988]***:*** In this method, the identification consists of two phases: the learning phase that consists in segmenting texts of each language L into words, then segmenting each word into bi-grams without repeating the letter more than once, building a bi-grams profile for each language and use it as a reference profile, then calculate the frequency or the probability of occurrence of each bi-gram in this profile. The diagnostic phase that consists in establishing the bi-gram profile of the new text T, then find the nearest reference profile using the distance with the profile of the language L by calculating the product of bi-grams probabilities of the new text T found in the profile of the language L (Naive Bayes algorithm). This method assumes the possibility of segmenting texts into words which is not the case for some other languages. In addition, it is based on bi-grams, then it is necessary to not neglect working with tri or quad-grams to maintain the specificity of each language. For example, the 4-gram "tion" characterizes French and English, if you segment into bi-grams as follows: "ti" and "on", the system finds some difficulties to differentiate with "ti" and " on " of Spanish and Portuguese [Zamieri and Gebre, 2012, Jalam and Teytaud, 2002, Brown, 2012].

The Distance of Cavenar and Trenkle [Cavnar and Trenkle, 1994]***:*** This method consists of two phases: the acquisition phase that consists in establishing a tri-gram profile for each language *L,* than use it as a reference profile. The diagnostic phase which involves building a tri-gram profile for the new text *T*, then calculate distances between this profile and reference profiles of the different languages. The distance to be calculated is based on the sum of position errors (difference in ranks or "out-of-place" measurement [Cavnar and Trenkle, 1994]) between each tri-gram in the new text profile P_T and the same tri-gram in the reference profile of each language P_L if the tri-gram is present. Otherwise, it takes a maximum value of rank. The language of the new text is that for which the distance is minimal. Formally, the distance between the new text profile P_T and the language profile P_L is calculated as follows:

$$CT(P_T,\ P_L) = Min \begin{cases} \sum_{ng \in P_T} \left| Pos_{P_L(ng)} - Pos_{P_T(ng)} \right|. & if < ng > is\ present \\ DMAX. \quad if\ < ng > is\ absent \end{cases} \tag{6.1}$$

Where:

ng : a tri-gram

P_T, P_L: profile of the new text T, profile of the language L

$Pos_{P_T(ng)}$, $s_{P_L(ng)}$: position of the tri-gram «ng» in the profiles P_T , P_L if « ng » belongs to the language profile.

The Distance of Kullbach-Leibler (KL) [Sibun and Reynar, 1996]***:*** based on the relative entropy of Kullbach and Leibler as distance measurement. Formally, this distance measurement is calculated by the following equation:

$$KL(T_1\ , T_2) = \sum_{ng} f_2\ (ng).Log\left(\frac{f_2(ng)}{f_1(ng)}\right) \tag{6.2}$$

Where:

T_1, T_2: texts

$f_1(ng)$ $f_2(ng)$: frequencies of n-grams "ng" in texts T_1, T_2 successively. if the n-gram "ng" is absent from the text T_i a half-frequency is added to prevent the score from falling to - ω

The Distance of khi2 (χ 2) [Jalam and Teytaud, 2002]***:*** this distance is formally presented as follows:

$$\chi^2\ (T_1\ , T_2) = \sum_{ng} \left[\frac{(f_1(ng) - f_2(ng))^2}{f_2(ng)}\right] \tag{6.3}$$

With*:* $f_1(ng)$, $f_2(ng)$: frequencies of n-grams « ng » in texts T_1 , T_2 successively

(4)

$$f_i(ng) = \frac{Nb.Occurrences\ of <ng>\ in\ Ti}{Total\ of\ n-grams\ dans\ Ti} \tag{6.4}$$

6.2.4.2. Conventional methods used in categorization: Several methods exist in the field of documents categorization, their common difficulty is the very large dimension. Among these methods, we can note: decision trees (ID3, C4.5, CART, ...) [Rakotomalala, 2005] requiring dimension reduction, neural networks with back propagation, SVM and RBF methods [Joachims, 1998, Dimitrios et al., 2013].

6.3. Our proposed method

Our method is inspired from the method of [Cavnar and Trenkle, 1994] with the following differences and improvements:

[Cavnar and Trenkle, 1994] requires sorting profiles of different languages, as well as the new text profile in reverse order of frequencies before any calculation, this is not necessary for our method. which permits to save a considerable time required by sorting. The distance used in [Cavnar and Trenkle, 1994] is based on the sum of position errors between the new text tri-gram profile and the same tri-gram in the reference profile of each language if the tri-gram is present. If this is not the case, it takes a maximum value of position error. Here, we can note two disadvantages: The first is the calculation of the sum of the position error requires a huge computational effort, especially when we use corpus of large sizes. The second problem is in the choice of the maximum position error when the tri-gram is absent. Here, no method is specified. Thus to overcome the two disadvantages we propose the following method: we take each n-gram (n = 3, 4, 5) of the new text profile, and we look in the profile of each language. If this n-gram exists, we assign the value 1 to it, otherwise we assign the sum of the frequencies of all corpus n-grams, and then we calculate the sum which represents the distance. The text will be assigned to the language whose distance is minimal.

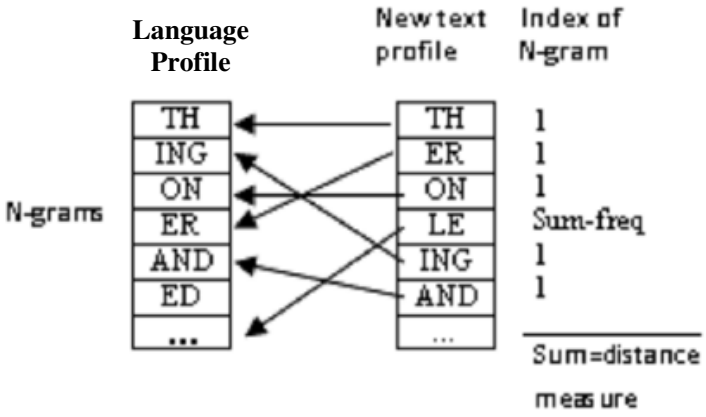


Figure 6.1. Principle of the proposed method

Formally our distance measure is calculated as follows:

$$CT(P_T, P_L) = Min \begin{cases} \sum_1^{ng \in P_T} a_i & a_i = 1 \quad if < ng > is\ present \\ sum\_freq & if < ng > is\ absent \end{cases} \tag{6.5}$$

6.4. Experimentations

6.4.1. Training and Test Corpus

We used for training a corpus that is constituted of 668 texts written in (10) languages. The test corpus is constituted of 199 texts. Texts belong to the headlines of international newspapers. We preferred to use reduced size documents to facilitate their processing and segmentation into words and n-grams. Both corpora may be distributed as follows:

Table 6.1. Training and test corpora used in experiments

Languages	Training Corpus	Test Corpus
	Number of texts	**Number of texts**
Arabic	68	17
French	80	19
English	79	20
German	74	20
Spanish	73	19
Italien	63	16
Greek	65	23
Russian	90	25
Turkish	58	18
Malysian	78	22
Total	668	199

6.4.2. Pre-processing Performed on Training and Testing corpora

Before proceeding to the phase of language identification itself, another phase of pre-processing on the training and test corpora is essential. This phase comprises the next tasks:

- Elimination of unnecessary characters (punctuation, digits, special foreign characters, abbreviations, ...).
- Conversion of uppercase to lowercase.
- Morph syntactic treatment on the text or text standardization
- Segmentation of the text into words and n-grams (*n=3, 4, 5*)
- Setting a minimum threshold s for the frequency and eliminate the n-grams whose frequencies are less than s.

6.4.3. Performed Processing

After finishing the phase of pre-processing on the training and test corpora we proceed to the following treatments:

- For the segmentation of texts, we used two approaches: bag of words, n-grams of characters.
- The results were tested for varying thresholds (*s=2, 3, 4*).
- For text segmentation into n-grams we tested results for varying values of n (*n = 3, 4, 5*).
- For the applied learning algorithm, we chose the algorithm of k-NN with k=1, as well as NB algorithm.
- We applied the 1-NN algorithm with a variety of distances such as: CT [5], KL [15], χ^2 [15], then our suggested method with the new associated distance.

6.5. Evaluation of obtained results

Table 6.2. Segmentation of the training corpus into words (s = 2)

Languages	# Texts	# gross words	#purified words	# Most freq words
Arabic	68	19010	4960	1933
French	80	14698	5757	2646
English	79	16672	5472	2761
German	74	4160	2498	681
Spanish	73	15249	5748	2498
Italien	63	3827	2350	603
Greek	65	6139	2693	688
Russian	90	5857	3728	921
Turkish	58	4292	2563	857
Malysian	78	4205	2045	744

Table 6.3. Segmentation of the training corpus into n-grams (N=3, s = 2)

Languages	# Texts	# 3g gross	# 3g purified	#3g Most freq
Arabic	68	53609	5276	3318
French	80	63956	7271	4862
English	79	61128	6796	4608
German	74	43121	6173	3650
Spanish	73	58399	6261	4138
Italien	63	34271	4523	2670
Greek	65	39279	6832	4095
Russian	90	58141	7324	4646
Turkish	58	34168	5150	3288
Malysian	78	50441	4854	3182

Table 6.4. Segmentation of the training corpus into n-grams (N=4, s = 2)

Languages	# Texts	# 4g gross	# 4g purified	#4g Most freq
Arabic	68	52707	10397	5005
French	80	63172	15998	6777
English	79	59347	15828	8813
German	74	41308	13351	5932
Spanish	73	58222	14157	7654
Italien	63	32998	10334	4902
Greek	65	37749	13690	6006
Russian	90	55932	17015	8320
Turkish	58	33025	10402	5517
Malysian	78	48309	10417	5688

Table 6.5. Segmentation of the training corpus into n-grams (N=5, s = 2)

Languages	# Texts	# 5g gross	# 5g purified	#3g Most freq
Arabic	68	52490	16162	6755
French	80	63048	24637	10946
English	79	59104	25016	10706
German	74	41028	20840	6684
Spanish	73	58108	21791	9454
Italien	63	32804	16958	5666
Greek	65	37521	19895	6401
Russian	90	55583	26971	9361
Turkish	58	32853	15477	6532
Malysian	78	48002	16939	6401

Table 6.6. Calculation of success rate and error rate for all learning algorithms (Approach: bag of words)

Algorithm	Suc_rate	Err_rate
N.Bayes	96,35	3,65
1NN with CT	98,88	1,12
1NN with KL	97,03	2,97
1NN with χ^2	98,74	1,26
The new method	99,49	0,50

Table 6.7. Calculation of success rate and error rate for all learning algorithms (Approach: n-grams)

Algorithm	N=3		N=4		N=5	
	Suc_rate	Err_rate	Suc_rate	Err_rate	Suc_rate	Err_rate
N.Bayes	97,04	2,96	98,72	1,28	98,93	1,07
1NN with CT	98,91	1,09	99,20	0,80	98,75	1,25
1NN with KL	97,89	2,11	98,14	1,86	96,37	3,63
1NN with χ^2	98,86	1,14	100	0	97,33	2,67
The new method	99,49	0,50	100	0	99,01	0,99

After the automatic pre-processing performed on both training and test corpus as it is indicated in section 6.4.2, we conducted the phase of texts segmentation into basic units, we have used for this, words, 3-grams, 4-grams, and 5-grams. The tables 6.2, 6.3, 6.4, and 6.5 summarize the obtained results in the segmentation phase. We note here that learning is applied to the most frequent tokens as it is shown in tables 6.2, 6.3, 6.4, 6.5 (column 5) because of their small number and because they always give the best success rate in the language recognition. For learning, we have also applied NB algorithm which is the best known algorithm in the field, and because it gives good results, plus a 1-NN algorithm referring to several pseudo-distances. And finally, we have applied our new method with its own pseudo-distance. The obtained results show that our method always gives better results compared to the other methods (suc_rate = 99.49%) as it is shown in table 6.7. For segmentation into 4-grams the recognition rate (suc_rate) is 100% (table 6.7, columns 4, 5). Considering the used data, especially the corpus of texts used in experiments, which we perfected arbitrarily. In addition to implementations that we made for all learning algorithms, we believe that the results are very significant and can be improved in other future works.

6.6. Conclusion and perspectives

In this chapter we treated the problem of language identification of texts in a multilingual textual corpus. We explained the two most known approaches in the field to segment a text into basic units, including: “bag of words” and “n-grams” approaches. The realized implementations show the limitation of the “bag of words” approach compared to the “n-grams” approach which is more general, independent of languages and always gives the best results. In language identification, we implemented several algorithms based on different pseudo-distances, namely: CT, KL, χ^2. In the light of the obtained results, we proposed a new method equipped with its own distance. This new method is inspired from the CT method with the advantage that this last one does not require sorting, and it's based on a simple and less expensive distance especially for corpus with large sizes. The obtained results by applying our new method are very significant, in term of computation time and in accuracy when recognizing the language of the text. Our perspective is to apply the new method on a corpus of semi-structured documents, and generalize it in the field of text categorization, as well as any other task of document classification.

CHAPTER 7

Contextual Categorization Using a New Panoply of Similarity Metrics

Chapter content

7.1. Introduction

Text classification has as objective to group thematically similar texts in a single set (group/class). The importance of such categorization approach is to organize knowledge so that they can perform some specific treatments including; information retrieval and efficient information extraction. The increasing volume of digital documents available on networks, the need for automatic classification were felt both on the internet (search engines) and within companies (classification of internal documents, dispatches of news agencies, etc ...). there exist two approaches in automatic text classification: unsupervised classification in which groups of documents are formed automatically by the machine during treatment according to similarity criteria [Salton and McGill, 1983, Iwayama and Tokunaga, 1995] and supervised classification (categorization) in which these groups are defined in advance by an expert [Joachims, 1998, Sebastiani, 2002, Yang and Liu, 1999]. In this chapter, we study the supervised classification. We use a k -NN approach based on similarity metrics known in the field of language identification. We also propose a new simple and effective method to improve the results of categorization. The chapter is organized as follows: in section 7.2 we present a state of the art about the achieved domain, in particular: text representation approaches, categorization methods and similarity metrics used in the following areas: text categorization, language identification. Section 7.3 presents clearly our contribution in this field, in particular, the application of similarity metrics used in the language identification in the field of text categorization, the proposal of a new simple and flexible method for text categorization. Section 7.4 shows the experiments carried out on training and test corpora and the obtained results. In Section 7.5 we try to evaluate the obtained results and compare them with existing works. We conclude the chapter by section 7.6 which summarizes the realized work and proposes some ideas for possible improvements and future work in the same context.

7.2. State of the Art

7.2.1. Language Identification and Documents Categorization

7.2.1.1. Language Identification: Identifying the language of a text can be defined as the assignment of this text to the language in which it is written. So this is a kind of automatic classification where classes are languages (Ar, En , Fr, ...) . This identification becomes important because of the increasing availability of textual data in different languages on the web. A real recognition of the text language is not possible if we just consider the word as a basic unit of information, it could be possible for some languages such as French or English, but very difficult for some other languages such as Arabic, German or Chinese. One alternative approach consists in segmenting texts into characteristic n-grams.

7.2.1.2. Automatic Text Categorization: Automatic text classification is a complex process whose objective is to find an efficient algorithm that permit to assign a text to one or many classes (categories, groups, labels, topics) with the highest success rate. We distinguish two types of categorization; single-label categorization in which a document belongs to exactly one category and multilabel categorization in which a document may belong to any number of categories. In the present work we focalize on the single-label categorization. Hence, a similarity measure is required to find documents relevant to a given query in information retrieval IR and to find the categories closest to a given document in text categorization TC. One application of TC is the assignment of an article to one of yahoo groups.

7.2.2. Approaches of Text Representation

In the field of text categorization, it is clear that training algorithms are not able to treat texts directly. But we must proceed to a very important preliminary phase known as representation phase. This phase generally consists of the representation of each document to categorize by a vector, whose components are for example: words, sentences, or other lexemes in order to make it exploitable by learning algorithms. As it was explained in chapter 3, there are many methods to represent a text before any processing on it including:

1. Representation with bag of words.
2. Representation with sentences.
3. Representation with lexical roots (Stems).
4. Representation with lemmas.
5. Conceptual representation.
6. Representation based on N-grams

In this study, we preferred to use only two methods, notably: the "bag of words" method and the "n-grams of characters" method because of their popularity and simplicity of implementation. In sections 7.2.2.1 and 7.2.2.2, we recall the definition of each one (for more detail about the other methods see chapter 3)

7.2.2.1. ***Representation with Bag of Words:*** It's the most known and the simplest representation of texts which was introduced with the appearance of the vector model [Salton, 1971, Salton and McGill, 1983]. In this method, all texts are transformed into vectors in which each component represents a term. Thus, terms are the words which constitute the text. We note also that with this representation, the size of the vector representing the document is equal to the size of the vocabulary which is generally very large and often consists of several tens of thousands of words. However, the great dimension of these data lets the majority of classification algorithms difficult to apply, in addition, the representation of textual data is typically hollow [Young-Min and al., 2008]. To solve this problem, several statistical techniques are used to reduce the size of vector space representing texts and thus to increase the efficiency of this method of representation and consequently to improve the quality of the obtained categorization.

