Naive Bayesian Learning from Structural Data

Michelangelo Ceci

Dipartimento di Informatica
Università degli Studi di Bari
ceci@di.uniba.it

Promotor: Prof. Donato Malerba

A dissertation submitted in partial satisfaction of the requirements
for the degree of
Doctor of Philosophy in Computer Science
in the Graduate Division of the
University of Bari, Italy

Credits

This dissertation was typeset using these shareware programs:

- TeXnicCenter 1 Beta 6.21
  available at: `http://www.texniccenter.org/`
- MikTEX 2.1
  available at: `http://www.miktex.de`

Thesis Supervisor

Prof. Donato Malerba

Chairperson of the Supervisory Committee

Member of the Supervisory Committee

Member of the Supervisory Committee

Submitted *January 2005*

Contents

List of Figures

List of Tables

Acknowledgments

First, I would like to thank my adviser Prof. Donato Malerba for his support since I was a university student. He gave me the opportunity to embark on this work and taught me the importance of the research. I have been inspired to his passion in his work, his brilliance and his dedication and I have benefited greatly from his example and guidance.

I would like to thank Prof. Floriana Esposito, chair of the LACAM (Knowledge Acquisition & Machine Learning Lab), for the opportunities she gave me and for the useful guidance and suggestions. I would also like to thank Prof. Peter A. Flach who supervised my work during my period abroad giving me useful suggestions with his brilliance and competence.

The work in this thesis is the product of collaboration with a number of people. Donato, who has not just supervised this work, but has been concretely involved in its development. Annalisa Appice, Margherita Berardi and Antonio Varlaro who are three phenomenal colleagues and were concretely involved in its development. In the thesis I'll use the term "we" instead of "I" in order to remark their contribution.

I would also like to thank all my colleagues of the LACAM lab and all my colleagues of the ML group at Bristol who collaborated with me (Marco, Pasquale, Paolo, Simon, Gianni, Francesca, Oronzo, Oriana, Nico, Stefano, Nico, Mara, Vincenzo, Tom, etc...) for their support and for being patient with me.

Finally but not least, I would like to thank my girlfriend, Marina, for sharing my joys and tears during my Ph.D. studies, my parents for the kind of encouragement and support they have provided me and friends who were close to me during this period (*you know who you are*).

Abstract

The amount of data being collected in databases today far exceeds our ability to analyze them without the use of automated analysis techniques. Data Mining is evolving to provide automated analysis solutions and is defined as the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data and describing them in a concise and meaningful manner. Data mining is a discipline lying at the interface of statistics, database technology, pattern recognition, machine learning, and other areas. In recent years increasing attention has been given to probabilistic and statistical approaches that historically provide an intellectual background to the analysis of collected data when uncertainty in data has to be taken into account and when is not possible to create complete and consistent model of the world. One of the most widely used statistical methods in Data Mining for classification tasks is the Naive Bayes Classifier that, despite of its strong assumptions, has been proved to be useful in many application domains.

Studies on Naive Bayes Classifiers, like most studies in Data Mining have focused on a relatively simple representation of data: a database relation, or a standard data table, or a set of points in a feature space. In fact, the relational model is clean and simple, and a relational table can be easily mapped into the mathematical concept of matrix. However, with the advent of the information age, we have witnessed to a dramatic growth of applications in government, business, education and science, many of which are sources of various data, organised in different structures and formats. The chances that computers have provided have enlarged the meaning of "data", have defined new sorts of problems in knowledge discovery, and are leading to the development of completely new classes of models and data analysis algorithms that take the "structure" of data into account.

The structure of data can be in various forms. In this work we consider two common interpretations of structured data: the occurrence of relations between categories of the units of analysis, that is, between the principal entities of a statistical study (categorization structure) or the occurrence of relations between the units of analysis and/or the units observation, that is, the secondary entities of the statistical study that are correlated with the units of analysis (unit structure).

In this thesis we face and deeply investigate the problem of Naive Bayesian learning from these two forms of structured data. In particular, for the case of categorization structure, we propose a framework for the usage of Naive Bayes classifiers in the case of hierarchically related categories, while, for the case of unit structure, we resort to a multi-relational approach to Data Mining.

A principle that guided the writing of this thesis was that it should present a balance of theory and practice. In particular, proposed algorithms have been empirically evaluated on real datasets and applications principally concern the field of Document Engineering. Document Engineering is the computer science discipline that investigates systems for documents in any form and in all media and is concerned with principles, tools and processes that improve our ability to create, manage, and maintain documents. It includes, among other topics, Text Categorization, Document Image Classification, Document Retrieval and Document Understanding.

Chapter 1

Introduction

The amount of data being collected in databases today far exceeds our ability to analyze them without the use of automated analysis techniques. The field of knowledge discovery in databases (KDD) is evolving to provide automated analysis solutions. Knowledge discovery is defined as the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data and describing them in a concise and meaningful manner. This process is interactive and iterative, involving numerous steps. Information flows forwards from one stage to the next, as well as backwards to previous stages. The main step is data mining.

Data mining automates the process of finding relationships and patterns in raw data and delivers results that can be either utilized in an automated decision support system or assessed by a human analyst. A data mining task consists of analysing data collected in databases in order to help answer questions such as:

- What goods should be promoted to this customer?
- What is the probability that a certain customer will respond to a planned promotion?
- Can one predict the most profitable securities to buy/sell during the next trading session?
- Will this customer default on a loan or pay back on schedule?
- What medical diagnose should be assigned to this patient?
- How large the peak loads of a telephone or energy network are going to be?
- Why the facility suddenly starts to produce defective goods?
- What is the main topic of a textual (web) document?
- Can one predict whether a molecule is mutagenic or not?