7.2.2.2. Representation Based on N-grams: An n-gram of X is defined as a sequence of N consecutive X. X can be a character or a word [Nicolas, 2008]. An n-gram of character is thus a consecutive sequence of N characters [Shannon, 1948, Cavnar and Trenkle, 1994] which cannot be ordered. (e.g., the 3-grams of the sentence "Hello Sir" are: "Hel, ell, llo, lo_, o_S, _Si, Sir [Miller and al., 1999, Biskri and Delisle, 2001, Jalam and Teytaud, 2002]). The n-grams profile of a document consists of the list of the most frequent in the reverse order of their frequencies. The approach of text segmentation into characteristic n-grams has several advantages, in particular:

- Avoid the use of lemmatization and stemming phase on the text which requires an algorithmic and linguistic effort.
- Tolerant to spelling, typing, and OCR mistakes.
- This approach operates independently of languages.

- Segmentation into words is difficult for some languages. E.g., in Arabic the names and additional subjects are in some cases attached to verbs and the string is thus a sentence like: (katabtouhou) (I wrote it).

7.2.3. Text Categorization Methods

7.2.3.1. Conventional Methods: Several methods exist in the field of text categorization, their common difficulty is the very large dimension. Among these methods, we can note: decision trees (ID3, C4.5, CART, ...) [Rakotomalala, 2005], neural networks with back propagation [Wiener, 1993, Wiener *et al.*, 1995, Schütze *et al.*, 1995], SVM and RBF methods [Joachims, 1998, Dimitrios and al., 2013].

7.2.3.2. Nearest Neighbors Methods: as we have seen in the field of language identification presented in chapter 6. A large variety of learning algorithms used in the field of text categorization are based on the concept of distance or similarity. The principle is always the same; it is to find the text of the training set, which is nearest in distance to the new text to classify and to assign its category to the new text. We can also increase the number of k texts nearest to the text to classify if that is necessary. In this case, the category of the new text is the same as the majority of their k nearest neighbors (the majority category). The main challenge for these methods is how to define a metric of similarity. Practically, we can find many distances, the most used are: The scalar product (Inner product), the Euclidean distance, the cosine distance, Manhatann, dice, jaccard,… and others.

7.2.4. Similarity Metrics Used in Language Identification

As the majority of the learning algorithms used in the field of the language identification are based on the concept of distance or similarity as it was mentioned in section 6.2.4.1 of chapter 6 and the previous section of this chapter. Then, the choice of such distance is a common difficulty for these algorithms. The researchers in this domain proposed a new collection of pseudo-distances, whose use had given very satisfactory results. Among these pseudo-distances we can note the following:

- The distance of Beesley.
- The distance of Cavnar and Trenkle CT.
- The distance of Kullbach-Leibler KL.
- The distance of $\chi 2$, ...

In the present work, we have exploited the capacities provided by Cavnar and Trenkle method and its associated pseudo-distance, especially when modeling and developing our proposed solution. So, it is important to recall the main steps of this method:

1. Establish a tri-gram profile P_L for each language (reference profile).
2. Build a tri-gram profile P_T for the new text to identify the language.
3. Calculate the distances between the profile P_T and the reference profile of each language.
4. The language of the new text is that for which the distance is minimal.
5. The calculation of distance is based on the sum of position errors (difference in ranks or out-of-place) between each tri-gram in P_T and the same tri-gram in P_L if the tri-gram is present. Otherwise, it takes a maximum value of rank.

Formally, the distance between the new text profile P_T and the language profile P_L is calculated as follows:

$$CT(P_T,\ P_L) = Min \begin{cases} \sum_{ng \in P_T} \left| Pos_{P_L(ng)} - Pos_{P_T(ng)} \right|. & if\ < ng > is\ present \\ DMAX. \quad if\ < ng > is\ absent \end{cases} \tag{7.1}$$

Where:

ng : a tri-gram

P_T, P_L : profile of the new text *T*, profile of the language *L*

$Pos_{P_T(ng)}$, $Pos_{P_L(ng)}$: position of the tri-gram «ng» in Profiles P_T, P_L si « ng » belongs to the language profile.

7.3. Our Proposed Approach

7.3.1. Application of Language Identification Metrics in Text Categorization

Research on text categorization generally uses the similarity measurements mentioned in section 7.2.3.2. In our work we used a new panoply of distances which are used particularly in the field of the language identification. The experimental results obtained in this field show the effectiveness of this projection in term of success rate and simplicity of implementation.

7.3.2. Presentation of the New Method

Our method is inspired from the method of [Cavnar and Trenkle, 1994] with the following differences and improvements:

- [Cavnar and Trenkle, 1994] have applied their method to identify the language of a text, For us, we applied it in contextual categorization of texts (topical TC).
- [Cavnar and Trenkle, 1994] in their method require sorting profiles of different languages (categories in our case), as well as the profile of the new text to be classified according to the reverse order of frequencies before any calculation, this is not necessary for our method, which permits to save a considerable time required by sorting.
- [Cavnar and Trenkle, 1994] work only with trigrams (*n=3*), although our method is more general (*n=3, 4, 5, …*).
- The calculation of distance used in [Cavnar and Trenkle, 1994] is based on the sum of position errors between the new text trigram profile and the same trigram in the reference profile of each language if the trigram is present. If this is not the case, the distance takes a maximum value of position error. Here, we can note two disadvantages: The first is the calculation of the sum of the position error requires a huge computational effort, especially when we use corpus of a large size. The second problem is in the choice of the maximum position error when the trigram is absent. Here, no method is specified. Thus to overcome the two disadvantages we propose the following method: we take each n-gram (*n = 3, 4,5*) of the new text profile, and we look in the profile of each language. If this n-gram exists, we assign the value 1 to it, otherwise we assign the sum of the frequencies of all n-grams in the corpus, and then we calculate the sum which represents the distance. The text will be assigned to the language whose distance is minimal. Formally our distance measure is calculated as follows:

$$CT(P_T, P_L) = Min \begin{cases} \sum_1^{ng \in P_T} a_i \quad a_i = 1 & if < ng > is\ present \\ sum\_freq & if < ng > is\ absent \end{cases} \tag{7.2}$$

7.4. Experimentations

7.4.1. Training and Test Corpus

We used in our experiments the Reuters21578 corpus whose documents are issued of the dispatches of international newspapers. The original corpus gathers 116 categories (93 for another version of the same corpus) for which the first ten (10) categories are bulkiest. For this purpose, and in order to facilitate the implementation of learning algorithms and to reduce the computing times, we used only the first ten (10) categories (Acq, Corn, Crude, Earn, Grain, Interest, Moneyfx, Ship, Trade, Wheat), with a total of 7193 documents for the training set and 2747 documents for the test set. Another correction on the corpus which was also necessary is the elimination of documents having big sizes; we have to leave only reasonable size documents (between 1 and 3ko). Fortunately, the number of these documents is not important (a few tens) and thus does not influence the obtained results. The two corpora of training and test can be presented as follows:

Table 7.1. Training and test corpora used in experiments

Categories	Training Corpus	Test Corpus
	Number of texts	***Number of texts***
ACQ	1650	719
CORN	181	53
CRUDE	389	187
EARN	2877	1086
GRAIN	433	146
INTEREST	347	121
MONEYFX	538	166
SHIP	197	88
TRADE	369	112
WHEAT	212	69
Total	7193 (72 %)	2747 (28 %)

7.4.2. Pre-processing Performed on Training and Test corpus

Before proceeding to the phase of categorization itself, another phase of pre-processing on the training and test corpora is very important. This phase comprises the following tasks:

- Elimination of unnecessary characters (punctuation, digits, special foreign characters, abbreviations, etc).
- Conversion of uppercase to lowercase.
- Morphosyntactic processing on the text (text standardization).
- Segmentation of texts into words and n-grams (*n=3, 4, 5*).
- Setting a minimum frequency threshold *s* and eliminate the n-grams whose frequencies are less than *s*.

7.4.3. Performed Processing

After finishing the phase of pre-processing on the training and test corpus, we proceed to the following treatments:

- For the segmentation of texts, we used two approaches: Bag of words, n-grams of characters.
- The results were tested for varying thresholds (s=2, 3, 4).
- For text segmentation into n-grams we tested results for varying values of n (n = 3, 4, 5).
- For the applied learning algorithm, we chose the algorithm of k-NN with *k=1*, as well as NB algorithm.
- We applied the 1-NN algorithm with a variety of distances such as: C&T, KL, χ^2, then our suggested method with the new associated distance.

7.5. Evaluation of the Obtained Results

7.5.1. Segmentation Phase (Tokenization)

In this phase, we segmented the texts of the used corpora (the training and the test corpora) into basic units called "tokens". For this purpose, we have used two well known approaches: bag of words and n-grams of characters. The main goal of such segmentation is to facilitate the transformation of each text into a numeric vector as we have explained previously. The obtained results of this phase are summarized in the following tables:

Table 7.2. Segmentation of the training corpus into words (S = 3)

Categories	# Texts	# gross words	# purified words	# Most freq.words
ACQ	1650	131214	15321	5998
CORN	181	20854	4077	1406
CRUDE	389	53214	7833	3147
EARN	2877	159900	14091	4545
GRAIN	433	47166	6945	2761
INTEREST	347	33441	4842	1947
MONEYFX	538	60585	6947	3032
SHIP	197	21337	5123	1763
TRADE	369	56513	7250	3058
WHEAT	212	22738	4395	1575

Table 7.3. Segmentation of the training corpus into n-grams (N=3, S = 3)

Categories	# Texts	# 3g gross	# 3g purified	#3g Most.Freq
ACQ	1650	927168	20034	12635
CORN	181	118180	8920	4731
CRUDE	389	264286	11976	7177
EARN	2877	1180274	20653	13186
GRAIN	433	282329	12205	7194
INTEREST	347	180078	8817	5213
MONEYFX	538	310976	10411	6542
SHIP	197	134944	10025	5441
TRADE	369	260005	10249	6400
WHEAT	212	137411	9285	5051

Table 7.4. Segmentation of the training corpus into n-grams (N=4, S = 3)

Categories	# Texts	# 4g gross	# 4g purified	#4g Most.Freq
ACQ	1650	844884	60343	28259
CORN	181	111600	19936	7357
CRUDE	389	252769	30632	13456
EARN	2877	1009809	63970	30989
GRAIN	433	264300	31274	13240
INTEREST	347	165644	20120	8801
MONEYFX	538	290819	25862	12213
SHIP	197	126412	25969	8990
TRADE	369	251595	23069	12191
WHEAT	212	127626	21052	8032

Table 7.5. Segmentation of the training corpus into n-grams (N=5, S = 3)

Categories	# Texts	# 5g gross	# 5g purified	#5g Most.Freq
ACQ	1650	834098	122143	41571
CORN	181	110615	33084	8544
CRUDE	389	251161	55801	18036
EARN	2877	987806	129798	42163
GRAIN	433	261712	57414	17459
INTEREST	347	163864	34901	11303
MONEYFX	538	288262	47448	17233
SHIP	197	125227	38675	10604
TRADE	369	250443	48131	160855
WHEAT	212	126360	35281	9349

7.5.2. Learning phase

This phase consists in applying some learning algorithms on the vectors representing the texts of the training corpus. This task allows to build a model to predict the category of any unlabeled text. For this purpose, we have used many algorithms such as: Naïve bayes algorithm, kNN algorithm with a variety of pseudo-disatnce and our proposed algorithm.

After building a predictive model, we apply it on the texts of the test corpus. The objective is to measure the performance of this model with each algorithm using two parameters: the success rate and the error rate. The obtained results in this phase are showed in the following tables:

Table 7.6. Calculation of success rate and error rate for all learning algorithms (Approach: bag of words)

Algorithm	Suc_Rate	Err_Rate
N.Bayes	83,76	16,24
1NN with CT	82,92	17,08
1NN with KL	83,87	16,13
1NN with χ^2	82,81	17,19
SVM (RBF)	55,36	44,64
SVM (SIG)	47,57	52,43
The new method	87,57	12,43

Table 7.7. Calculation of success rate and error rate for all learning algorithms (Approach: n-grams)

Algorithm	N=3		N=4		N=5	
	Suc_Rate	*Err_Rate*	*Suc_Rate*	*Err_Rate*	*Suc_Rate*	*Err_Rate*
N.Bayes	80,27	19,73	84,91	15,09	85,73	14,27
1NN with CT	66,12	33,88	72,05	27,95	78,66	21,34
1NN with KL	67,23	32,77	74,19	25,81	83,01	16,99
1NN with χ^2	68,27	31,73	77,69	22,31	84,44	15,56
SVM (RBF)	48,34	51,66	55,84	44,16	54,82	19,73
SVM (SIG)	52,74	47,26	55,4	44,60	54,67	45,33
The new method	73,45	26,55	82,46	17,54	88,19	11,81

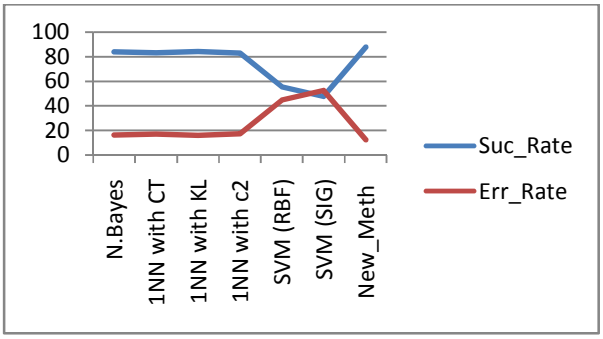


Figure 7.1. Suc_Rate and Err_Rate (App.Sac of words).

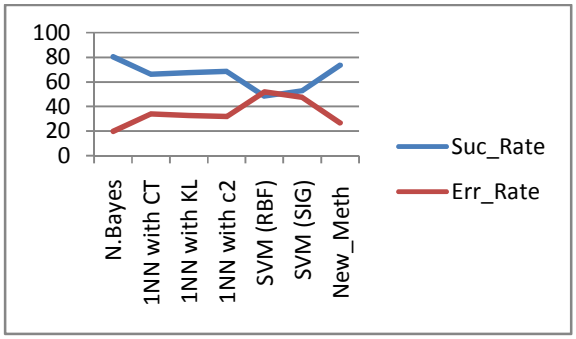


Figure 7.2. Suc_Rate and Err_Rate (App.n-grams, n=3)

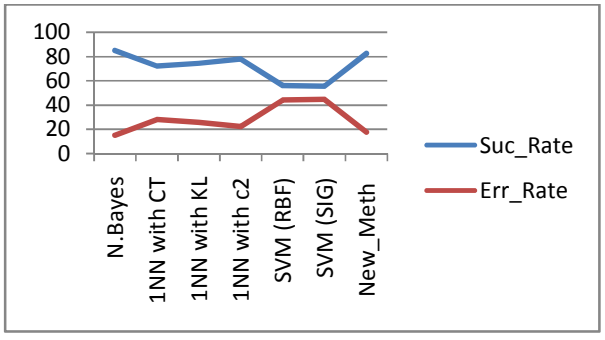


Figure 7.3. Suc_Rate and Err_Rate (App.n-grams, n=4).

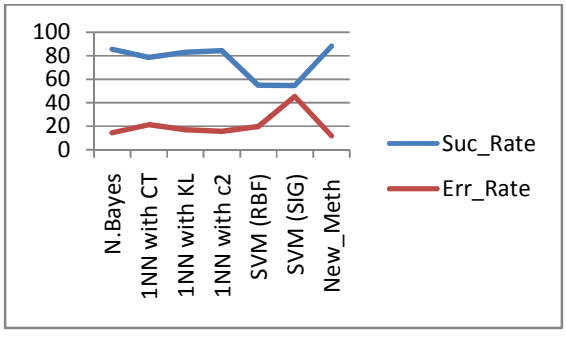


Figure 7.4. Suc_Rate and Err_Rate (App.n-grams, n=5).

7.5.3. Interpretation of the obtained results

First, we applied an automatic pre-processing on both training and test corpus as it is explained in section 7.4.2. Then, we began the phase of texts segmentation into basic tokens (words, 3-grams, 4-grams, and 5-grams). The results of this phase are summarized in tables 7.2, 7.3, 7.4, and 7.5. after that, we applied some learning algorithms on the most frequent tokens as it is shown in tables 7.2, 7.3, 7.4, 7.5 (column 5) because of their small number and because they always give the best success rate in text classification. For learning, we have chosen: NB algorithm which is the best known algorithm in the field, and because it gives good results, SVM with RBF and sigmoid kernels (LIBSVM) plus a 1-NN algorithm referring to several pseudo-distances. And finally, we have applied our new method with its own pseudo-distance. The obtained results showed that our method always gives better results compared to the other methods (Suc_Rate > 70 %). For segmentation into 5-grams, the success rate in categorization is 88% (Table 7.7, columns 6, 7). Considering the used data especially the corpus of texts used in experiments. In addition to implementations that we made for all learning algorithms, we believe that the results are very significant and can be improved in other future works.

7.6. Conclusion and perspectives

This chapter treats the problem of text categorization in a heterogeneous textual corpus. We presented the approaches used to segment texts into basic units, notably: the “bag of words” approach and the n-grams” approach. The realized implementations show the limitation of the “bag of words” approach compared to the “n-grams” approach which is more general, and always gives the best results. For the thematic categorization, we implemented several algorithms based on different pseudo-distances, namely: CT, KL, χ^2. This panoply of pseudo-distance are often used in the field of language identification and gave satisfactory results. In the light of the obtained results, we applied our proposed method equipped with its own distance. We recall that this method is inspired from the CT method as it was presented in chapter 6. It has many advantages such as: it does not require sorting, and it's based on a simple and less expensive distance especially for corpus with large sizes. The obtained results by applying our new method are very significant, in term of computation time and in accuracy when categorizing texts. Our perspective is to apply the new method on a corpus of semi-structured documents, and generalize it in other tasks of text categorization like: text summarizing, author recognizing, contextual categorization and author recognizing in the same time.

CHAPTER 8

Arabic Text Categorization: An Improved Stemming Algorithm to Increase the Quality of Categorization

Chapter content

8.1. Introduction

Arabic is one of the oldest and the most used language in the world. It is spoken by over 300 million people in the Arabic world, and used by more than 1.7 billion Muslims over the world because it is the language of the Holy Quran. Here we can distinguish two types of Arabic; a more classical language, as found in the Holy Quran or poetry, a standardized modern language, and regional dialects [Abu Hawas and Keith, 2014]. We note also that the Arabic language is a Semitic language [Ghazzawi, 1992], based on 28 cursives letters written from right to left. The word in Arabic is formed of the root part and some affixes (antefixes, prefixes, infixes, suffixes) that form the word (سألتمونيها Saaltmwnyha). The Arabic root extraction is a very difficult task. This is not the case for other languages as English or French, because Arabic is a very rich language with a very difficult structure and complex morphology. Arabian linguists show that all nouns and verbs of the Arabic language are derived from a set of roots containing about 11347 roots; more than 75 % of them are trilateral roots [Al-Kamar, 2006]. There are many applications based on the roots of words in Arabic processing such as: text's classification, text summarizing, information retrieval, data and text mining. [Ghwanmeh, 2009, Yousef et al, 2010]. The Arabic word's roots can be classified according to the vowel letters (ي ، و ، أ a, w, y) into two types [Gaafar et al., 2007]. The first one is called the strong roots that do not contain any vowel (فتح، خرج، ذهب go, come out , open), the second one is called vocalic roots that contain at least one vowel (وعد ، أوى shelter, promise). Arabic roots can be further classified according to the number of their characters into four types: Trilateral roots which form most words in the Arabic language [Al-Kamar, 2006] (e.g., علم ، كتب ، خرج know, write, come out), Quadrilateral roots(e.g., دحرج طمأن ، roll, assure), Quinquelateral roots (e.g., ، انطلق ، اقتصد ، انكسر broken, economize, start) and Hexalateral roots (اقشعّر ، استحسن ، استعمل use, enjoy, tremble). There are two classes of methods used to extract the roots of Arabic words. The first class is based on morphological rules. So its methods simulate the same process as that of an expert linguist during his analysis of a given Arabic word [Abu Hawas and Keith, 2014, Al-Omari et al., 2013, Al-Shalabi, 2005, Khodja and Garside, 1999, Momani and Faraj, 2007], which makes the process of extracting a root difficult and complex because of the diversity of morphological formulas and the multiplicity of word forms for the same root when changing the original characters position in the word (e.g., علم ، عالم ، علوم ، عوالم ، معالم know, scientist, sciences, worlds, landmarks) [Al-Nashashibi, 2010a, Hajjar et al., 2010]. The second class is formed of statistical methods which are simple, fast, and do not require any morphological rules but some calculations. [Ababneh et al., 2012, Al-Nashashibi, 2010b, Al-Shalabi et al., 2003, Al-Shalabi et al., 2008, Duwairi, 2007, Ghwanmeh, 2005, Kanaan et al., 2005].

In this chapter, we propose an improved statistical algorithm which permits to build an Arabic stemmer based on the extraction of words' roots and the approach of n-grams of characters without using any morphological rule. The chapter is organized as follows: the second section presents some related works, so we review some papers that treat the problem of extraction of Arabic words' roots. In the third section we introduce our new algorithm. The fourth section presents the experiments that we have done to test our new method and also displays the obtained results. In the fifth section we established a comparison between our proposed algorithm and two other algorithms notably: Khodja algorithm and Yousef et al algorithm. The sixth section presents a discussion of the obtained results and. In the last section we conclude our work with a summary and some ideas to improve it in the future.

8.2. Related Work

Many researchers proposed some algorithms to extract Arabic words roots, some of these algorithms are based on morphological rules. Thus, they are called morphological methods. Others do not use any morphological rule but some statistical calculations, so they are called statistical algorithms. In the first class of algorithms, we can note the following:

[Khodja, 1999, Kodja and Garside, 1999] root extractor removes the longest suffix and the longest prefix. It then, matches the remaining word with verbal and noun patterns, to extract the root. The roots extractor makes use of several linguistic data files such as a list of all diacritics, punctuation characters, definite articles, and stop words [Larkey and Connell, 2001, Larkey and Ballesteros, 2002, Larkey and al, 2007, Sawalha and Atwell, 2008].

[Al-Nashashibi et al, 2010b] proposed a linguistic approach for root extraction as a pre-processing step for Arabic text mining. The proposed approach is composed of a rule-based light stemmer and a pattern-based infix remover. They propose an algorithm to handle weak, eliminated-long vowel, hamzated and geminated words. The accuracy of the extracted roots is determined by comparing them with a predefined list of 5,405 trilateral and quadrilateral roots. The linguistic approach performance was tested on in-house texts collection consisting in eight categories, the author achieved a success ratio about 73.74%.

[Abu Hawas, 2013] presented a new Arabic root extraction algorithm that tries to assign a unique root for each Arabic word without having an Arabic roots list, a words patterns list, or the Arabic words' prefixes and suffixes list. The algorithm predicts the letter positions that may form the word root one by one, using rules based on the relations between the Arabic word letters and their placement in the word. This algorithm consists in two parts. The first part gives the rules that distinguish between the Arabic definite letter "الـ AL, La" and the original word letters "الـ". The second part segments each word into three parts and classifies its letters according to their positions. The author tested her proposed algorithm using the Holy Quran words and obtained an accuracy of 93.7% in the root extracting process.

In the second class of algorithms, we can note the following:

[Al-shalabi et al, 2003] Developed a root extraction algorithm which does not use any dictionary. Their algorithm categorizes all Arabic letters according to six integer weights, ranging from 0 to 5, as well as the rank of the letter which is determined by the position this letter holds in a word. The weight and rank are multiplied together, and the three letters with the smallest product constitute the root of the word. We note that [Al-shalabi et al, 2003] did not explain or clarify why or on which basis did he use such ranking or weighting.

[Momani and Faraj, 2007] Proposed an algorithm to extract tri-literal Arabic roots, this algorithm consists in two steps; in the first step, they eliminate stop words as well prefixes and suffixes. In the next step, they remove the repeated words' letters until only three letters are remain. Then they arrange these remaining letters according to their order in the original word, which form the root of the original word. The obtained results of this algorithm were very promising and give an accuracy of root's extraction over 73%.

[Hmeidi et al., 2010] Proposed a new way to extract the roots of Arabic words using n-grams technique. They used two similarity measures; the dissimilarity measurement, or the "Manhattan distance measurement" and the "Dice's measurement". They tested their algorithm on the Holy Quran and on a corpus of 242 abstracts from the Proceedings of the Saudi Arabian National Computer Conferences. They concluded from their study that combining the n-grams with the Dice's measurement gives better results than using the Manhattan distance measurement.

[Boudlal et al, 2011] Proposed a new algorithm to find the system that assigns, for every non vowel word a unique root depending on the context of the word in the sentence. The proposed system consists in two modules; the first one consists in analysing the context by segmenting the words of the sentence into its elementary morphological units in order to extract its possible roots. So, each word is segmented into three parts (prefix, stem and suffix). In the second module, they based on the context to extract the correct root among all possible roots of the word. For this purpose, they used a Hidden Markov Models (HMM) approach, where the observations are the words and the possible roots are the hidden states. They validate their algorithm using NEMLAR Arabic writing corpus that consists in 500,000 words, and their proposed algorithm gives the correct root in more than 98% of the training set and 94% of the testing set.

[Yousef et al, 2014] proposed a new algorithm which uses the n-grams technique. An n-gram is a basic text analysis tool that is used in natural language processing. In this technique, both the word and its assumed root are divided into pairs (called bi-gram, or di-gram) then the similarity between the word and the root is calculated using equation (1) [Farkes, 1992]. This process is repeated for each root in the roots list:

$$S = 2 * C/(A + B) \quad (8.1)$$

Where:

A = Number of unique bi-grams in the word (A)

B = Number of unique bi-grams in the root (B)

C = Number of similar unique pairs between the word (A) and the root (B)

To use equation (1) for extracting the word's root, we must have: the word (A) and the potential roots (B) to compare with, then the similarity measuring is conducted by computing the value of (S) between the word (A) and each potential roots (B).

8.3. The Proposed Algorithm

In our new algorithm, we also use the n-grams technique to extract Arabic words roots. For this purpose, we proceed according to the following steps:

Step 1: we segment the word for which we want to find the root, and all the roots of the list into bigrams (2-grams).

For example if we have the word "يذهبون" and a list of six (06) roots (فتح ، خرج ، ذهب ، وجد ، وهب ، نهب), we proceed to the segmentation step as follows:

W = "يذهبون" ➔ (يذ، يه ، يب ، يو ، ين ، ذه ، ذب ، ذو ، ذن ، هب ، هو ، هن ، بو ، بن ، ون)

R_1 = "فتح" ➔ (فح ، تح ، فت)

R_2 = "خرج" ➔ (خر ، خج ، رج)

R_3 = "ذهب" ➔ (ذه ، ذب ، هب)

R_4= "وجد" ➔ (وج ، ود ، جد)

R_5 = "وهب" ➔ (وه ، وب ، هب)

R_6 = "نهب" ➔ (نه ، نب ، هب)

Step 2: we calculate the following parameters:

N_W : The number of bigrams in the word *w*

N_{R_i} : The number of bigrams in the root *Ri*

N_{WR_i} : The number of common bigrams between the word *W* and the root *Ri*

$N_{W\bar{R}_i}$: The number of bigrams belonging to the word w and do not belong to the root *Ri.*
($N_{W\bar{R}_i}$=N_W − N_{WR_i})
$N_{R_i\bar{W}}$: The number of bigrams belonging to the root *Ri* and do not belong to the word *w.*
($N_{R_i\bar{W}}$= N_{R_i} - N_{WR_i})

For the previous example we have:
N_W=15, N_{R_1}=3, N_{R_2}=3, N_{R_3}=3, N_{R_4}=3, N_{R_5}=3, N_{R_6}=3, N_{WR_1}=0, N_{WR_2}=0, N_{WR_3}=3, N_{WR_4}=0, N_{WR_5}=1, N_{WR_6}=1, $N_{W\bar{R}_1}$=15, $N_{W\bar{R}_2}$=15, $N_{W\bar{R}_3}$=12, $N_{W\bar{R}_4}$=15, $N_{W\bar{R}_5}$=14, $N_{W\bar{R}_6}$=14, $N_{R_1\bar{w}}$=3, $N_{R_2\bar{w}}$=3, $N_{R_3\bar{w}}$=0, $N_{R_4\bar{w}}$=3, $N_{R_5\bar{w}}$=2, $N_{R_6\bar{w}}$=2.

Step3: we take only the roots having at least one common bigram with the word w (≥ 1) as candidate roots among the list of all roots in order to reduce the calculation time.
In our previous example, we can take only the roots: R_3 = "ذهب" , R_5 = "وهب" , R_6 = "نهب" with = 3, 1, 1 respectively.

Step4: we calculate the distance *D(w, Ri)* between the word W and each candidate root *Ri* (R_3, R_5, R_6) according to the following equation :

$$D(w, R_i) = 2 * N_{wR_i} + k * N_{w\bar{R}_i} + k * N_{R_i\bar{w}} \quad (8.2)$$

Where: *k* is a constant which must take a high value (we put here *k=100*)
For the previous example we obtain:
*D(w, R_3) = 2*3+100*12+100*0 = 1206*
*D(w, R_5) = 2*1+100*14+100*2 = 1602*
*D(w, R_6) = 2*1+100*14+100*2 = 1602*

Step5: in the last step, we assign the root that has the lowest value of distance *D(w, Ri)* among the candidate roots to the word *W*. It is the required root.
In our example, the root of the word "يذهبون" is "ذهب"

Finally, we note that our new algorithm has the following advantages:

1. Does not require the removal of affixes whose distinction from the native letters of the word is quite difficult.
2. Works for any word whatever the length of the root, i.e., 3-lateral, 4-drilateral, 5-lateral, 6-lateral roots.
3. Valid for strong and vocalic roots which generally pose problems in Arabic during their derivation, because of the complete change of their forms.
4. Does not use any morphological rule nor patterns but only simple calculations of distances.
5. Very practical algorithm and easy to implement on machine.

8.4. Experimentations and Obtained Results

8.4.1. The Used Dataset

To validate our proposed algorithm, we used three corpuses which can be classified according their sizes into: small corpus, middle corpus, and large corpus (see table 8.1). Each one is constituted of many files as indicated below:

1. The file of derived forms (gross words) which contains morphological forms of words derived from many Arabic roots.
2. The file of roots which contains many Arabic roots. We note that these roots are trilateral, quadrilateral, quinquelateral, and hexalateral. We also note that many of them are vocalic roots which contain at least one vowel (see table 8.2).
3. The file of golden roots which contains the correct roots of all words present in our corpus. This golden list was prepared by an expert linguist and used as a reference list, i.e., by comparing the list of obtained roots (extracted by the system) and the reference list (established by the expert), we can calculate the roots extraction accuracy (success ratio). The extraction process and the obtained results are shown in tables 8.3 and 8.4 as well as figures 8.1 and 8.2.

Table 8.1. Corpus used in experiments.

Corpus	Size of derived words' file	Size of the roots' file	Size of the golden roots' file
Small corpus	50	25	50
Middle corpus	270	135	270
Large corpus	2250	600	2250

Table 8.2. A sample of 3-lateral, 4-lateral, 5-lateral, 6-lateral roots.

Trilateral roots	Quadrilateral roots	Quinquelateral roots	Hexalateral roots
زرع	أكرم	انطلق	استعمل
صنع	أعان	انكسر	استحسن
تجر	أعطى	احتوى	استعان
أبى	علّم	اجتمع	اعشوشب
نفر	ربّى	اخضرّ	ادهامّ
على	برّأ	تقدّم	اخضارّ
دار	قاتل	تحطّم	اجلوذّ
طار	حاسب	تحدّى	احرنجم
عطس	داين	تعاطى	افرنقع
صدع	زلزل	تدحرج	اطمأنّ

8.4.2. The Obtained Results

After applying our proposed stemming algorithm for Arabic language, we obtained the results showed in the following tables:

Table 8.3. Extraction of some Arabic roots using our new algorithm.

Word	Nearest roots	Nb.Common bi-grams	Distance values	Extracted root	Correct root
يتعلمون	كَلم ، عالج ، علم ، عمل ، كمن	3 ، 1 ، 3 ، 2 ، 1	2806 ، 3202 ، **2506** ، 2704 ، 2902 ،	علم	علم
كاتب	اقتصد ، كتب	1 ، 3	1402 ، **306**	كتب	كتب
معلم	كَلم ، عالج ، علم ، عمل ، كمل	3 ، 1 ، 3 ، 3 ، 1	1006 ، 1402 ، **706** ، 708 ، 1102	علم	علم
كتاتيب	اقتصد، كتب، تأتأ	1 ، 3 ، 1	2002 ، **906** ، 1402	كتب	كتب
اقتصاد	قصد ، اقتصد ، عقد	3 ، 10 ، 1	1106 ، **420** ، 1502	اقتصد	اقتصد
يقصدون	قصد ، اقتصد ، عقد	3 ، 3 ، 1	**1206** ، 1906 ، 1602	قصد	قصد
علاج	عالج ، علم ، عمل	5 ، 1 ، 1	**210** ، 702 ، 702	عالج	عالج
معالجة	عالج ، علم ، عمل ، كمل	6 ، 1 ، 2 ، 1	**912** ، 1602 ، 1404 ، 1602	عالج	عالج
استخدم	اقتصد ، خمد ، خدم	3 ، 2 ، 3	1906 ، 1404 ، **1206**	خدم	خدم
خادم	اقتصد ، خمد ، خدم	1 ، 2 ، 3	1402 ، 504 ، **306**	خدم	خدم
كمون	كلم ، كمل، كمن	1 ، 1 ، 3	702 ، 702 ، **306**	كمن	كمن
سنستدرجهم	اقتصد ، خدم ، درج ، هزم	1 ، 1 ، 3 ، 1	3802 ، 3102 ، **2706** ، 3102	درج	درج
متذبذب	كتب ، ذبذب	1 ، 4	1002 ، **508**	ذبذب	ذبذب
هزائم	هزم	3	**706**	هزم	هزم
يهزمونهم	كمن ، هزم	1 ، 3	2006 ، **2402**	هزم	هزم
المربّون	كلم ، عالج ، علم ، كمن ، ربّى ، طار	1 ، 1 ، 1 ، 1 ، 3 ،1	2902 ، 3202 ، 2902 ، 2902 ، **2806** ، 2902	ربّى	ربّى
طيران	طار	2	**904**	طار	طار
طائرات	اقتصد ، طار	3 ، 1	2102, **1006**	طار	طار

Table 8.4. Obtained results when extracting the words roots.