One of the fundamental tasks in data mining is Classification. In the usual classification setting, input or training data consists of multiple examples, each having multiple attributes or features. Each example is tagged with a class label. The goal

is to learn the target concept associated with each class by finding regularities in examples of a class that characterize the class in question and discriminate it from the other classes. This problem has been extensively studied in the literature and several and disparate approaches have been proposed.

Most of proposed solutions for classification are able to classify a new untagged example in one of the possible classes on the basis of an induced complete and consistent model of the world. However, in many problem domains it is not possible to create complete and consistent models of the world. Therefore it is necessary to act in uncertain worlds (which the real world is). Furthermore, the very act of preparing knowledge to support Data Mining tasks requires that we leave some facts unknown, unsaid or approximately summarized. For example, if we encode the knowledge about the "satisfaction of a customer" in a rule, the rule will have many exceptions which we cannot afford to enumerate and the conditions under which the rules apply are usually ambiguously defined and difficult to satisfy in real life.

A way to act taking the uncertainty into account is by means of statistical approaches for learning. In statistical approaches for learning, a "belief" is associated to the decision taken. It is often based on the attempt to draw statistical conclusions from the conditional probability $P(H|E)$, that is the probability of an hypothesis H (to be true) given that the event E has been observed (to be true). This idea is called Bayesian statistics and derives from the so-called "Bayes Theorem" (Thomas Bayes-1763) [Bay63]. The Bayes theorem paves the way to the idea to use a concept of intuitive probability in statistical theory and practice.

One of the most studied approaches in Bayesian statistics for classification purposes is the Naive Bayes classifier. The naive Bayes classifier is based on the estimation of the posterior probability that an example belongs to a class according to the Bayesian statistical framework. The naive Bayes classifier is also based on the assumption that, given the class, attributes are independent each other. This assumption is clearly false if the predictor variables are statistically dependent. However, even in this case, Domingos and Pazzani [DP97] empirically and formally proved that the naive Bayesian classifier can give good results. Due to its simplicity and its performances in large-scale datasets, it is used in a wide range of applications.

This thesis faces the problem of mining Naive Bayes statistical classifiers in presence of structured data taking into account different aspects related to both theoretical and applicative problems. In particular, several aspects have been investigated and different algorithms have been also proposed. Experiments mainly concern the field of Document Engineering. Document Engineering is the computer science discipline that investigates systems for documents in any form and in all media. It is concerned with principles, tools and processes that improve our ability to create, manage, and maintain documents.

1.1 Motivations

The problem of mining statistical classifiers has been extensively investigated in the literature. However, the rapid growth of information available and new and more complex applications demand the use of more sophisticated approaches that are able to deal with both complex data and previously known background knowledge.

Studies on Naive Bayes Classifiers, like most studies in Data Mining, have focussed on a relatively simple representation of data: a database relation, or a standard data table, or a set of points in a feature space. In fact, the relational model is clean and simple, and a relational table can be easily mapped into the mathematical concept of matrix. However, with the advent of the information age, we have witnessed to a dramatic growth of applications in government, business, education and science, many of which are sources of various data, organized in different structures and formats. The chances that computers have provided have enlarged the meaning of "data", have defined new sorts of problems in knowledge discovery, and are leading to the development of completely new classes of models and data analysis algorithms that take the "structure" of data into account.

The structure in structured data can be in various forms [BCM00]. A first type of structured data is represented by *tree-structured* or taxonomic attributes, that is attributes whose domain values are ordered in a rooted hierarchical tree. In data mining taxonomies are used to support generalisation-based knowledge discovery or attribute-oriented induction [HCC92] in order to reduce the computational complexity of the mining algorithms, while in machine learning taxonomies define some form of background knowledge to be used during the learning process [AAK95]. A different, but someway related, form of structure in the attribute domain is that of *relational variables/attributes*, as they are referred to in the field of data analysis, that are characterized by the definition of a dissimilarity matrix on the domain values [Ler00].

In all these examples the "structure" is in the attribute domain, that is, in the definition of a possibly weighted binary relationship defined on the value set. In symbolic data analysis [BD00] another type of binary relation is considered, which involves a variable describing an observed object and the set of values that the variable can take. In general the relation can be an order/equality relation (e.g., "number-of-inhabitants $\leq$ 100") or set inclusion (e.g., "gross-national-product $\subseteq$ [40,50]"). This different type of "structure" is very useful when the unit of analysis is not a single individual but a class (or group) of individuals. For instance, the description of a group of daily connections to a department network can be obtained by *aggregating* the values of the attributes *Destination-IP*, *Nation-Time-Zone* and *Start-Hour*. The result is a conjunction of different binary relations involving the three variables and (a set of) domain values (e.g. "Destination-IP $\leq$ 87 AND Nation-Time-Zone $>$ -4 AND Start-Hour $\in$ [22..24]"). How to extract patterns from this kind of "structured data" describing different groups of individuals is indeed the main goal of the research area known as symbolic data analysis.