Corpus	Nb.Roots	Nb.Words	Correct Results	Wrong Results	Success Rate(%)	Error Rate (%)
Small	25	50	49	1	98,00	2,00
Middle	135	270	253	17	94,07	5,93
Large	600	2250	2028	222	90,13	9,87

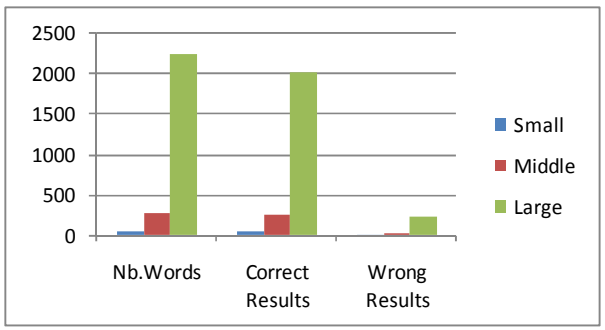


Figure 8.1. Correct and wrong results in number of words.

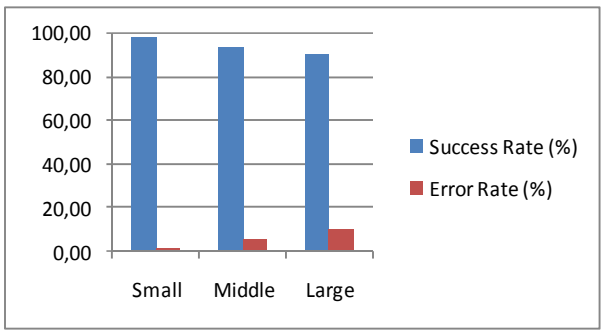


Figure 8.2. Calculation of success rate and error rate.

8.5. Comparison with Other Algorithms

To show the effectiveness of our proposed algorithm, we concluded our work by establishing a comparison with other known algorithms. For this purpose, we took a sample words list and tried to extract the root of each word using three well-known algorithms which are: Khodja stemmer, Yousef et al. stemmer, and our proposed stemmer, the obtained results were shown in table 8.5. On the other hand, we illustrated the obtained results when applying the three above algorithms on the three corpuses used in the experimentation, namely: the small corpus, the middle corpus, and the large corpus, then, we summarized the obtained accuracy for each algorithm in table 8.6 and figure 8.3.

Table 8.5. Extraction of some roots using the three algorithms.

Word	Extracted Root			
	Khodja Algorithm	**Yousef N Algorithm**	**Our new Algorithm**	**Corr. root**
يتعلّمون	علم	علم	علم	علم
كاتب	كتب	كتب	كتب	كتب
كتاتيب	Not stemmed	كتب	كتب	كتب
اقتصاد	قصد	اقتصد	اقتصد	اقتصد
سنستدرجهم	Not stemmed	درج	درج	درج
متلألئ	Not stemmed	لألأ	لألأ	لألأ
المربّون	ربن	ربّى	ربّى	ربّى
طائرات	طور	طار	طار	طار
ولولة	ليل	ولول	ولول	ولول
وقيعة	قوع	وقع	وقع	وقع
يزنونهم	زنن	نهب	وزن	وزن
زلازل	Not stemmed	تنازل	زلزل	زلزل
حواسيب	Not stemmed	نسي	حسب	حسب
ناسج	نسج	سجد	نسج	نسج
نوازل	نزل	تنازل	نزل	نزل

Table 8.6. Illustration of obtained accuracy for the three algorithms.

Corpus	Size		The obtained accuracy (suc_ rate, err_ rate)%					
	Nb.roots	*Nb.words*	*Khodja Algorithm*		*Yousef N Algorithm*		*Our new Algorithm*	
Small	25	50	68,00	32,00	92,00	8,00	98,00	2,00
Middle	135	270	83,70	16,30	63,33	36,66	94,07	5,93
Large	600	2250	68,43	31,57	61,32	38,68	90,13	9,87

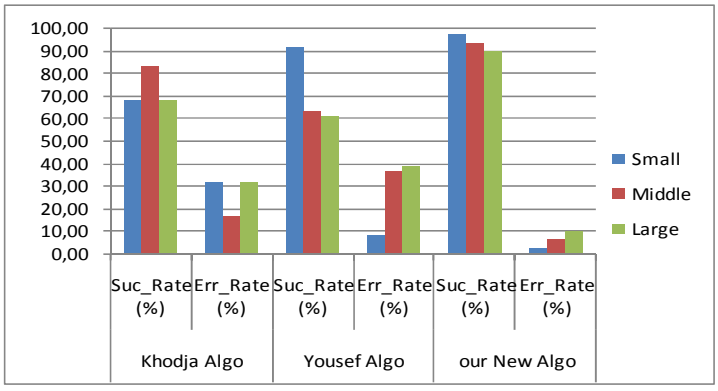


Figure 8.3. Comparison between three algorithms.

8.6. Discussion

From table 8.5, we see that Khodja algorithm fails sometimes in getting the correct root of the given word and for many words it produced one of two results: (1) not stemmed (i.e., حواسيب , سنستدرجهم) (2) a new word and sometimes a wrong word that does not exist in Arabic (i.e.,(قوع ، وقيعة) (طور ، طائرات)). The same thing can be said for Yousef et al. algorithm. Although it gives better results than Khodja algorithm, it fails for many words like: ((سجد، ناسج) ,(زنن، يزنونهم)). For the same cases, our algorithm always gives the correct root and the failure is very limited. From Table 8.6 and figure 8.3, we can deduce that our proposed algorithm gives the best results for the three used corpuses with a very high accuracy. We note here the value 98 % for the small corpus, 94,07 % for the middle corpus, and 90,13 % for the large corpus.

8.7. Conclusion and Perspectives

In this chapter we have studied how we can reduce the size of terms in Arabic text categorization by stemming. For this purpose, we exposed the most known algorithms in the field, including morphological algorithms mainly based on the use of morphological rules of Arabic, and statistical algorithms which are the newest in the field, and require only simple calculations of distances. We also proposed a new statistical algorithm based on bigrams technique. This algorithm is fast, does not require the removal of affixes nor the use of any morphological rules, capable to find all types of roots, i.e., 3-lateral, 4-lateral, 5-lateral, and 6-lateral roots. There is no difference between strong roots and vocalic roots in our new algorithm. We also established a comparison between our proposed algorithm and two other well-known algorithms in the field, namely: Khodja algorithm, Yousef et al. algorithm. The first one sometimes fails in getting the correct root of the given word and for many words it produced one of two results: (1) not stemmed word (2) a completely new word and sometimes a wrong word that does not exist in Arabic. The same thing can be said for the second one. Although it gives better results than the first, it fails for many words. For the same cases, our new algorithm always gives the correct root, the failure is very limited, and the obtained success ratio of root extraction is very promising. In our future work, we plan to apply our new algorithm on a corpus of Arabic words with big sizes, to improve the obtained success rate, and to apply it in extracting the root of words in other languages such as English and French.

Conclusion

Text categorization task consists in assigning a set of texts to a set of predefined categories (classes/groups/labels). It is a very important task in text mining process used in several applications and domains such as: text indexing, text filtering and routing, hierarchical web page categorization, etc. We distinguish two kinds of text categorization: the first one is the monolingual text categorization whose texts are written in a unique language. The second is the multilingual text categorization whose texts are written in various languages. This last one is justified by: the growing number of documents from collections available on networks and shared across the world, the massive expansion of internet and the increasing availability of textual data on it, the increasing number of internauts whose native languages are different, the new tendency of globalization that dealt with many domains, and the necessity of scientific cooperation and exchange between universities around the world.

There are three main phases in the multilingual text categorization process: language identification, automatic translation and text categorization. The main problematic of our thesis is how to exploit the concepts and algorithms of machine learning in contextual categorization of multilingual texts. We subdivided this problematic into the following sub-issues:

1. Design of a text classifier based on a prediction model that must be the most perfect possible and can support the variability of the language of input texts. To this end, we adopted the three schemes of which are: the trivial scheme, the scheme based on a single language for learning, and the scheme based on the mix of training sets. Our work in this area has allowed us to test and validate the previous three schemes and confirm their reliability.
2. Propose effective solutions to improve the performance of the most critical phases of text categorization including: the language identification phase and the selection of the most relevant terms phase (vocabulary reduction).For this purpose, we have studied the well-known methods of language identification, compared between them, and discussed their advantages and weaknesses. This comprehensive study has guided us to propose a new algorithm which is: simple, fast, and effective inspired of Trenkle and Cavnar method. This algorithm remains among the best known methods in this field of research. On the other hand, we studied some statistical methods that are useful in feature selection such as: the mutual information method MI, the information gain method IG method, and the chi-2 method (χ^2) method, and others, as well as some linguistic methods such as the stemming and lemmatization methods. We focalized on their role in the selection of relevant terms to reduce the size of the vocabulary and increase the efficiency of learning algorithms. Consequently, we noticed the importance of stemming and its positive effect on the quality of the obtained categorization especially for some inflected languages such as Arabic. We proposed an improved algorithm for Arabic stemming based on a full statistical approach based in turn on the extraction of the word root and the technique of n-grams of characters. The advantage of the new

algorithm is that: it is simple, fast, and does notrequireany prior language knowledge. We applied it on Arabic text categorization and information retrieval for the same language. The improvement in terms of reducing the size of term vocabulary and the accuracy of the obtained results for the two areas of research were highly significant.

3. The extension of the Arabic stemming algorithm led to the design and the implementation of a multilingual stemmer as an alternative solution of the classical stemming approach which is the monolingual approach. This one remains very restrictive, and depends on the language for which it has been developed. We tested our multilingual stemmer on three languages, including: Arabic, French, and English. The obtained results were promising. Hence,the new stemmer remains open to support other languages and undergo more improvement.
4. For the categorization itself, we proposed a new panoply of pseudo-distances inspired of the field of language identification. We applied this new panoply of pseudo-distances to categorize texts of Reuters21578 corpus. We have also developed a simple and effective method to categorize collection texts according to their themes.

As perspective for this research project, we propose the following tasks:

1. Use the obtained results in other related areas such as: automatic monolingual and multilingual summarization, the author identification, multi-criteria identification, for instance, author identification and subject categorization at a time.
2. Use multilingual ontologies to guide the categorization process and improve the used prediction model.
3. Extend the obtained solutions so that they effectively treat the case of semi-structured documents.
4. Exploit the opportunities offered by the combination of learning algorithms (cascade merge and parallel merge of classifiers).
5. Perform a data framework that consists of: Arabic corpora, multilingual corpora and use it in future research project.
6. Improve the obtained results in the field of multilingual stemming.

APPENDICES

A.1. English Stopwords

A	Both	Hadn't	I'll	Of	so	through	Which
About	But	Has	I'm	Off	Some	to	While
Above	by	Hasn't	I've	On	such	Too	Who
After	Can't	Have	If	Once	Than	Under	Who's
Again	Cannot	Haven't	In	Only	That	Until	Whom
Against	Could	having	Into	or	That's	Up	Why
All	Couldn't	He	Is	Other	The	Very	Why's
Am	Did	He'd	Isn't	Ought	Their	Was	With
An	Didn't	He'll	It	Our	Theirs	Wasn't	Won't
and	Do	He's	It's	ours	Them	We	Would
Any	Does	Her	Its	Ourselves	Themselves	We'd	Wouldn't
Are	Doesn't	Here	Itself	Out	Then	We'll	Yes
Aren't	doing	Here's	Let's	Over	There	We're	yet
As	Don't	Hers	Me	Own	There's	We've	You
At	Down	Herself	More	Same	These	Were	You'd
Be	during	Him	Most	Shan't	They	Weren't	You'll
Because	Each	Himself	Must	She	They'd	What	You're
Been	Few	His	Mustn't	She'd	They'll	What's	You've
before	For	How	My	She'll	They're	when	Your
Being	From	How's	Myself	She's	They've	When's	Yours
below	Further	I	No	Should	This	Where	Yourself
Between	Had	I'd	Nor	Shouldn't	those	Where's	yourselves

A.2. French stopwords

Alors	dans	Eu	Maintenant	Pour	Sont	Toutes
Au	des	Fait	Mais	Pourquoi	Sous	Trop
Aucun	Du	Faites	Mes	Quand	Soyez	Très
Aussi	Dedans	Fois	Mien	Que	sur	Tu
Autre	Dehors	Haut	mienne	Quell	Ta	Votre
Avant	Depuis	Hors	Moins	Quelle	Tandisque	Vous
Avec	Deux	Ici	Mon	quelles	Tellement	Ça
Avoir	Doit	Il	Même	quels	Tels	être
Bon	Donc	Ils	Ni	Qui	telle	
Car	Droite	Je	Notre	Sa	telles	
Ce	Début	Juste	Nous	Sans	Tes	
Cela	Elle	La	Ou	Ses	Tien	
Celle-ci	Ells	Le	Où	Seulement	Tienne	
Ces	en	Les	Par	Si	Ton	
Ceux	Encore	Leur	Parceque	Sien	Tous	
Chaque	Essai	Leurs	Pas	Sienne	Tout	
Comme	Est	Là	Peu	Son	Toute	
Comment	Et	Ma	Plupart	Que	sur	

A.3. Arabic stopwords

أ	الماضي	بها	شدم	كلم	نفسه	ولم
بـ	المقبل	بين	ششر	كما	نهاية	ومن
إبّان		ثلاثة	ششرة	لا	هذا	وهو
اجل	الوقت	ثم	شلى	لدى	هذه	وهي
احد	ألى	جميع	شليه	لقاء	هناك	يكون
اخرى	اليوم	حاليا	شليها	لكن	هو	يمكن
اذا	اما	حتى	شن	للامم	هي	يوم
اربعة	امام	حوالى	شند	لم	و	
اطار	امس	حول	شندما	لن	واحد	
اكثر	ان	حيث	غدا	له	واضاف	
الا	انه	حين	غير	لها	واضافت	
الاخيرة	انها	خلال	ف	لوكالة	واكد	
الان	او	دون	فان	ما	وان	
الاول	اول	ذلك	فى	مايو	واوضح	
الاولى	اي	سنة	في	مساء	وفي	
التى	ايام	سنوات	فيه	مع	وقال	
الثاني	ايضا	صفر	فيها	مقابل	وقالت	
الثانية	بان	ضد	قبل	مليار	وقد	
الذى	بسبب	ضمن	قد	مليون	وقف	
الذين	بشكل	شام	قوة	من	وكان	
السابق	بعد	شاما	كان	منذ	وكانت	
اللاحق	بعض	شدة	كانت	منها	ولا	
الف	به	شدد	كل	نحو		

A.4. German Stopwords

aber	deine	einer	ja	musst	sonst	werden
als	dem	eines	jede	müssen	soweit	werdet
am	den	er	jedem	müßt	sowie	weshalb
an	der	es	jeden	nach	und	wie
auch	des	euer	jeder	nachdem	unser	wieder
auf	dessen	eure	jedes	nein	sich	wieso
aus	deshalb	für	jener	nicht	unsere	wir
bei	die	hatte	jenes	nun	unter	wird
bin	dies	hatten	jetzt	oder	vom	wirst
bis	dieser	hattest	kann	seid	von	wo
bist	dieses	hattet	kannst	sein	vor	woher
da	doch	hier	können	seine	wann	wohin
dadurch	dort	hinter	könnt	sich	warum	zu
daher	du	ich	machen	sie	was	zum
darum	durch	ihr	mein	sind	weiter	zur
das	ein	ihre	meine	soll	weitere	über
daß	eine	im	mit	sollen	wenn	
dass	einem	in	muß	sollst	wer	
dein	einen	ist	mußt	sollt	werde	

A.5. Spanish Stopwords

algún	como	encima	hace	modo	quien	tengo	usais
alguna	con	entonces	haceis	muchos	sabe	tiempo	usamos
algunas	conseguimos	entre	hacemos	muy	sabeis	tiene	usan
alguno	conseguir	era	hacen	nos	sabemos	tienen	usar
algunos	consigo	eramos	hacer	nosotros	saben	todo	usas
ambos	consigue	eran	haces	otro	saber	trabaja	uso
ampleamos	consiguen	eras	hago	para	sabes	trabajais	va
ante	consigues	eres	incluso	pero	ser	trabajamos	vais
antes	cual	es	intenta	podeis	si	trabajan	valor
aquel	cuando	esta	intentais	podemos	siendo	trabajar	vamos
aquellas	dentro	estaba	intentamos	poder	sin	trabajas	van
aquellos	desde	estado	intentan	podria	sobre	trabajo	vaya
aqui	donde	estais	intentar	podriais	sois	tras	verdad
arriba	dos	estamos	intentas	podriamos	solamente	tuyo	verdadera
atras	el	estan	intento	podrian	solo	su	verdadero
bajo	ellas	estoy	ir	podrias	somos	sus	vosotras
bastante	ellos	fin	la	por	soy	ultimo	vosotros
bien	empleais	fue	largo	por qué	su	un	voy
cada	emplean	fueron	las	porque	sus	una	yo
cierta	emplear	fui	lo	primer	también	unas	
ciertas	empleas	fuimos	los	puede	teneis	uno	
cierto	empleo	gueno	mientras	pueden	tenemos	unos	
ciertos	en	ha	mio	puedo	tener	usa	

A.6. Italian Stopwords

a	che	doppio	io	no	promesso	sette	te
adesso	chi	due	la	noi	qua	sia	tempo
ai	cinque	e	lavoro	nome	quarto	siamo	terzo
al	comprare	ecco	le	nostro	quasi	siete	tra
alla	con	fare	lei	nove	quattro	solo	tre
allo	consecutivi	fine	lo	nuovi	quello	sono	triplo
allora	consecutivo	fino	loro	nuovo	questo	sopra	ultimo
altre	cosa	fra	lui	o	qui	soprattutto	un
altri	cui	gente	lungo	oltre	quindi	sotto	una
altro	da	giu	ma	ora	quinto	stati	uno
anche	del	ha	me	otto	rispetto	stato	va
ancora	della	hai	meglio	peggio	sara	stesso	vai
avere	dello	hanno	molta	pero	secondo	su	voi
aveva	dentro	ho	molti	persone	sei	subito	volte
avevano	deve	il	molto	piu	sembra	sul	vostro
ben	devo	indietro	nei	poco	sembrava	sulla	
buono	di	invece	nella	primo	senza	tanto	

A.7. Some categories in the Penn tree bank POS set.

N°	Tag	Description	N°	Tag	Description
1	CC	Coordinating conjunction	19	PRP$	Possessive pronoun
2	CD	Cardinal number	20	RB	Adverb
3	DT	Determiner	21	RBR	Adverb, comparative
4	EX	Existential there	22	RBS	Adverb, superlative
5	FW	Foreign word	23	RP	Particle
6	IN	Preposition/subordinating conjunction	24	Sym	Symbol
7	JJ	Adjective	25	TO	to
8	JJR	Adjective, comparative	26	UH	Interjection
9	JJS	Adjective, superlative	27	VB	Verb, base form
10	LS	List item maker	28	VBD	Verb, past tense
11	MD	Modal	29	VBG	Verb, gerund or present participle
12	NN	Noun, singular or mass			Verb, past participle
13	NNS	Noun, plural	30	VBN	Verb, non-3rd person singular present
14	NNP	Proper noun, singular	31	VBP	Verb, 3rd pers singular present
15	NNPS	Proper noun, plural			Wh-determiner
16	PDT	Predeterminer	32	VBZ	Wh-Pronoun
17	POS	Possessive ending	33	WDT	Possessive wh-pronoun
18	PRP	Personal pronoun	34	WP	Wh-adverb
			35	WPS	
			36	WRB	

A.8. An example of trilateral, quadrilateral, quinquelateral, hexalateral roots

Trilateral roots	Quadrilateral roots	Quinquelateral roots	Hexalateral roots
زرع	أكرم	انطلق	استعمل
صنع	أشان	انكسر	استحسن
تجر	أشطى	احتوى	استعان
جمع	حطّم	اقتصد	اخشوشن
أبى	شلّم	اجتمع	اشوشب
نفر	ربّى	اخضرّ	ادهامّ
شلى	برّأ	تقدّم	اخضارّ
دار	قاتل	تحطّم	اجلوذّ
طار	حاسب	تحدّى	احرنجم
شطس	داين	تعاطى	افرنقع
سعل	طمأن	تنازل	اقشعرّ
صدع	زلزل	تدحرج	اطمأنّ

A.9. Extraction of some words roots using the three algorithms

Word	Extracted Root			Correct root
	Kodja stemmer	Yousef et al stemmer	Our proposed stemmer	
يتعلّمون	**علم**	**علم**	**علم**	**علم**
كاتب	**كتب**	**كتب**	**كتب**	**كتب**
كتاتيب	**Not stemmed**	**كتب**	**كتب**	**كتب**
اقتصاد	**قصد**	**اقتصد**	**اقتصد**	**اقتصد**
معالجة	شلج	شالج	شالج	شالج
سنستدرجهم	**Not stemmed**	**درج**	**درج**	**درج**
متلألئ	**Not stemmed**	**لألأ**	**لألأ**	**لألأ**
يهزمونهم	Not stemmed	هزم	هزم	هزم
المربّي	**ربأ**	**ربّى**	**ربّى**	**ربّى**
المربّون	**ربن**	**ربّى**	**ربّى**	**ربّى**
طائر	طور	طار	طار	طار
طائرات	**طور**	**طار**	**طار**	**طار**
طيور	**Not stemmed**	**طار**	**طار**	**طار**
طيران	**أعان**	**طار**	**طار**	**طار**
كهرباء	Not stemmed	كهرب	كهرب	كهرب
واقعة	قعي	وقع	وقع	وقع
ولولة	**ليل**	**ولول**	**ولول**	**ولول**
وقيعة	**قوع**	**وقع**	**وقع**	**وقع**
ميزان	ميّز	أشان	وزن	وزن
يزنونهم	**زنن**	**نهب**	**وزن**	**وزن**
مهيمنون	هوم	هيمن	هيمن	هيمن
منقولة	قول	كمل	نقل	نقل
ينفعونهم	فعي	نفع	نفع	نفع
منظمات	ظمأ	نظم	نظم	نظم
الطمأنينة	Not stemmed	طمأن	طمأن	طمأن
مطمئنّون	طمي	اطمأن	اطمأن	اطمأن
زلازل	**Not stemmed**	**تنازل**	**زلزل**	**زلزل**
حواسيب	**Not stemmed**	**نسي**	**حسب**	**حسب**
تجارة	جور	تجر	تجر	تجر
بكاء	بكأ	بكى	بكى	بكى
اتصالات	صلّى	اتصل	اتصل	اتصل
الإثم	stopword	أثم	أثم	أثم
الأحاسيس	حوس	احسّ	احسّ	احسّ
متعجّل	شجل	تعب	شجل	شجل
شطاس	شطس	طار	شطس	شطس
ناجحات	نجح	نام	نجح	نجح
نجاح	نجح	احرنجم	نجح	نجح
ناسج	**نسج**	**سجد**	**نسج**	**نسج**
طامعون	طمع	جمع	طمع	طمع
مهدّم	هدم	تقدّم	هدم	هدم
متوهّج	وهج	وهب	وهج	وهج
منابت	نبت	ابتهل	نبت	نبت
نبوغ	**نبغ**	**نبت**	**نبغ**	**نبغ**
نابغة	نبغ	نام	نبغ	نبغ
منابع	نبع	كمن	نبع	نبع
نوازل	**نزل**	**تنازل**	**نزل**	**نزل**
منتشر	نشر	كمن	نشر	نشر

Bibliography

[Aas and Eikvil , 1999] Aas, K. and Eikvil, L. (1999).*Text categorization: a survey*. Technical report.Norwegian Computing center.

[Ababneh et al., 2012] Ababneh, M., Al-Shalabi, R., Kanaan, G., and Al-Nobani A., "Building an effective Rule-Based Light Stemmer for Arabic Language to Improve Search Effectiveness," the *International Arab Journal of Information Technology*, 9(4), July 2012.

[Abbott, 2013] Abbott, A. (2013). Introduction to text mining: Virtual data intensive summer school (Pdf slides), Abbott analytics Inc, July, 10, 2013.

[Abu Hawas and Keith, 2014] Abu Hawas F., Keith E., (2014). Rule-based Approach for Arabic Root Extraction: New Rules to Directly Extract Roots of Arabic Words. Journal of computing and Information Technology – CIT, Zaghreb, pp.57–68.

[Abu Hawas, 2015] Abu Hawas, F. (2013). Exploit relations between the word letters and their placement in the word for Arabic root extraction. *Computer Science Journal*, Vol.14, pp.27-431, 2013.

[Agrawal and Srikant, 2000] Agrawal, R. and Srikant, R. (2000).Privacy-preserving data mining.In *Proc. of 2000 A CMSIGMOD Inti. Conf. on Management of Dat*a, pages 439- 450, Dallas, Texas, 2000. ACM Press.

[Aizerman et al., 1964] Aizerman, M., Braverman, E.M., and Rozonoer., L.I. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control* 25:821–837.

[Al-Kamar, 2006] Al-Kamar R. (2006). *Computer and Arabic language computerizing*.Dar Al Kotob Al-Ilmiya, Cairo, Egypt.

[Al-Nashashibi, 2010a] Al-Nashashibi, M.Y., Neagu, D., Yaghi, A.A. (2010a). An improved root extraction technique for Arabic words," *Proc.2nd International Conference on Computer Technology and Development(ICCTD)*.

[Al-Nashashibi, 2010b] Al-Nashashibi, M. Y., Neagu, D., Ali, A. Y. (2010). Stemming Techniques for Arabic Words: A Comparative Study,"*Proc.2nd International Conference on.Computer Technology and Development (lCCTD 2010)*, pp.270–276.

[Al-omari et al., 2013] Al-omari, A., Abuata, B., Al-kabi, M. (2013). Building and Benchmarking New Heavy/Light Arabic Stemmer.In *Proceedings of 4th International Conference on Information and Communication Systems (ICICS'13)*.

[Al-shalabi et al., 2003] Al-shalabi, R., Kanaan, G., Al-Serhan, H. (2003). New Approach for Extracting Arabic Roots. In *Proceedings of the International Arab Conference on Information Technology (ACIT'2003)*, Alexandria, Egypt, pp. pp.42–59.

[Al-shalabi et al., 2008] Al-Shalabi, R., Kanaan, G., Ghwanmeh, S. (2008). Stemmer Algorithm for Arabic Words Based on Excessive Letter Locations, *IEEE Conf*, 2008.

[Al-shalabi, 2005] Al-shalabi, R. (2005). Pattern-based stemmer for finding Arabic roots," Information Technology Journal, 4(1): 38–43.

[Andoni et al., 2006] Andoni, A., Mayur, D., Nicole, I., Piotr, I., and Vahab, M. (2006). Locality-sensitive hashing using stable distributions. In *Nearest Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press.

[Androutsopoulos and al., 2000] Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., and Spyropoulos, C.D. (2000).An Experimental Comparison of Naive Bayesian and Keyword-Based Anti-Spam Filtering with Personal E-mail Messages. In *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*. Athens, ACM Press, New York: 160–167.

[Apté *et al.,* 1994] Apté, C., Damerau, F. and Weiss, S. (1994). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, 1994. [Forman, 2003] Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification.*Journal of Machine Learning Research*, 3:1289–1305, 2003.

[Baker and McCallum, 1998] Baker, L. D., and McCallum, A. K. (1998).Distributional Clustering of Words for Text Classification. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*. Melbourne, Australia, ACMPress, New York: 96–103.

[Beesly, 1988] Beesley, K. (1988). Language Identifier: A Computer Program for Automatic Natural Language Identification on On-Line Text, In *Proceedings of the 29th Annual Conference of the American Translators Association*, pages: 47–54.

[Berry and Kogan, 2010] Berry, M.W. and Kogan, J. (2010).*Text Mining Applications and Theory*, ISBN: 978-0-470-74982-1 , 2010, Copyright John Wiley & Sons, Ltd.

[Berry and Linoff, 2004] Berry, M.J.A. and Linoff, G. (2004).*Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management.* Wiley Computer Publishing, 2nd edition, 2004.

[Bhandari et al., 1997] Bhandari, I., Colet, E., Parker, J., Pines Z., Pratap R., Ramanujam K. (1997). Advanced Scout: data mining and knowledge discovery in NBA data.*Data Mining and Knowledge Discovery*, 1 (1): 121-125.

[Bikel *et al.,* 1997] Bikel, D., Miller, S., Schwartz, R., and Weischedel, R. (1997). Nymble: A high-performance learning name finder. In *The Fifth Conference on Applied Natural Language Processing*, pages 194–201. ACM, New York, 1997.

[Bikel *et al.,* 1997] Bikel, D., Schwartz, R., and Weischedel, R. (1997).An algorithm that learns what's in a name.*Machine Learning*, 34(1–3):211–231, 1999.

[Biskri and Delisle, 2001] Biskri, I., Delisle, S., (2001). Les n-grammes de caractères pour l'aide à l'extraction de connaissance dans des bases de données textuelles multilingues, *TALN 2001*, Tours, 2-5 juillet 2001

[Boudlal et al., 2011] Boudlal, A., Belahbib, R., Belahbib, A., Mazroui, A. (2011). A markovian approach for Arabic root extraction.the*International Arab Journal of Information Technolog IAJIT*, Vol. 8, pp.91-98, 2011.

[Brachman and Anand, 1996] Brachman, R., and Anand, T. (1996). In "The Process of Knowledge Discovery in Databases: A Human Centered Approach." *Advances in Knowledge Discovery and Data Mining*.U.M.

[Brill, 1995] Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995. http://www.cis.upenn.edu/~adwait/penntools.html.

[Brown, 2012] Brown, R.D. (2012). Finding and identifying text in plus de 900 languages, Published by Elsevier Ltd. All rights reserved.

[Buckley et al., 1994] Buckley, C., Salton, G., Allan, J. and Singhal, A. (1994). Automatic query expansion using SMART: Trec 3. In *Information Retrieval Conference*.

[Buckley et al., 1994a] Buckley, C., Allan, J., and Salton, G. (1994a). Automatic routing and ad-hoc retrieval using SMART: TREC 2. In *Proc. TREC*,

[Buckley et al., 1994b] Buckley, C., Salton, G., and Allan, J. (1994b). The effect of adding relevance information in a relevance feedback environment.In *Proc. SIGIR*, pp. 292–300. ACM Press.

[Burges, 1998] Burges, C.J. C. 1998. A tutorial on support vector machines for pattern Recognition. *Data Mining and Knowledge Discovery* 2(2):121–167.

[Caropreso et al., 2000] Caropreso, M., Fernanda, M., Matwin, S., Sebastiani, F., (2000). *Statistical Phrases in Automated Text Categorization*. Department of Computer Science of the University of URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.24. 5902

[Caropreso et al., 2001] Caropreso, M. F., Matwin, S., and Sebastiani, F. (2001). A Learner-Independent Evaluation of the Usefulness of Statistical Phrases for Automated Text Categorization." In *Text Databases and Document Management*: *Theory and Practice*. A.G. Chin, ed. Hershey, PA, Idea Group Publishing: 78–102.

[Cavnar and Trenkle, 1994] Cavnar, W.B. and Trenkle, J. M. (1994).N-gram-based text categorization. In proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, pages 161-175, Las Vegas, USA. Available on: http://www.nonlineardynamics.com/trenkle/papers/sdair-94-bc.ps.gz.

[Chai and al, 2002] Chai, K. M., Ng, H. T., and Chieu, H. L. (2002).Bayesian Online Classifiers for Text Classification and Filtering. In *Proceedings of SIGIR-02, 25th ACM International Conference on Research and Development in Information Retrieval*. Tampere, FI,ACMPress, New York: 97–104.

[Chakrabarti et al., 2003]. Chakrabarti, S. (2003). *Mining the web: Discovery Knowledge from Hypertext Data.*Morgan Kaufmann Publishers.

[Chang and Lin, 2011] Chang, C.-C.and Lin, C.-J. (2011). LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1-27:27, 2011.

[Charniak, 1997] Charniak, E. (1997). Statistical techniques for natural language parsing.*AI Magazine*, 18(4):33–43, 1997.

[Chen et al., 1996] Chen, M.S., Han, J. and Yu, P. S. (1996).Data. Mining: An Overview from a Database Perspective. *IEEE Transactions on /(nowledge abd Data Engineering,* 8(6):865-883.

[Chen et al., 2005] Chen, P-H, Lin, C-J.and Sch¨olkopf, B. (2005). A tutorial on *v*-support vector machines.*Applied Stochastic Models in Business and Industry* 21:111–136.

[Cherkassky and Mulier, 1998]Cherkassky, V. and Mulier, F. (1998). *Learning from Data: Concepts, Theory, and Methods.*Wiley interscience, 1998.

[Chiang, 2000] Chiang, D. (2000). Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of the ACL 2000*, pages 456–463. ACL, East Stroudsburg, 2000.

[Cohen , 1995a] Cohen,W.W. (1995a). Learning to Classify English Text with ILP Methods."In *proceedings of Advances in Inductive Logic Programming*. L. D. Raedt, ed. Amsterdam, IOS Press: 124–143.

[Cohen , 1995b] Cohen,W.W. (1995b). Text Categorization and Relational Learning. In *Proceedings of ICML-95, 12th International Conference on Machine Learning*. Lake Tahoe, NV, Morgan Kaufmann Publishers, San Francisco: 124–132.

[Cohen and Hirsh, 1998] Cohen, W. W., and Hirsh, H. (1998). Joins that Generalize: Text Classification Using Whirl. In *Proceedings of KDD-98, 4th International Conference on Knowledge Discovery and Data Mining*. New York, AAAI Press, Menlo Park, CA: 169–173.

[Cohen and Singer, 1996] Cohen, W. W., and Singer, Y. (1996).Context-Sensitive Learning Methods for Text Categorization. In *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*. Zurich, ACM Press, New York: 307–315.