Another form of "structure" in the data is represented by *dependencies between variables* or attributes. In the case of hierarchical pairwise variable dependencies

the set of values taken by a variable Y depends on the set of values taken by another value X (e.g. "if obj-type=river then color=blue" or "if gender=male then number-of-pregnancies=not-applicable"). More in general a weight can be associated with each dependence. It represents the "strength" of the dependence and often corresponds to a probability value, such as in the case of probabilistic causal relationships (e.g. "if driving-speed=very-high then mortal-accident=yes with probability 0.4"). A further generalization of this type of "dependency structure" is represented by probabilistic graphical models, such as Bayesian networks, which are characterised by the fact that each variable is directly influenced by only a few others (e.g. "both student's intelligence and difficulty of material affect the degree of understanding of a subject").

Both taxonomic and symbolic data are extensions of *classical data tables*, where objects are described by a fixed set of attribute-value pairs, possibly with some form of attribute dependencies. By representing the units of analysis as rows of the data table and attributes as columns, we can easily see that all types of "structures" presented above affect either a single column, or multiple columns, but they never express some kind of dependence between rows. The term "relational data" has been used by the data analysis community to introduce a different type of "structure" concerning a relationship between each pair of objects. The most common case of relational data is when we have (a matrix of) dissimilarity data between objects, each of which can be described by the same fixed set of attributes [HB02]. Technically, this kind of "relational data" must be represented by two tables of a relational database, one describing the objects to be analysed and the other describing the relations (e.g., the dissimilarity matrix) between them.

Recently, the term *relational data* has also been adopted by the data mining community to refer to the more general (and complex) case in which multiple relationships exist between objects, which can even be described by different sets of attributes. This means that the unit of analysis is not necessarily a single row of a data table but is composed by multiple rows in multiple tables. In this "structured" unit of analysis it is necessary to distinguish the *target object* of analysis (rows of a *target table* representing the principal entities under study) from the other task *task relevant* objects: discovered patterns (e.g., generalizations) must refer to target objects which may or may not have some relationships with other task relevant objects. Studies on how to analyse or mine this kind of structured data fall in the recently established research area of *(multi-)relational data mining* (MRDM) [DL01]. The data model that can suitably represent the units of analysis studied in MRDM is that of relational database. However, to be able to analyse relational databases containing multiple relations properly, specific algorithms have to be written that cope with the structural information that occurs in relational databases [KBSV99].

In this thesis we investigate two cases of structured data presented above in the context of Naive Bayes classification. The first case concerns the presence of a taxonomical relation on the categories of the units of analysis (categorization structure) and the second case concerns the presence of relationships between objects composing the units of analysis (unit structure).

1.2 Contribution

The main contribution of this thesis concerns the extension of statistical classifiers and, in particular, Bayesian classifiers, to deal with two particular cases of structured data, namely categorization structure and unit structure. The former will be investigated in the context of propositional learning, while to represent the unit structure, we will resort to the multi-relational data mining setting.

1.2.1 Classification in a hierarchy of categories

Classification in a hierarchy of categories is receiving growing attention in the literature especially in some specific application domains, such as text classification, functional genomics, and in general, in applications where it is possible to define a is-a relation between categories. Indeed, for what concerns text classification, many popular search engines and text databases arrange examples (documents) in topic hierarchies, such as Yahoo, Google Directory, Medical Subject Headings (MeSH) in MEDLINE, Open Directory Project (ODP) (www.dmoz.org) and Reuters Corpus Volume I (RCV1). In functional genomics, the problem of predicting the functional class of a gene cabn be considered as a problem of hierarchical classification since genes are organized hierarchically. For example in the Munich Information Center for Protein Sequences (MIPS) hierarchy [1], the top level of the hierarchy has classes such as: "Metabolism", "Energy", "Transcription" and "Protein Synthesis". Each of these classes is iteratively subdivided into more specific classes, so to obtain a hierarchy which is up to 4 levels deep. An example of a subclass of "Metabolism" is "amino-acid metabolism", and an example of a subclass of this is "amino-acid biosynthesis". An example of a gene in this subclass is YPR145w (gene name ASN1, product "asparagine synthetase") [BBD+02] [CK01]. In such applications pre-defined categories are organized in a hierarchical structure (tree-like structure). Such a structure reflects relations between concepts in the application domain covered by the classification.

This hierarchical arrangement is essential when the number of categories is quite high and the use of a non-hierarchical classifier (flat classifier) would lead to a fragmentation of the class, producing many classes with few members. On the other hand, the hierarchical classification arranges examples hierarchically, thus supporting a thematic search by browsing topics of interests. The structural relationship among categories can be taken into account when devising the classification process. While in flat classification a given example is assigned to a category on the basis of the output of one classifier, in hierarchical classification, the assignment of a document to a category can be done on the basis of the output of multiple sets of classifiers, which are associated to different levels of the hierarchy and distribute examples among categories in a top-down way. The advantage of this hierarchical view of the classification process is that the problem is partitioned into smaller subproblems, each of which can be effectively and efficiently managed. Another motivation is given by the observation that at different levels of the hierarchy the

[1] http://mips.gsf.de/proj/yeast/catalogues/funcat/

same example can be represented in a different way. In particular, it is possible to use different abstractions of the same object varying the level of the hierarchy (e.g. it is possible to emphasize some features rather than others at different levels of the hierarchy).

Although there are several approaches that face the problem of hierarchical classification of examples, they are often strictly related to the application in hand and lack of a general and domain free approach to the problem. In this thesis, we propose a general framework for hierarchical classification of examples. It supports the change of representation of examples at different levels of hierarchy. The framework includes a tree distance-based thresholding algorithm for the classification of examples in internal categories of the hierarchy. It can be applied to any classifier, such as naive Bayes, that returns a degree of membership (e.g. probabilistic or distance based) of an example to a category. The framework can manage a variety of situations in terms of hierarchical structure: examples can be assigned to any node in the hierarchy, some nodes can have no associated examples and internal nodes can have only one child.