[Cohen and Singer, 1999] Cohen, W. W., and Singer, Y. (1999).“Context-Sensitive Learning Methods for Text Categorization.*ACM Transactions on Information Systems* 17(2): 141–173.

[Cohen, 1995a] Cohen,W.W. (1995a). Learning to Classify English Text with ILP Methods.”In *Advances in Inductive Logic Programming*. L. D. Raedt, ed. Amsterdam, IOS Press: 124–143.

[Cohen, 1995b] Cohen,W.W. (1995b). Text Categorization and Relational Learning. In Proceedings of ICML-95, 12th International Conference on Machine Learning. Lake Tahoe, NV, Morgan Kaufmann Publishers, San Francisco: 124–132.

[Cohen, 1996] Cohen,W., and Singer, Y. (1996). Context-Sensitive Learning Methods for Text Categorization.In *Proceedings of SIGIR-96, 19thACM.International Conference on Research and Development in Information Retrieval*. H.-P Frei, D. Harman, P. Schauble and R.Wilkinson, eds. Zurick, Switzerland, ACM Press, New York, 307–315.

[Cohen, 1998] Cohen, W. W. (1998). Web-based Information System that Reasons with Structured Collections of Text.*In Autonomous Agent’98* 1998

[Cohen, 1999] Cohen, W. W. and Fan, W. (1999). Learning Page-Independent Heuristics for Extracting Data from Web Pages. In *WWW8 Conference Refereed Papers.*

[Cowie and Lehnert, 1996] Cowie, J., and Lehnert,W. (1996). “Information Extraction.”*Communications of the Association of Computing Machinery* **39**(1): 80–91.

[Crammer and Singer, 2001] Crammer, K., and Singer, Y. (2001).On the algorithmic implementation of multiclass kernel-based machines.*JMLR* 2:265–292.

[Creecy et al., 1992] Creecy, R.H., Masand, B.M., Stephen J. Smith, and David L. Waltz. 1992. Trading MIPS and memory for knowledge engineering. *CACM* 35(8):48–64. doi: http://doi.acm.org/10.1145/135226.135228.

[Cristianini and Shawe-Taylor, 2000] Cristianini, N., and Shawe-Taylor, J. (2000).*Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge.

[Cutler, 1997] Cutlern, Y. S. M. and Leng, W. (1997).Using the Structure of HTML Documents to Improve Retrieval.*In USENIX Symposium on Internet Technologies and Systems*.

[Dawson, 1974] Dawson, J. (1974). Suffix removal and word conflation. *Bul.of.Assoc.for Liter&Ling.Comp*. 2, 3, 33–47 (1974)

[de Loupy, 2001] de Loupi, C., (2001). L’apport de connaissances linguistique en recherche documentaire. In *TALN 01*.

[Deerwester, 1990] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990).“Indexing by Latent Semantic Analysis.*Journal of the American Society of Information Science* , 41(6): 391–407.

[Denoyer, 2004] Denoyer, L. (2004). *Apprentissage et inference statistique dans les bases de documents structurés : Application aux corpus de documents textuels*. PhD thesis, Université de Paris 6, france

[Dimitrios et al., 2013] Dimitrios, A., Pritsos, C., Stamatatos, E. (2013). Open-Set classification for Automated Genre Identification ,*ECIR 2013*, LNCS 7814, pp. 207–217, 2013. Springer-Verlag, Berlin Heidelberg, 2013

[Domingos and Hulten, 2000] Domingos, P. and Hulten, G. (2000). Mining high-speed data streams. In *Proc. of the 6th Jntl.Conf.on Knowledge Discovery and Data Mining,* pages 71-80, Boston, Massachusetts, 2000. ACM Press.

[Dumais *et al.*, 1998] Dumais, S. T., Platt, J., Heckerman, D., and Sahami, M. (1998).Inductive Learning Algorithms and Representations forText Categorization.In *Proceedings of 7th International Conference on Information and Knowledge Management*. Bethesda, MD, ACM Press, New York: 148–155.

[Dumais, 1991] Dumais, S. (1991). Improving the retrieval of information from external sources.*Behavior research methods, instruments&computers*, 23(2): 229-236

[Dunham, 2002] Dunham, M.H. (2002). *Data Mining: Introductory and Advanced Topics.* Prentice Hall.

[Dunning, 1994] Dunning, T. (1994).*Statistical Identification of Languages*. Technical Report MCCS 94-273, Computing Research Laboratory URL : http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.48.

[Duwairi, 2007] Duwairi, R. (2007). Arabic Text Categorization.*The International Arab Journal of Information Technolog*, vol. 4, No. 2, 125–131, 2007.

[Emmanuel, 2007] Emmanuel, E., and et Chappelier, J-C. (2007). *Inclusion de sens dans la représentation de documents textuels: état de l’art,* URL : http://infoscience.epfl.ch/record/115458/files/inclusion_de_sens.pdf

[Eyheramendy and al., 2003] Eyheramendy, S., David, L., and David, M. (2003). On the Naive Bayes model for text categorization.In *Proc. InternationalWorkshop on Artificial Intelligence and Statistics*.Society for Artificial Intelligence and Statistics.

[Fayyad et al, 1996a] Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P. and Uthurusamy, R. (1996) *Advances in Knowledge Discovery and Data Mining*.AAAI/MIT Press, 1996.

[Fayyad et al., 1996b] Fayyad, U.M., Piatetsky-Shapiro, G. and Smyth, P. (1996). From Data Mining to Knowledge Discovery: An Overview. In *Advances in Knowledge Discovery and Data Mining,* pages 1- 34.*AAAI* Press.

[Feldbaum, 1998] Feldbaum, C. (1998). *Wordnet: An Electronic Lexical Database*. MIT Press, Cambridge, 1998.

[Feldman and Hirsh, 1996a] Feldman, R., and Hirsh, H. (1996a). "Exploiting Background Information in Knowledge Discovery from Text."*Journal of Intelligent Information Systems* **9**(1): 83–97.

[Feldman and Hirsh, 1996b] Feldman, R., and Hirsh, H. (1996b). *Mining Associations in Text in the Presence of Background Knowledge*.In Proceedings of the 2nd International Conference on Knowledge Discovery from Databases. Portland, OR, AAAI Press, Menlo Park, CA: 343–346.

[Feldman and Sanger, 2007] Feldman, R. and Sanger, J. (2007).*The text mining handbook: Advanced Approaches in Analyzing Unstructured Data*, Cambridge University press, ISBN-13 978-0-521-83657-9 (hardback), © Ronen Feldman and James Sanger 2007.

[Feldman et al., 1998] Feldman, R., Fresko, M., Hirsh, H., Aumann, Y., Liphstat, O., Schler, Y., and Rajman, M. (1998).Knowledge Management: A Text Mining Approach. In *Proceedings of the 2nd International Conference on Practical Aspects of Knowledge Management (PAKM98)*. Basel, Switzerland.

[Frakes, 1992] Frakes, W.B. (1992). *Stemming Algorithms*, Information Retrieval: Data Structures and Algorithms, Frakes, W.B. and R. Baeza-Yates (Eds.), Prentice-Hall India, 1992.

[Freund and Shapire, 1997] Freund, Y. and Schapire, R.E. (1997).A decision-theoretic generalization of on-line learning and an application to boosting.*Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[Freund and Shapire, 1999] Freund, Y. and Schapire, R.E. (1999).A short introduction to boosting.*Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, 1999.

[Friedman, 1979]. Friedman, J. H. (1979). Data Mining and Statistics: What's the Connection?.www-stat.stanford.ed u/ ~j hf/ftp/dm-stat. ps.

[Fuhr and Buckely, 1991] Fuhr, N., Buckely, C. (1991). A probabilistic learning approach for document indexing.In *ACM Transactions on Information Systems*. Volume 9, pages 223 - 248

[Fuhr and Großjohann, 2001] Fuhr, N. and Großjohann, K. (2001).Xirql : a query language for information retrieval in XML documents. In Kraft, D. H., Croft, W. B., Harper, D. J. et Zobel, J., éditeurs : In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 172–180. ACM Press.

[Furnkranz et al., 1998] Furnkranz, J., Mitchell, T., Rilo, E. (1998).*A Case Study in Using Linguistic Phrases for Text Categorization on the WWW*. School of Computer Science Carnegie Mellon University URL : www.cs.utah.edu/~riloff/pdfs/final-webslog-paper.pdf

[Gaafar et al., 2007] Gaafar, J., Gaafar, M. (2007).*Arabic Verbs and Essentials of Grammar 2E (Verbs and Essentials of Grammar Series)*, 2nd ED, McGraw-Hill, Inc, pp. 160.

[Gadri and Moussaoui, 2014b] Gadri, S., Moussaoui, A., Linda, B-F. (2014b). Language Identification: Proposition of a New Optimized Variant for the Method of Cavenar and Trenkle, 1st *International Conference on Artificial Intelligence and Information Technology ICAIIT'14*, University Kasdi Merbah, Ouargla, Algeria, 10-12 Mars 2014.

[Gadri and Moussaoui, 2014c]Gadri, S., Moussaoui, A., Linda B-F. (2014c). Language Identification: A New Fast Algorithm to Identify the Language of a Text in a Multilingual Corpus, *The 4th International Conference on Multimedia Computing and Systems ICMCS'14*, IEEE Conference, University of Marrakesh, Marrakesh, Morocco, 14-16 Avril 2014.

[Gadri and Moussaoui, 2014d]Gadri, S., Moussaoui, A. (2014d).Contextual Categorization of Documents Using a New Panoply of Similarity Metrics, *International Conference on Advanced Technology & Sciences (ICAT'14)*, 12-15 August, 2014, Antalya, Turkey,

[Gadri and Moussaoui, 2015a] Gadri, S., Moussaoui, A., (2015a). Arabic Texts Categorization: Features Selection Based on the Extraction of Words' Roots, *5th IFIP International Conference on Computer Science and its Applications (CIIA'2015)*, 20, 21 May 2015, Springer Conference, TaharMoulay University, Saida - Algeria.

[Gadri and Moussaoui, 2015b] Gadri, S., Moussaoui, A. (2015b). Arabic Text Categorization: An Improved Algorithm Based on N-grams technique to Extract Arabic Words Roots. *International Arab Journal of Information Technology IAJIT*, accepted for publication, 15(2), March 2018.

[Gadri and Moussaoui, 2015c] Gadri, S., Moussaoui, A., (2015c). Information Retrieval: A New Multilingual Stemmer Based on a Statistical Approach, *3rd International Conference on control, engineering & information Technology (CEIT2015)*, 25-27 may, IEEE Conference, Tlemcen, Algeria.

[Gadri and Moussaoui, 2015d] Gadri, S., Moussaoui, A. (2015d). Multilingual information retrieval: increasing the effectiveness of search by stemming, *19th International Conference on Circuits, Systems, Communications and Computers (CSCC 2015)*, July 16-20, 2015, Zakynthos Island, Greece. Conference Link: http://2015.cscc.co/

[Gadri and Moussaoui, 2015e] Gadri, S., Moussaoui, A. (2015e). Multilingual Text Categorization: Increasing the Quality of Categorization by a Statistical Stemming Approach, *International Conference on intelligent Information Processing Security and Advanced Communication (IPAC 2015)*, ACM Conference, November 23-25, 2015 Batna, Algeria.

[Geiger et al., 2012] Geiger, W.M. ., Rauch, J., Hornik, K. (2012). *Text categorization in R: A Reduced N-grams Approach*, Springer-Verlag, Berlin Heidelberg.

[Ghazzawi, 1992] Ghazzawi S. (1992). *The Arabic Language in the Class Room*, 2nd ED, Georgetown University, Washington DC, 1992

[Ghwanmeh et al., 2005] Ghwanmeh, S., Al-Shalabi, R., Kanaan, G., Khanfar K. (2005).An Algorithm for extracting the Root of Arabic Words,"*Proc.5th International Business Information Management Conference (IBIMA)*, Cairo, Egypt.

[Ghwanmeh et al., 2009] Ghwanmeh, S., Kanaan, G., Al-Shalabi, R., Rabab'ah, S. (2009).Enhanced algorithm for extracting the root of Arabic words. In *Proceedings of the 6th international Conference on Computer Graphics, Imaging Visualization*, Aug. 11-14, EEE Xp Press, Tianjin, Chain, pp: 388-391.

[Giguet, 1998] Giguet, E. (1998). *Méthode pour l'analyse automatique de structures formelles sur documents multilingues* , PhD thesis, Université de Caen, France, 1998.

[Giorgetti and Sebastiani, 2003a] Giorgetti, D., and Sebastiani, F. (2003a). "Automating Survey Coding by Multiclass Text Categorization Techniques."*Journal of the American Society for Information Science and Technology* **54**(12): 1269–1277.

[Giorgetti and Sebastiani, 2003b] Giorgetti, D., and Sebastiani, F. (2003b). Multiclass Text Categorization for Automated Survey Coding. In *Proceedings of SAC-03, 18th ACM Symposium on Applied Computing.* Melbourne, Australia, ACM Press, New York: 798–802.

[Glymour et al., 1997] Glymour, C., Madigan, D., Pregibon, D. and Smyth, P. (1997).Statistical Themes and Lessons for Data Mining.*Data Mining and KDD Journal,*1(1):11-28.

[Grefenstette, 1995] Grefenstette, G. (1995). Comparing Two Language Identification Schemes. In *Proceedings of the 3rd International Conference on the Statistical Analysis of Textual Data*, Rome, Italy, 1995.

[Grishman, 1997] Grishman, R. (1997). "Information Extraction: Techniques and Challenges." In Materials of Information Extraction International Summar School – SCIE '97.Springer, Berlin: 10–27.

[Grivel, 2007] Grivel, L. (2007). Introduction au text mining pour la veille et l'intelligence économique. Séminaire « veille et text mining », Rabat, 24-25 mai 2007, Maroc

[Grust, 2002] Grust, T. (2002).Accelerating XPath Location Steps. In *Proceedings of the 21st International ACM SIGMOD Conference on Management of Data*, pages 109-120.

[Hajjar et al., 2010] Hajjar, A.E.S.A., Hajjar, M., Zreik, K. (2010). A system for evaluation of Arabic root extraction methods," In *Proc.5th International Conference on Internet and Web Applications and Services (ICIW)*, May 9-15, IEEE Xplore Press, Barcelona, pp.506-512, 2010.

[Han and Kamber, 2006] Han, J. and Kamber, M. (2009).*Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, Elsevier, 2nd edition, USA

[Han and Karypis] Han, E-H., and Karypis, G. (2000). Centroid-based document classification: Analysis and experimental results. In *PKDD*, pp. 424–431.

[Han et al., 2002] Han, J., Altman, R.B., Kumar, V., Mannila, H. and Pregibon, B. (2002). Emerging scientific applications in data mining.*Communications of the ACM,* 45(8):5-58..

[Hand et al., 2001] Hand, D.J., Mannila, H. and Smyth, P. *Principles of Data Mining.*MIT Press, 2001.

[Hastie et al., 2001] Hastie, T., Tibshirani, R. and Friedman, J.H. (2001).*The Elements of Statistical Learning: Data Mining, Inference, Prediction.* Springer, New York, 2001.

[Hawking and Smeaton, 1997] Hawking, D. and Smeaton, A. (1997).eds. Toronto, ACM Press, New York: 190–197.

[Hayes and Weinstein, 1990] Hayes, P. J., and Weinstein, S. P. (1990). Construe/Tis: A System for Content-Based Indexing of a Database of News Stories. In *Proceedings of IAAI-90, 2nd Conference on Innovative Applications of Artificial Intelligence*. Boston, AAAI Press, Menlo Park, CA: 49–66.

[Hayes et al., 1988] Hayes, P. J., Knecht, L. E., and Cellio, M. J. (1988).A News Story Categorization System. In *Proceedings of ANLP-88, 2nd Conference on Applied Natural Language Processing*. Austin, JX, Association for Computational Linguistics, Morristown, NJ: 9–17.

[Hayes et al., 1990] Hayes, P. J., Andersen, P. M., Nirenburg, I. B., and Schmandt, L. M. (1990). Tcs: A Shell for Content-Based Text Categorization. In *Proceedings of CAIA-90, 6th IEEE Conference on Artificial Intelligence Applications*. Santa Barbara, CA, IEEE Computer Society Press, Los Alamitos, CA: 320–326.

[Hayes, 1992] Hayes, P. (1992). "Intelligent High-Volume Processing Using Shallow, Domain-Specific Techniques." In *proceedings of Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*. P. S. Jacobs, ed. Hillsdale, NJ, Lawrence Earlbaum: 227–242.

[He et al., 2000] He, J., Tan, A-H., and Tan, C-L.(2000). A comparative study on chineese text categorization methods.In PRICAI Workshop on text and web mining. Pages 24-35

[Hmeidi et al., 2010] Hmeidi, I.I., Al-Shalabi, R., Al-Taani, A.T., Najadat, H., and Al-Hazaimeh, S.A. (2010). A novel approach to the extraction of roots from Arabic words using bigrams. *Journal of American Society for Information Science and Technology*., Vol. 61, pp.583-591.

[Hofmann and Klinkenberg, 2013] Hofmann, M., Klinkenberg, R. (2013).*RapidMiner: Data Mining Use Cases and Business Analytics Applications*. Chapman & Hall, USA.

[Hsu, 1998] Hsu, C. N. (1998). Initial Results on Wrapping Semistructured Web Pages.In *proceedings of AAAI-98 Workshop on AI and Information Integration.*

[Huang, 2000] Huang, C and Darwiche, A. (2000). Inference in Belief Networks: a Procedural Guide. *In International Journal of Approximate Reasoning*, Vol. 11, pages 1-158.

[Hull, 1996] Hull, D. (1996). "Stemming Algorithms – A Case Study for Detailed Evaluation."*Journal of the American Society for Information Science* **47**(1): 70–84.

[Ide and Véronis, 1998] Ide, N., and Véronis, J. (1998). Word sense disambiguation: The state of the art. *Computational Linguistics*, 24(1):1–40.

[Indurkhya and Damerau, 2010]Indurkhya N, Damerau, F. (2010).*Handbook of Natural Language Processing, Second Edition*.CRC Press/Taylor and Francis, Boca Raton/London, 2010.

[Indyk, 2004] Indyk, Piotr. 2004. *Nearest neighbors in high-dimensional spaces*. In J. E. Goodman and J. O'Rourke (eds.), Handbook of Discrete and Computational Geometry, 2nd edition. pp. 877–892. Chapman and Hall/CRC Press.

[Iwayama and Tokunaga, 1995] Iwayama, M., Tokunaga, T. (1995). Cluster-based text categorization: A comparison of category search strategies", FOX E. A., INGWERSEN P., FIDEL R., Eds., *Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development In Information Retrieval*, Seattle, 1995, NewYork, p. 273–281.

[Iyer et al., 2000] Iyer, R. D., Lewis, D. D., Schapire, R. E., Singer, Y., and Singhal, A. (2000). Boosting for Document Routing. In *Proceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management*. A. Agah, J. Callan, and E. Rundensteiner, eds. McLean, VA, ACM Press, New York: 70–77.

[Jackson and Moulinier, 2002] Jackson, P. and Moulinier, I. (2002). *Natural language prosessing for online applications: Text retrieval, extraction and categorization,* Chapter 4: Text Categorization, pages 119 - 163 . John Benjamins Publishing Compagny, Amsterdam/Philadelphia, USA.

[Jaillet, 2004] Jaillet, S. (2004). Catégorisation automatique de documents. LIRMM UMR 5506, 161 rue Ada, 34392 Montpellier Cedex 5 – France URL : http://www.lirmm.fr/doctiss04/art/I02.pdf

[Jalam and Chauchat, 2002] Jalam, R. and Chauchat, J.-H.(2002). « Pourquoi les n-grammes permettent de classer des textes? Recherche de mots-clefs pertinents à l'aide des n-grammes caractéristiques ». In Morin, A. and Sébillot, P., editors, 6èmes Journées internationales d'Analyse statistique des Données Textuelles, volume 1, pages 381–390, St. Malo France. IRISA, INRIA URL : http://www.agrocampusrennes.fr/tice/jalam/pub/jalam_jadt02.ps

[Jalam and Teytaud, 2002] Jalam, R, Teytaud, O. (2002). Identification de la Langue et Catégorisation de Textes basées sur les N-grams, *Journées Francophones d'extraction et de gestion de connaissances*, France

[Jalam, 2003] Jalam, R. (2003). *Apprentissage automatique et catégorisation de textes multilingues*. Thèse de doctorat, Université Lumière, Lyon 2.

[Jardine and Rijsbergen, 1971] Jardine, N. and van Rijsbergen, C. (1971). The use of hierarchical clustering in information retrieval. *Information Storage and Retrieval*, 7:217–240, 1971.

[Jeong et al., 2004] Jeong, B., Lee, D., Lee, J., et Cho, H. (2004). Towards XML mining : The role of kernel methods.

[Joachims, 1997] Joachims, T. (1997). A probabilistic analysis of the Rocchio algorithm with tfidf for text categorization. In Proc. *ICML, pp. 143–151*. Morgan Kaufmann.

[Joachims, 1998] Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*. C. Nedellec andC. Rouveirol, eds. Chemnitz, Germany, Springer-Verlag, Heidelberg: 137–142.

[Joachims, 1999] Joachims, T. (1999).Transductive Inference for Text Classification Using Support Vector Machines. In *Proceedings of ICML-99, 16th International Conference on Machine Learning*. I. Bratko and S. Dzeroski, eds. Bled, Morgan Kaufmann Publishers, San Francisco: 200–209.

[Joachims, 2000] Joachims, T. (2000).Estimating the Generalization Performance of a SVM Efficiently. In *Proceedings of ICML-00, 17th International Conference on Machine Learning*. P. Langley, ed. Stanford, CA, Morgan Kaufmann Publishers, San Francisco: 431–438.

[Joachims, 2001] Joachims, T. (2001).A Statistical Learning Model of Text Classification with Support Vector Machines. In *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*. W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, eds. New Orleans, ACM Press, New York: 128–136.

[Joachims, 2002]Joachims, T. (2002a). Learning to Classify Text Using Support Vector Machines.Dordrecht, Kluwer Academic Publishers.

[Joachims, 2006a] Joachims, T. (2006a). Training linear SVMs in linear time.In *Proc. KDD*, pp. 217–226.ACM Press.doi: http://doi.acm.org/10.1145/1150402.1150429. [265, 302, 319]

[Joachims, 2006b] Joachims, T. (2006b). *Transductive support vector machines*. In Chapelle et al. (2006), pp. 105–118.

[Jurafsky and Martin, 2008] Jurafsky, D., and Martin, J. (2008). Speech and Language Processing: An Introduction to Natural Language Processing*, Computational Linguistics, and Speech Recognition*. Pearson, Upper Saddle River, 2008.

[Kadri and Moussaoui, 2013]Kadri, S, K., Moussaoui, A. (2013). An Effective Method to Recognize the Language of a Text in a collection of Multilingual Documents. *10th International Conférence on Electronics, Computer and Computation ICECCO 2013*, IEEE Conference, TurgutOzal University, Ankara, Turkey, 07-08 November 2013.

[Kadri and Moussaoui, 2014a] Kadri, S., Moussaoui, A. (2014a). Utilisation des Métriques de l'Identification de la Langue dans la Catégorisation Contextuelle de Documents.*1st International Symposium on Informatics and its Applications ISIA'14*, University of M'sila, M'sila, Algeria, 25-26 February, 2014, Conference Link : http://www.univ-msila.dz/ISIA14/

[Kanaan et al., 2005] Kanaan, G., Al-Shalabi, R., Al-Kabi, M. (2005). New Approach for Extracting Quadrilateral Arabic Roots," *Abhath Al-Yarmouk, Basic Science and Engineering*, 14(1): 51-66.

[Kantardzic, 2003] Kantardzic, M. (2003).*Data Mining: Concepts, Models, Methods, and Algorithms.* Wiley-IEEE Press, Piscataway, NJ.

[Khodja, 1999] Khodja S., Garside R. (1999). Stemming Arabic text. Technical report, Computing Department, Lancaster niversity, 1999., [online] available: http://www.comp.lancs.ac.uk/computing/users/khoja/stemmer.ps, last visited 1999.

[Khodja, 1999] Khodja S. (1999). Stemming Arabic Text. [online] Av: http://zeus.cs.pacificu.edu/shereen/research.htm, last visited 2014

[Kodratoff, 2001] Kodratoff, Y. (2001). *Machine Learning and Its applications*, Chapter comparing machine learning and knowledge discovery in databases: an application to knowledge discovery in texts, pages 1-21. Springer Verlag LNAI 2049.

[Kołcz and Yih, 2007] Kołcz, A., and Yih, W-T. (2007). Raising the baseline for high-precision text classifiers. In *Proc. KDD*.

[Kumar, 2007] Kumar, V. (2007). High Performance Data Mining: Application for Discovery of Patterns in the Global Climate System, Courses of university of Minnesota, USA.

[Lambert, 2000] Lambert, D. What Use is Statistics for Massive Data? In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discouery,* pages 54-62, 2000.

[Larkey and Connell, 2001] Larkey, L., Connell, M. (2001).Arabic information retrieval at UMass in TREC-10".*Proc.TREC 2001*, Gaithersburg: NIST, 2001.

[Larkey et al., 2002] Larkey, S., Ballesteros, L., Margaret, E., Connell, M. (2002). Improving Stemming for Arabic Information Retrieval: Light Stemming and Occurrence Analysis. In P*roc. 25th ACM International Conference on Research and Development on IR (SIGIR'02)*, Tampere, FIN, pp.275–282, 2002.

[Larkey et al., 2007] Larkey, S., Ballesteros, L., Margaret, M., Connell, E. (2007). Light Stemming for Arabic Information Retrieval: Arabic Computational Morphology Text. *Speech and Language Technology*, Vol. 38, pp.221-243, 2007

[Lebart, 2001] Lebart, L. (2001). Classification of textual Data. Séminaire de recherche à : the School of Computer Science and information Systems, Birkbeck College. Available on :http://www.dcs.bbk.ac.uk/seminars/autumn01.html.

[Ledmi, M] Ledmi, M. (2010). *Classification automatique des documents XML.* Thèse de magister en Informatique. Centre universitaire Abbas Laghrour, Khenchela, Algérie

[Lee et al., 2001] Lee, J., Lee, K. et Kim, W. (2001). Preparations for semantics-based

[lefèvre, 2000] Lefèvre, P. (2000). *La recherche d'information du texte integral au thesaurus*. Hermès Science, Paris

[Lewis and Ahn, 1995] Lewis,D.D., Ittner, D. J., and Ahn, D.D. (1995). Text Categorization of LowQuality Images. In *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval*. Las Vegas, NV, ISRI, University of Nevada, Las Vegas, NV: 301–315.

[Lewis and Ringuette, 1994] Lewis, D. D., and Ringuette, M. (1994).A Comparison of Two Learning Algorithms for Text Categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*. Las Vegas, NV, IRSI, University of Nevada, Las Vegas: 81–93.

[Lewis, 1992] Lewis, D. (1992). Feature selection and feature extraction for text categorization. In *Proceedings of the Speech and Natural Language Workshop*, pages 212–217. Morgan Kaufmann, San Francisco.

[Lewis, 1992] Lewis, D.D. (1992a). An evaluation of phrasal and clustred representation on text categorization task.In Proceedings of SIGIR-92, 15th ACM international conference on research and development on information retrieval. Pages 37-50, Kobenhaven, DK, ACM press, New York, USA.

[Lewis, 1998] Lewis, D. D. (1998). Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*. C. N'edellec andC. Rouveirol, eds. Chemnitz, Germany, Springer-Verlag, Heidelberg: 4–15.

[LI and Jain, 1998] Li, Y. H., and Jain, A. K. (1998).*Classifiation of text documents*. Department of Computer Science and Engineering, Michigan State University, East Lansing, Michigan, USA URL : http://comjnl.oxfordjournals.org/cgi/content/abstract/41/8/537

[Li and Yamanishi, 2002] Li, H., and Yamanishi, K. (2002).Text Classification Using ESC-based Stochastic Decision Lists.*Information Processing and Management.* 38(3): 343–361.

[LI et al., 2009] Li, S., Xia, R., Zong, H., Huang, C-R., (2009). A framework of feature selection methods for text categorization. In proceedings of the 47th annual meeting of the ACL and the 4th IJCNLP of the AFNLP, pages : 692-700, Suntec, Singapore

[Lini et al., 2001] Lini, L., Lombardini, D., Paoli, M., Colazzo, D. et Sartiani, C.

[Liu and al., 2002] Liu, Y., Yang, Y., and Carbonell, J. (2002).Boosting to Correct the Inductive Bias for Text Classification.In *Proceedings of CIKM-02, 11thACMInternational Conference on Information and Knowledge Management*. McLean, VA, ACM Press, New York: 348–355.

[Liu, 2004] Liu, Y. (2004). A comparative study on feature selection methods for drug discovery.*J.Chem.Inf.Comput.Sci*, Vol. 44, pages:1823-1828

[Luhn, 1959] Luhn, H. (1959). Auto-encoding of documents for information retrieval systems. In M. Boaz, editor, *Modern Trends in Documentation*, pages 45–58. Pergamon Press, London.

[Maldenic and Grobelnik, 1998] Maldenic, D. and GRobelnik, M. (1998). Word sequences as features in text learning. In *Proceedings of ERK-98, the seventh Electrotechnical and computer science conference*, pages: 145-148, Ljubljana, SL.

[Maltese and Mancini, 1991] Maltese, G., and Mancini, F. (1991).A Technique to Automatically Assign Parts-of-Speech to Words Taking into Account Word-Ending Information through a Probabilistic Model.In *Proceedings of Eurospeech 1991*. Genoa, Italy, Genovalle Institute fuer Kommunikations Forschung und Phonetick, Bonn, Germany: 753–756.

[Manning et al., 2008] Manning, C., Raghavan, P., Schutze, H. (2008). *Introduction to information retrieval*.Cambridge University press.

[Maron and Kuhns, 1960] Maron, M. E., and Kuhns, J.L. (1960).On relevance, probabilistic indexing, and information retrieval.*JACM* 7(3):216–244.

[Maron, 1961] Maron, M. E. (1961). Automatic Indexing: An Experimental Inquiry. *Journal of the Association for Computing Machinery* , 8(3): 404–417.

[Mathieu, 2000] Mathieu, S. (2000). *Réseaux de neurones pour le traitement automatique du langage : conception et réalisatiion de filters d'informations ».* Thèse de Doctorat de l'Université Pierre et Marie Curie - Paris VI soutenue le 19 Décembre 2000, a été effectuée au Laboratoire d'Électronique de l'ESPCI (Paris), URL : http://www.neurones.espci.fr/

[McCallum and Nigam, 1998] McCallum, A. K., and Nigam, K. (1998).Employing EM in Pool-Based Active Learning for Text Classification. In *Proceedings of ICML-98, 15th International Conference on Machine Learning*. J. W. Shavlik, ed. Madison, WI, Morgan Kaufmann Publishers, San Francisco: 350–358.

[McCallum and Nigam, 1998] McCallum, A., and Nigam, K. (1998). A comparison of event models for Naive Bayes text classification. In *Working Notes of the 1998 AAAI/ICML Workshopon Learning for Text Categorization*, pp. 41–48.

[Miline and Keefe, 2012] R.M. Milne, R.A. O'Keefe, A. Trotmana, "A study in language identification, ADCS December 05 - 06 2012, Dunedin, New Zealand, 2012

[Miller et al., 1999] Miller, E., Shen, D., Liu, J., Nicholas, C. (1999). Performance and Scalability of a Large-Scale N-gram Based IR Sys.*Journal of Digital Information*, 1(5).

[Mitchell, 1997] Mitchell, T. (1997).*Machine Learning*.McGraw-Hill, Boston, MA, 1997.

[Momani and Faraj, 2007] Momani, M., Faraj, J. (2007).A novel algorithm to extract tri-literal Arabic roots," In *Proc.IEEE/ACS International Conference on Computer Systems and Applications*, May 13-16, EEE Xp ress, Amman, pp.309-315, 2007.

[Mouliner, 1997] Moulinier, I. (1997).Feature Selection: A Useful Preprocessing Step. In *Proceedings of BCSIRSG-97, 19th Annual Colloquium of the British Computer Society InformationRetrieval Specialist Group*. J. Furner and D. Harper, eds. Aberdeen, UK, Springer-Verlag, Heidelberg, Germany: 1–11.

[Moulinier, 1996] Moulinier, I. (1996). *Une approche de la catégorisation de textes par l'apprentissage symbolique*. Thèse de doctorat, université Paris 6, paris

[Munoz et al., 1999] Munoz, M., Punyakanok, V., Roth, D., and Zimak, D. (1999). A Learning Approach to Shallow Parsing. Technical Report 2087, University of Illinois at Urbana-Champaign: 18

[Muslea, 1999] Muslea, I., Minton, S. and Knoblock, C. (1999). Hierarchical Wrapper Induction for Semi Structured Information Sources. *Journal of Autonomous Agents and Multi-Agent Systems*.

[Mustonen, 1965] Mustonen, S. (1965). Multiple Discriminant Analysis in Linguistic Problems, Statistical Methods in Linguistics, pp 147, 195—197, Page visitée le 15 juin 2000 à l'adresse http://www.nodali.sics.se/bibliotek/kval/smil, 1965.

[Nakache, 2007] Nakache, D. (2007). *Extraction automatique des diagnostics à partir des comptes rendus médicaux textuels».* Thèse de doctorat soutenu le 26 septembre 2007, URL : *http://pagesperso-orange.fr/nakache/doctorat/*

[Nedjeh et al, 2009] Nedjeh, N.; de Macedo, M . L. ; Kacprzyk, J. ; França, M.G.F, de Souza, A.F. (2009). *Intelligent text categorization and clustering*. Scientific Publishing Services, Chennai, India.

[Ng et al., 1997] Ng, H.T., Goh, W.B., and Low, K.L. (1997). Feature selection, perceptron learning and a usability case study for text categorization. IN Proceedings of SIGIR-97, 20th ACM international conference on research and development in information retrieval, pages 67-73, Philadelphia, USA, ACM press, New York.