1.2.2 Classification in Multi-Relational Data Mining

In hierarchical classification the predicted attribute is supposed to be structured. Anyway, it does not consider the eventuality that the training data represent relations between the target objects and the task relevant objects (unit structure). In order to deal with this second problem, it is necessary to resort to multi-relational data mining. Multi-Relational Data Mining is a new branch of data mining research that overcomes the problem of *single table assumption* that assumes that the training set can be represented as a single relational table, where each row corresponds to an example and each column to a predictor variable or to the target variable. This assumption is made in classical data mining and seems quite restrictive in some data mining applications, where data are stored in a database and are organized into several tables for reasons of efficient storage and access ore are stored in the form of complex objects. In this context, both predictor variables and the target variable are represented as attributes of distinct tables (relations) eventually related each other by means of foreign key constraints defining a structure in the data.

Several approaches for classification in multi-relational setting have been proposed in the literature, but often the problem is solved by moulding a relational database into a single table format, such that traditional attribute-value algorithms are able to work on [KHS01]. This approach is known it the literature as propositionalization. Two techniques have been proposed for propositionalization. The former is based on the principle that it is possible to consider a single relation reconstructed by performing a relational join operation on the tables. This technique is fraught with many difficulties in practice [DR98] [Get01b]. It produces an extremely large and impractical to handle table with lots of data being repeated. A different technique is the construction of a single central relation that summarizes and/or aggregates information which can be found in other tables. Also this approach has some drawbacks, since information about how data were originally structured is

lost [KRZ+03]. In order to overcome such limitations, structural Multi-relational data mining approaches are being used, especially for the classification task [Blo98] [FGKP99a] [KW01b] [PK95] [FL04].

In this thesis we present two different solutions to the problem of mining classifiers from relational data. In particular, we intend to extend the naive Bayes classification to the case of relational data. The first solution is based on the use of a set of first-order classification rules in the context of naive Bayesian classification. We present differences, benefits and drawbacks of the proposed approach with respect to similar approaches reported in the literature. The second solution is inspired by recent studies on the usage of association rules for classification purposes [DZWL99a] [BG03a]. This approach, named associative classification [LHM98], presents several advantages. First, differently from most of classifiers as decision trees, association rules consider the simultaneous correspondence of values of different attributes, hence allowing to achieve better accuracy [BG03a]. Second, associative classification makes association rule mining techniques applicable to classification tasks. Third, the user can decide to mine both association rules and a classification model in the same data mining process [LHM98]. Fourth, the associative classification approach helps to solve understandability problems [CM93a] [PMS97a] that may occur with some classification methods. We propose a multi-relational associative classifier that performs the classification at different granularity levels and takes advantage from domain specific knowledge in form of rules that support qualitative reasoning.

1.3 Outline of Thesis

This thesis is organized as follows: in the next chapter the problem of Naive Bayesian probabilistic classification in classical data mining is introduced. The problem of hierarchical classification is also described and investigated. In the third chapter the Naive Bayes classification in multi-relational Data Mining is described and several approaches are proposed. In the fourth chapter some applications of Naive Bayes classification in the field of document engineering are proposed. Other applications are reported in chapter five. Finally, in the sixth chapter conclusions are drawn and future works are proposed.

Chapter 2

Naive Bayesian Hierarchical Classification

This chapter formally defines the classification problem in data mining focusing the attention on the Naive Bayesian classification. We review some of the seminal works reported in the literature and we illustrate the classification problem when hierarchical relations between target categories are taken into account. We also propose a general framework for hierarchical classification of examples. It can be applied to any classifier that returns a degree of membership, such as probabilistic or distance-based classifier.

2.1 Data Mining and statistical classification

Knowledge discovery in databases (KDD) is defined as the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data and describing them in a concise and meaningful manner [FPSM92]. This process is interactive and iterative, involving numerous steps with many decisions being made by the user [FPSS96]. The overall KDD process involves the following steps:

1. Understanding the application domain: includes defining relevant prior knowledge and goals of the application.

2. Extracting the target data set: includes selecting a data set or focusing on a subset of variables.

3. Data cleaning and preprocessing: includes basic operations, such as noise removal and handling of missing data. Data from real-world sources are often erroneous, incomplete, and inconsistent, perhaps due to operation error or system implementation flaws. Such low quality data needs to be cleaned prior to data mining.

4. Data integration: includes integrating multiple, heterogeneous data sources.

5. Data reduction and projection: includes finding useful features to represent the data (depending on the goal of the task) and using dimensionality reduction or transformation methods.

6. Choosing the task of data mining: includes deciding the purpose of the model derived by the data mining algorithm (e.g., summarization, classification, regression, clustering, discovering association rules and functional dependencies, rule extraction, or a combination of these).

7. Choosing the data mining algorithm(s): includes selecting method(s) to be used for searching patterns in data, such as deciding on which model and parameters may be appropriate.

8. Data mining: includes searching for patterns of interest in a particular representational form or a set of such representations.

9. Interpretation: includes interpreting the discovered patterns, as well as the possible visualization of the extracted patterns. One can analyze the patterns automatically or semiautomatically to identify the truly interesting/useful patterns for the user.

10. Using discovered knowledge: includes incorporating this knowledge into the performance system, taking actions based on knowledge.

Information flows forwards from one stage to the next, as well as backwards to previous stages.

KDD (or Data Mining) on its extensive exception brings together techniques from machine learning, pattern recognition, statistics, databases, linguistics and visualization in order to extract information from large databases. It is an interdisciplinary field with a general goal of predicting outcomes and uncovering relationships in data. It uses automated tools employing sophisticated algorithms to discover hidden patterns, associations, anomalies and/or structure from large amounts of data stored in data warehouses or other information repositories. Data mining tasks can be descriptive, i.e., discovering interesting patterns describing the data, and predictive, i.e., predicting the behavior of the model based on available data.

In many predictive data mining tasks, we can assume that data are generated independently and with an identical and unknown distribution P on some domain X and are associated with a value in some domain Y according to an unknown function g. The domain of g is spanned by m independent (or predictor or explanatory) random variables X_i (both numerical and categorical), that is $X = X_1 \times X_2 \times \cdots \times X_m$ and the goal is to predict the dependent (or response or target) variable Y. An inductive data mining algorithm takes a training sample $S = \{(x, y) \in X \times Y | y = g(x)\}$ as input and returns a function f which is hopefully close to g on the domain X. In this scenario, when the target variable Y is a symbolic attribute ($Y = C_1, C_2, \ldots, C_L$), the inference task is called *classification*, when Y is a continuous value, the inference task is called *regression*. In this thesis we focus our attention in classification. In particular, we tackle the problem of automatic

classification of data characterized by a non-deterministic nature for which it is not possible to create a complete and consistent model of the world to be modelled.

In such cases, a typical choice is to resort to probabilistic classification that permits to act taking the uncertainty into account. Statistical classification finds its roots in Statistical Decision Theory [Ber93] and can be formally stated as a problem of expected risk minimization. Let D be the set of functions such that $D = \{f|f : X \rightarrow Y\}$ and R_S be a risk function defined in D such that:

$$R_S(f) = E\{L(y, f(x))|(x, y) \in S\} \tag{2.1}$$

where $L(y, f(x))$ is the loss incurred when the true y is estimated by $\hat{y} = f(x)$. The goal is to find a function f_{opt} such that

$$f_{opt} = argmin_f R_S(f) \tag{2.2}$$

Under the general setting of Statistical Decision Theory, several methods and techniques have been studied in the literature. Such methods are able to find the f_{opt} function according to different loss functions and different function estimations. They include: Linear Regression, Linear Discriminant Analysis, Logistic Regression, Decision Trees, K-Nearest Neighbor, Support Vector Machines and Bayesian Networks. In the following subsections we briefly describe such methods.

2.1.1 Linear Regression

Linear models are classifiers that partition the sample space X and associate each partition with a class value. Generally, in linear models, the domain X is spanned in m numerical random variables and the term "linear" pertains the fact that *"decision boundaries"* are represented in form of planes. Classical linear models for classification are: Linear Regression, Linear Discriminant Analysis and Logistic Regression.

In Linear Regression, the prediction model is computed according to the linear regression function:

$$\hat{\mathbf{Y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

Where $\mathbf{X}$ represents the input matrix $N \times (m+1)$ (N is the number of training examples) with a row for each training example and composed by (m+1) columns corresponding to m inputs and a leading column of 1's for the intercept. $\mathbf{Y}$ is the indicator $N \times L$ response matrix of 0's and 1's, with each row having a single 1.

Let $\hat{\mathbf{B}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$ be the $m + 1 \times L$ coefficient matrix and x be a new instance to be classified, the classifier computes the fitted output L vector

$$f(x) = [(1, x)\hat{\mathbf{B}}]^T$$

and returns the best class:

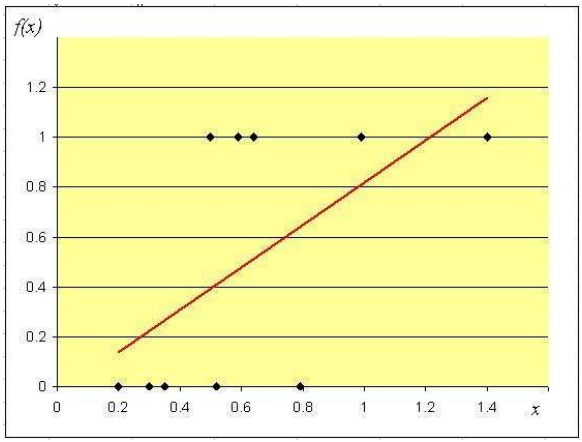

Figure 2.1: Simple case of a 2-class linear Regression application in 2-dimensional sample space

$$\hat{y} = argmax_{i=1..L}\ f_i(x) \tag{2.3}$$

Linear regression provides a simple approach to use statistical regression for classification problems (see Figure 2.1). However, the approach has a severe limitation when the number of classes is $L \geq 3$, especially prevalent when L is large. Because of the rigid nature of the regression model, classes can be masked by others [HTF01] and, in extreme situations, even if classes are perfectly separated by linear decision boundaries, yet linear regression completely misses some classes. To overcome this limitation, Frank et al. [FWI+98] proposed an approach that finds a different regression model for each partition of the sample space following an approach similar to classification trees, namely model trees [WW97][MECA04].

2.1.2 Linear Discriminant Analysis

In Linear Discriminant Analysis, the main idea is to estimate the class posterior distribution for optimal classification. The class posterior distribution is a discrete probability distribution that a given example is classified in a class $\hat{P}(Y|X)$.