[Nicolas, 2009] Nicolas, B. (2008). *Extraction et regroupement de descripteurs morphosyntaxiques pour des processus de Fouille de Textes* . Thèse de doctorat Soutenue le 8 décembre 2008, présentée à l'Université des Sciences et Techniques du Langue doc pour obtenir le diplôme de DOCTORAT URL : *http://tel.archives-ouvertes.fr/tel-00462206/fr/*

[Nigam et al, 2000] Nigam, K., and Ghani, R. (2000).Analyzing the Applicability and Effectiveness of Co-training.In *Proceedings of CIKM-00, 9thACMInternational Conference on Information and Knowledge Management*. A. Agah, J. Callan, S. Gauch, and E. Rundensteiner, eds. McLean, VA, ACM Press, New York: 86–93.

[Olteanu, 2002] Olteanu, D., Meuss, H. Furche, T. and Bry, F. (2002). XPATH: Looking Forward Lecture Notes in Computer Science, volume 24, pages 109-127.

[Pang_Ning et al., 2013] Pang-Ning T., Steinbach, M., Kumar, V. (2013).*Introduction to Data Mining*. Pearson/Addison Publisher, USA.

[Paulheim et al, 2014] Paulheim, H., Bryl, V., Meusel, R., Lehmberg, O. (2014). *Data Mining I: Text Mining*, slides of courses, University of Mannheim.

[Paulheim, 2015] Paulheim, H. (2015). Data Mining I: Introduction and course Outline, University of Mannheim, July 2015.

[Peter and Sheridan, 2001] Peters, C. and Sheridan, P. (2001). Accès multilingue aux systèmes d'information. In *67th IFLA Council and General Conference,* August 16-25, 2001 URL :http://www.ifla.org/IV/ifla67/papers/099-183f.pdf

[Pham et al., 2008] Pham, T., Chevallet, J-P., Hwee, L-J. (2008). *Fusion de multi-modalités et réduction par sémantique latente.* Laboratoire Image Perception Access & Language. (IPAL) - UMI CNRS 2955 21 Heng Mui Keng Terrace, 119613, Singapore URL : http://asso-aria.org/coria/2008/39.pdf

[Platt, 1998] Platt, J.C. (1998). Sequential Minimal Optimization: A fast algorithm for training support vector machines. Technical report. Microsoft Research, USA

[Platt, 2000] Platt, J., (2000). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *A.J. Smola,* P.L. Bartlett, B. Sch¨olkopf, and D. Schuurmans (eds.)*, Advances in Large Margin Classifiers*, pp. 61–74. MIT Press.

[Porter, 1980] Porter, M. (1980).An algorithm for suffix stripping.*Program*, 14(3):130–137, 1980.

[Punyakanok, 2000] Punyakanok, V., and Roth, D. (2000).Shallow Parsing by Inferencing with Classifiers.In *Proceedings of the 4th Conference on Computational Natural Language Learning and of the 2nd Learning Language in Logic Workshop*. Lisbon, Association for Computational Linguistics, Somerset, NJ: 107–110.

[Quilan, 1986] Quinlan, J.R. (1996). (1986). Induction of Decision Trees.*Machine Learning*, 1(1):81–106, 1986.

[Quilan, 1996] Quinlan, J.R. (1996). Improved Use of Continuous Attributes in C4.5. *Journal of Artificial Intelligence Research*, 4:77–90..

[Rakotomalala, 2005] R. Rakotomalala, ''Arbres de décision, *Revue MODULAD*, 2005, N°33

[Ranka, 2011] Ranka, S. (2011).Introduction to Data mining. Slides of courses, University of Florida, Spring 2011, USA.

[Ratnaparkhi, 1995] Ratnaparkhi, A. (1995). A maximum entropy part-of-speech tagger.*Computational Linguistics*, 21(4):543–565, 1995. http://www.cis.upenn.edu/~adwait/penntools.html.

[Ratnaparkhi, 1999] Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models.*Machine Learning*, 34:151–178, 1999.

[Réhel, 2005] Réhel, S. (2005). *Catégorisation automatique de textes et cooccurrence de mots provenant de documents non étiquetés »* Maître de sciences (M.Sc.), Université Laval, faculté des sciences et de génie, Maîtrise en informatique, disponible sur la page Web : http://www.web-mining.fr/liens

[Rocchio, 1971] Rocchio, J. J. (1971). Relevance feedback in information retrieval.In Salton (1971b), pp. 313–323.

[Sahami et al., 1998] Sahami, M., Yusufali, S., and Baldonado, M. Q. (1998). SONIA: A Service for Organizing Networked Information Autonomously. In *Proceedings of DL-98, 3rd ACM Conference on Digital Libraries*. I.Witten, R. Aksyn, and F. M. Shipman, eds. Pittsburgh,ACMPress, New York: 200–209.

[Sahami, 1998] Sahami, M., ed. (1998).Learning for Text Categorization.Papers from the 1998 AAAIWorkshop.Madison, WI, AAAI Press, Menlo Park, CA.

[Sahami, 1999] Sahami, M. (1999).*Using Machine Learning to Improve Information Access*, PhD thesis, Computer Science Department, Stanford University, 1999.

[Salton and Buckley, 1987] Salton, G., and Buckley, C. (1987).*Term weighting approaches in automatic text retrieval*. Technical report, Cornell University, Ithaca, NY. [122]

[Salton and Buckley, 1988] Salton, G., and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *IP&M* 24(5):513–523. [123]

[Salton and McGill, 1983] Salton, G. and McGill, M. (1983).*Introduction to modern information retrieval*. McGill-Hill, New York.

[Salton, 1964] Salton, G. (1964). A document retrieval system for man-machine interaction. In *Proceedings of the 19th Annual International ACM National Conference*, pages L2.3.1–L2.3.20. ACM, New York.

[Salton, 1965] Salton, G. and Lesk, M. (1965). The SMART automatic document retrieval system: An illustration. *Communications of the ACM*, 8(6):391–398.

[Salton, 1971] Salton, G. (1971). *The SMART Retrieval System*.Prentice-Hall, Englewood Cliffs.

[Salton, 1975] Salton, G, Wong, A. and Yang, C. (1975). A vector space model for automatic indexing.*Communications of the ACM*, 18:613–620.

[Salton, 1980] Salton, G. and Wu, H. (1980). A term weighting model based on utility theory. In *Proceedings of SIGIR*, pages 9–22. ACM, New York.

[Sawalha and Atwell, 2008] Sawalha, M., Atwell, E. (2008).Comparative Evaluation of Arabic Language Morphological Analyzers and Stemmers. In *Proc. COLING-ACL*, 2008

[Schannon, 1948] Shannon, C. (1948). The MathematicalTheory of Communication, Bell System *Technical Journal*, vol27, 1948, p379–423/623–656.

[Schmid, 1994] Schmid, H. (1994).Probabilistic part-of-speech using decision trees. *In International conference on new methods in language processing,* Manchester, UK.

[Schôlkopf and Smola, 2001] Sch¨olkopf, B. and Smola, A.J. (2001).*Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press.

[Schütze *et al.*, 1995] Schutze, H., Hull,D. A., and Pedersen, J.O. (1995). A Comparison of Classifiers and Document Representations for the Routing Problem. In *Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval*. E. A. Fox, P. Ingwersen, and R. Fidel, eds. Seattle, ACM Press, New York: 229–237.

[Schutze, 1993] Schutze, H. (1993). Part-of-Speech Induction from Scratch.In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*. Columbus, OH, Association for Computational Linguistics, Morristown, NJ: 251–258.

[Scime, 2005] Scime, A. (2005). *Web Mining: Applications and Techniques*, State University of New York College at Brockport, USA, IDEA Group Publishing

[Sebastiani, 1999] Sebastiani, F.(1999).A Tutorial on Automated Text Categorization .In *Proceedings of ASAI-99, 1st Argentinian Symposium on Artificial Intelligence* , pages 7-35, Buenos Aires, AR. Available on : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.38.7770.pdf

[Sebastiani, 2002] Sebastiani, F. (2002). Machine Learning in Automated Text Categorization.*ACM Computing Surveys* **34**(1): 1–47.

[Shapire, 1990] Schapire, R. E. (1990). The strength of weak learnability.*Machine Learning*, 5(2):197–227.

[Shawe-Taylor and Cristianini, 2004] Shawe-Taylor, J., and Cristianini, N. (2004).*Kernel Methods for Pattern Analysis*.Cambridge University Press.

[Sibun and Reynar, 1996] Sibun, P., Reynar, J. (1996). Language Identification: Examining the Issues. *Symposium on Document Analysis and Information Retrieval*, Las Vegas, 1996, pp. 125–135.

[Singhal, 1996a] Singhal, Amit, Chris Buckley, and Mandar Mitra. 1996a. Pivoted document length normalization. In Proc. SIGIR, pp. 21–29.ACM Press.url: citeseer.ist.psu.edu/singhal96pivoted.html. [122] [Ng et al., 2000]

[Slonim and Tishbi, 2001] Slonim, N., and Tishby, N. (2001).The Power of Word Clusters for Text Classification. In *Proceedings of ECIR-01, 23rd European Colloquium on Information Retrieval Research*. Darmstadt, Germany Academic Press, British Computer Society, London.

[Smyth, 2001] Smyth, P. (2001). Breaking out of the Black-Box: Research Challenges in Data Mining. In *Proc. of the 2001 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discoue'1.*

[Song et al., 2009] Song, M., Wu, Y-f B. (2009). *Handbook of Research on Text and Web Mining Technologies*, Information Science Reference, Hershey, New York, Copyright © 2009 by IGI Global, ISBN 978-1-59904-990-8 (hardcover) -- ISBN 978-1-59904-991-5 (ebook).

[Souter et al, 1994] Souter, C., Churcher, C., Hayes, G., Hughes, J., Johnson, J. (1994). Natural Language Identification Using Corpus-Based Models, *Hermes Journal of Linguistics*, vol. 13, 1994.

[Steinbach et al., 2004] Steinbach, T., Kumar, V. (2004). Introduction to data mining: lecture notes for chapter 1 (pdf courses). Michigan State University, USA

[Sumathi and Sivanandam, 2006] Sumathi, S., Sivanandam, S.N. (2006*). Introduction to Data Mining and its Applications*, Springer-Verlag, Berlin Heidelberg, Ger.

[Tan and Cheng, 2007] Tan, S., and Cheng, X. (2007).Using hypothesis margin to boost centroid text classifier.In *Proc. ACM Symposium on Applied Computing*, pp. 398–403.ACM Press.doi: http://doi.acm.org/10.1145/1244002.1244096.

[Tsochantaridis, 2005] Tsochantaridis, I., Joachims, T., Hofmann, T. and Altun, Y. (2005).Large margin methods for structured and interdependent output variables.*JMLR* 6:1453–1484.

[Tzoukermann, 1997] Tzoukermann, E., Klavans, J., and Jacquemin, C. (1997).Effective Use of Natural Language Processing Techniques for Automatic Conflation of Multi-Word Terms: The Role of Derivational Morphology, Part of Speech Tagging, and Shallow Parsing. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Philadelphia, ACM Press, New York: 148–155.

[Uren, 2001] Uren, V. (2000).An evaluation of text categorization errors.In *proceedings of the one-day workshop on evaluation of information management systems*.Pages 79-87, London, UK, Queen Mary and Westfield College.

[Vapnik, 1995] Vapnik, V. (1995).The Nature of Statistical Learning Theory.Berlin, Springer-Verlag.

[Vapnik, 1998] Vapnik, V. N. (1998). *Statistical Learning Theory*.Wiley-Interscience.

[Weiss et al., 2010] Weiss, S.M., Indurkhya, N., Zhang, T. (2010). *Fundamentals of Predictive Text Mining*, Editors David Gries and Fred B. Schneider, Springer, ISBN 978-1-84996-225-4, DOI 10.1007/978-1-84996-226-1, © Springer-Verlag London Limited 2010.

[Wiener *et al.*, 1995] Wiener, E. D. (1995). A Neural Network Approach to Topic Spotting in Text.Boulder, CO, Department of Computer Science, University of Colorado at Boulder.

[Wisniewski et al., 2005] Wisniewski, G., Denoyer, L. et Gallinari, P. (2005). Classification automatique de documents structurés. Application au corpus d'arbres étiquetés de type XML. In *2ème Conférence en Recherche d'Informations et Applications (CORIA'05)*, Grenoble.

[Witten and Frank, 2005] Ian, H. W., Eibe, F. (2005).*Data Mining: Practical Machine Learning Tools and Techniques*, 3rd Edition, Morgan Kaufmann Publisher, USA.

[Witten, 2004] Witten, I., Moffat, A. and Bell, T. (2004). Managing gigabytes. Chapter 5: Information retrieval and web search, Cambridge University press, UK.

[Wu et al., 2000] Wu, X., Yu, P.S. and Piatetsky-Shapiro, G. (2000). Data Mining: How Research Meets Practical Development? *Knowledge and Information Systems,* 5(2):248-261, 2003.

[Xu and Croft, 1998] Xu, J. and Croft, B. (1998).Corpus-based stemming using cooccurrence of word variants.*ACM Topics on Information Systems*, 16(1):61–81, 1998.

[Xu and Kumar, 2009] Xu, W., and Kumar, V. (2009).*The top ten algorithms in data mining*, Champan&Hall Book/CRC Press, Data mining and knowledge discovery series, Taylor&Francis group LLC, USA

[Yang and Chute, 1993] Yang, Y., and Chute, C. G. (1993). An Application of Least Squares Fit Mapping to Text Information Retrieval. In *Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval*. R. Korthage, E. Rasmussen, and P. Willett, eds. Pittsburgh, ACM Press, New York: 281–290.

[Yang and Chute, 1994] Yang, Y., and Chute, C. G. (1994)."An Example-Based Mapping Method for Text Categorization and Retrieval."*ACM Transactions on Information Systems* 12(3): 252–277.

[Yang and Liu, 1999] Yang, Y., and Liu, X. (1999).A Re-examination of Text Categorization Methods. In *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*. M. Hearst, F. Gey, and R. Tong, eds. Berkeley, CA,ACMPress, New York: 42–49.

[Yang and Pedersen, 1997] Yang, Y., and Pedersen, J. O. (1997).A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of ICML-97, 14th International Conference on Machine Learning*. D. H. Fisher. Nashville, TN, Morgan Kaufmann publishers, San Francisco: 412–420.

[Yang and Zhang, 2007] Yang, J. et Zhang, F. (2007). XML document classification using extended vsm. In *INEX*, pages 234–244.

[Young-Min et al., 2008] Young-Min, K ., Pessiot, J-F., Amini, M-R., Gallinari, P. (2008). Apprentissage d'un espace de concepts de mots pour une nouvelle représentation des données textuelles. In *Proceedings of the 5th Conférence en Recherche d'Information et Applications*, 12-14 Mar 2008, Trégastel, France

[Young-Min, 2008] Young-Min, K ., Pessiot, J-F. , Amini, M-R., Gallinari, P., (2008). *Apprentissage d'un espace de concepts de mots pour une nouvelle représentation des données textuelles*. in*Proceedings of the 5th Conférence en Recherche d'Information et Applications*, 12-14 Mar 2008, Trégastel, France.

[Yousef et al., 2014] Yousef, N., Abu-Errub, A., Odeh, A., Khafejeh, H. (2014).An Improved Arabic Word's Roots Extraction Method Using N-gram Technique.*Journal of Computer Science*, 10(4).

[Yousef et al., 2010] Yousef, N., Al-Bidewi, I., Fayoumi, M. (2010). Evaluation of different query expansion techniques and using different similarity measures in Arabic documents. *European Journal of Scientific Research.,* vol. 43, pp.156-166.

[Zaki and Wagner, 2014] Zaki, J.M., Wagner, M.J.R. *Data mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University press, UK.

[Zampieri and Gebre, 2012] M. Zampieri, B.G. Gebre, "Automatic identification of language varieties: the case of Portuguese. In *Proceedings of KONVENS 2012* (Main track: poster presentations), Vienna, September 20.

[Zaragoza, 1999] Zaragoza, H., Massih-Reza, A., and Gallinari, P. (1999).A Dynamic Probability Model for Closed-Query Text Mining Tasks.*Draft submission to KDD '99*.

[Zhang and al., 2007] Zhang, H., Spiliopoulou, M., Mobasher, B., Lee Giles, C., McCallum, A., Nasraoui, O., Srivastava, J., Yen, J. (2007). In *proceedings of Advances inWeb Mining andWeb Usage Analysis. 9th International Workshop on Knowledge Discovery on theWeb,WebKDD 2007and 1st International Workshop on Social Networks Analysis*, SNA-KDD 2007 San Jose, CA, USA, August 12-15, 2007.

[Zhang and Yang, 2003] Zhang, J., and Yang, Y. (2003).Robustness of Regularized Linear Classification Methods in Text Categorization. In *Proceedings of SIGIR-03, 26th ACM International Conference on Research and Development in Information Retrieval*. J. Collan, G. Cormack, C. Clarke,

[Zytkow and Rauch, 1999] Zytkow, J.M., Rauch, J. (1999). Principles of Data Mining and Knowledge Discovery, In *proceedings of the3rd European Conference, PKDD'99*, Prague, Czech Republic, September 15-18.http://nmis.isti.cnr.it/sebastiani/Publications/ACMCS02.pdf