A simple application of the Bayes Theorem permits to estimate $P(Y|X)$:

$$P(Y = y|X = x) = \frac{P(X = x|Y = y)P(Y = y)}{\sum_{l=1..L} P(X = x|Y = C_l)P(Y = C_l)}$$

where $P(Y)$ is the prior probability and $P(X = x|Y = y)$ is the class density or likelihood.

Many techniques are based on the application of this formula, they mainly differ in the estimation of the likelihood. In Discriminant Analysis it is generally assumed that each class density is a multivariate Gaussian [DH73], that is:

$$f_l(x) = P(Y = C_l|X = x) = \frac{1}{(2\pi)^{p/2}|\Sigma_{C_l}|^{1/2}} e^{-1/2(x-\mu_{C_l})^T \Sigma_{C_l}^{-1}(x-\mu_{Y_l})} \tag{2.4}$$

where the Σ_{C_l} is the covariance matrix associated to the $l-th$ class value and μ_{C_l} is the vector of the attribute averages associated to the $l-th$ class value.

Linear Discriminant Analysis arises in the special case when we assume that the classes have the same covariance matrix, that is, $\Sigma_{C_l} = \Sigma$ for each $l = 1..L$. In this case the discriminant functions that partition the sample space in C_l and its complement $\neg C_l$ is (from 2.4, by passing to logarithms):

$$\delta_{C_l}(x) = x^T\Sigma^{-1}\mu_{C_l} - 1/2\mu_{C_l}^T\Sigma^{-1}\mu_{C_l} + log(\pi_{C_l}) \tag{2.5}$$

where π_{C_l} is the priori probability of class l.

The class is estimated by:

$$\hat{y} = argmax_{i=1..L}\ \delta_{C_l}(x) \tag{2.6}$$

Linear Discriminant Analysis generally provides an accurate model when the decision boundaries in the data can be described by linear models. Indeed, in such cases, the gaussian models are stable. [MSTC94].

2.1.3 Logistic Regression

Another linear method for classification is Logistic Regression. As in the case of Linear Discriminant Analysis, in logistic regression the goal is to estimate the posterior probability distribution $P(Y|X)$. [HL00]. The estimation is performed by means of linear functions in x. The model has the form:

$$\begin{aligned}
&log\frac{P(Y=C_l|X=x)}{P(Y=C_L|X=x)} = \beta_{10} + \beta_1^T x\\
&log\frac{P(Y=C_2|X=x)}{P(Y=C_L|X=x)} = \beta_{20} + \beta_2^T x\\
&...\\
&log\frac{P(Y=C_{L-1}|X=x)}{P(Y=C_L|X=x)} = \beta_{(L-1)0} + \beta_{L-1}^T x
\end{aligned}$$

The computation of the probabilities depends on the parameter

$$\beta = \{\beta_{10}, \beta_{20}, ..., \beta_{(L-1)0}, \beta_{L-1}\}$$

that is typically estimated by maximizing the following measure:

$$-\sum_{i=1}^{N}\sum_{l=1}^{L} sgn(C_l, y_i)log(P(Y = C_l, |X_i, \beta)) \tag{2.7}$$

where $sgn(C_l, y_i) = 1$ if $C_l = y_i$, 0 otherwise.

The maximization of this equation in typically obtained by finding the value of β that equals to zero the first derivative in β and leaves positive the determinant of the Hessian matrix.

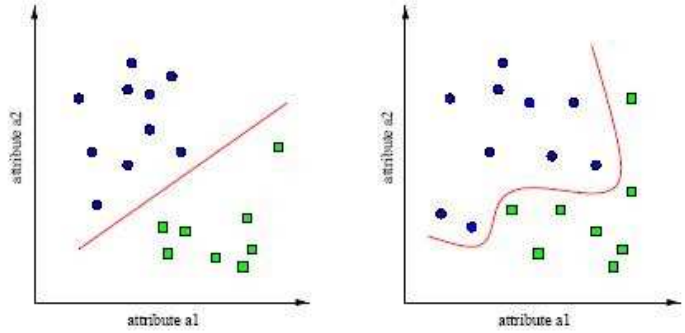

Figure 2.2: Linear separation of the sample space(a). Non linear separation of the sample space(b)

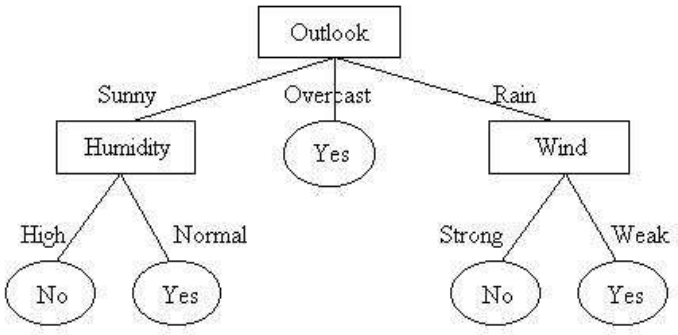

Figure 2.3: A decision tree for the didactic example *Play Tennis*

Logistic Regression models are used mostly as data analysis and inference tool, where the goal is to understand the role of an input variable in explaining the outcome [HTF01].

Logistic Regression, as most linear models provide an approach to define linear boundaries that partition the sample space. However, in most cases, the nature of the underlying problem is not linear and, in such case, the predicted model cannot be adequate for the problem in hand and different techniques have to be adopted (see Figure 2.2).

2.1.4 Decision Trees

Decision trees return a decision tree. A decision tree classifier represents a disjunction of conjunctions of constraints on the attribute values of examples. A decision tree can be represented by a tree structure characterized by two types of nodes: internal nodes, that represent a partition of the sample space according to the value of an attribute, and leaves, that represent the taken decision. In Figure 2.3, an example of decision tree for the didactic example *Play Tennis* [Qui86] is reported.

A new example is classified by starting at the root node of the tree, testing the attribute specified by this node and then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the subtree rooted at the new node. The entire process is repeated until a leaf

node is reached.

Several algorithms have been proposed to learn decision tree, and most of them are variations of the core algorithm ID3 [Qui86] and its successor C4.5 [Qui93]. The algorithm employs a top-down greedy search through the space of the possible decision trees. In particular, ID3 construct decision trees in a top-down fashion. The main step concerns the the selection of the attribute to be tested. For this purpose, a statistical test is generally used and the goal is to find the attribute that is most useful for classifying examples.

Formally, in the evaluation of a single node, the algorithm chooses the attribute that minimizes the information lost on the basis of an impurity function. An impurity function is a function

$$\Phi : [0,1]^m \rightarrow \Re^+ \tag{2.8}$$

such that, given a probability vector $PC =< p_1, p_2, ..., p_L >$ where $p_i = P(C_i)$, $\Phi(PC)$

- is minimal if all training examples belong to a single class ($\exists i = 1..m \;\; s.t. \;\; p_i = 1$)
- is maximal if examples are equally distributed over the classes ($\forall i = 1..m$, $p_i = 1/m$)
- is symmetric with respect to the components of PC
- is differentiable over $[0,1]^m$

If an internal node t associated with a probability vector $PC(t)$ represents a test on the j^{th} attribute and produces k different nodes $t_1, t_2, ..., t_k$, the best test maximizes the impurity lost:

$$Gain(t,j) = \Phi(PC(t)) \;\; - \;\; \sum_{i=1}^{k} \frac{N(t_i)}{N(t)} \Phi(PC(t_i))$$

where $N(t)$ represents the number of examples that fall in the node t
The impurity measure implemented in ID3 is the entropy:

$$E(PC) = -\sum_{j=1}^{L} p_i \lg_2 p_i$$

Another typically used measure is:

$$GainRatio(t,j) = Gain(t,j)/IV(t,j)$$

where

$$IV(t,j) = -\sum_{i=1}^{k} \;\; \frac{|N(t_i)|}{N(t)} \lg_2 \;\; \frac{|N(t_i)|}{N(t)}$$

Differently from the information gain, the gain ratio does not favour splits with multiple attributes.

Several variants have been investigated in the literature and from them we cite the Gini Index proposed by Breiman et al. [BFOS84]. The Gini index is another impurity measure defined as follows:

$$GIndex(PC) = 1 - \sum_{j=1}^{L} p_i^2$$

Differently from previously described methods, in the original formulation of decision tree learners the assumption is that the independent random variables $X_1, X_2, ...X_m$ are discrete. In order to deal with continuous attributes, a discretization by considering as "split points" the middle point between two consecutive values is typically used [BFOS84].

Decision trees provides a simple and efficient way to learn classifiers. The main advance in the use of decision trees is in the understandability if the induced model. In fact, a decision tree can always be transformed in a set of rules that are easily understandable to a human. The number of extracted rules equals the number of leaves in the tree.

2.1.5 K-Nearest Neighbors

K-Nearest Neighbor (K-NN) is a particular classifier that achieves flexibility in estimating the class of a given example over the whole sample space, by fitting a different but simple model separately at each query point x'. This is done by using only those observations close to the target point x'. More precisely, given a new example x' (query point), the method uses a subset of the training set, $N_k(x')$, to compute the class y' associated to x'. $N_k(x')$ is the neighborhood of x' defined by the k closest points in the training set. Formally:

$$f(x') = argmax_{y' \in Y} \sum_{x \in N_k(x')} \delta(y', y)$$

where y is the class associated to x

$$\delta(y', y) = \begin{cases} 1 & \text{if} \quad y' = y \\ 0 & \text{if} \quad y' \neq y \end{cases}$$

When the value of y' is not unique (there is no dominant class) then $f(x')$ can be considered “unknown”.

Figure 2.4 reports an example of classification of a new instance (red point). The closest example is a "plus", this means that a 1-NN classifier returns "plus". A 2-NN classifier returns "unknown" because in the neighborhood there are 2 examples and there is no dominant class.

A variant of the standard K-NN algorithm gives more importance to the instances that are closest to the target point (locally weighted K-NN). This approach

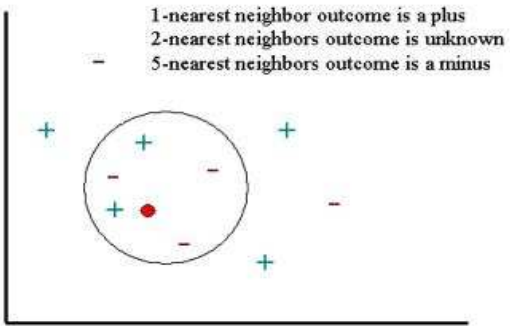

FIGURE 2.4: Knn Example

assigns a greater weight to instances that are close to the target point and reduces the weight for more distant instances:

$$f(x') = argmax_{y' \in Y} \sum_{x \in N_k(x')} \frac{1}{d(x', x)^2} \delta(y', y)$$

If $d(x', x) = 0$ for some x, then $f(x')$ assumes the value $y = f(x)$.

In the example reported in Figure 2.4, differently from the classical 2-NN classifier, the weighted 2-NN classifier classifies the new instance as "plus" instead of "unknown".

In the K-NN method, great importance is given to the distance measure. The measure is used both in the definition of the neighborhood $N_k(x')$ and in the definition of the weights.

Typically used measures are [Mit97]:

Euclidean distance. This distance is defined for real values and is based on the following equation:

$$d(x', x'') := \sqrt{\sum_{i=1}^{m} (x'_i - x''_i)^2}$$

Minkowski distance. This distance is defined for real values and, given a parameter q, it is computed by means of the following equation:

$$d(x', x'') := (\sum_{i=1}^{m} (x'_i - x''_i)^q)^{1/q}$$

Hamming distance. This distance is defined in $\{0, 1\}^m$ and is computed by counting the number of differences in the values of the input vectors.

Stanfill-Waltz distance. [SW86] Stanfill and Waltz introduced the Value Difference Metric (VDM) to define the similarity for discrete attributes. The VDM computes a distance for each pair of the different values a symbolic feature can assume. It essentially compares the relative frequencies of each pair of symbolic values across all classes. Two attribute values have a small distance if their relative frequencies are approximately equal for all output classes.

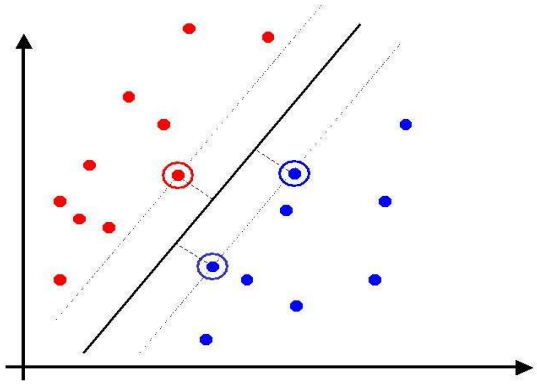

Figure 2.5: Support vectors used to define a separating hyperplanes

K-NN is an instance-based learner because, in contrast to learning methods that construct a general, explicit description of the target function, K-NN simply stores the training examples and uses them in the classification of a new instance. This property makes the K-NN method a high effective method in case of large datasets. Another important aspect is that K-NN is particularly robust to noisy training data.

A practical limitation of the K-NN method is that the distance measure is computed considering all attributes and not a subset of them. In the case that some attributes are not statistically significant in the prediction of the class, K-NN is not able to isolate them. The typically adopted solution uses weights to give more or less importance to "significant" or "non-significant" attributes respectively.

2.1.6 Support Vector Machines

Recently, a new learning technique has emerged and become quite popular in practical domains because of its good performance and its theoretical foundations in the statistical learning theory: support vector machines (SVMs), proposed by Vapnik [Vap95].

Given a set of positive and negative examples (SVMs are defined for two-classes problems), an SVM identifies the hyperplane in R^m that linearly separates positive and negative examples with the maximum margin (*optimal separating hyperplane*). In general, the hyperplane can be constructed as the linear combination of all training examples, however, only some examples, called support vectors, do actually contribute to the optimal separating hyperplane (see Figure 2.5), which can be represented as:

$$f(x) = (\sum_{x^* \in SUPP} (y^* \alpha_i x^*))^T x + b \tag{2.9}$$

where $SUPP \subseteq S$ represents the set of the support vectors and y^* is the class associated to x^*. The coefficients α and b are determined by solving a large-scale quadratic programming problem for which efficient algorithms exist, which are guaranteed to find the global optimum.

SVMs are based on the Structural Risk Minimization principle: a function that can classify training data accurately and which belongs to a set of functions with the lowest capacity (particularly in the VC-dimension) [Vap95] will generalize best, regardless of the dimensionality of the feature space m. Therefore, SVMs can generalize well even in large feature space, such as those used in text categorization. In the case of the separating hyperplane, minimizing the VC-dimension corresponds to maximizing the margin.

The linear separability appears to be a strong limitation. However, SVMs can be generalized to non-linearly separable training data by mapping the data into another *feature space* F via a non-linear map:

$$\Phi : \Re^m \rightarrow F \tag{2.10}$$

and then performing the above linear algorithm in F. Generally the map introduce new features that do take into account the correlation between the input features. The solution has the form:

$$f(x) = (\sum_{x^* \in SUPP} (y^* \alpha_i \Phi(x^*)))^T \Phi(x) + b \tag{2.11}$$

that is non linear in the original feature set.

However, training a Support Vector Machine requires the solution of a very large quadratic programming (QP) problem and optimized algorithms have been proposed in the literature. They include: SVM^{light}, proposed by Joachims [Joa, Joa99a] and Sequential Minimal Optimization (SMO) proposed by Platt [Pla98]. The latter is very fast and is based on the idea of breaking the large quadratic programming (QP) problem down into a series of smaller QP problems that can be solved analytically.

2.1.7 Bayesian Networks

Bayesian Networks [Pea88] are compact graphical representations for high- dimensional joint distributions. They exploit the underlying conditional independences in the domain and the fact that only a few aspects of the domain affect each other directly.

Formally, a Bayesian Network B is composed of a structure and parameters. The structure is a directed acyclic graph that encodes a set of conditional independence relationships among variables. The nodes of the graph correspond directly to the attributes and the directed arcs represent dependence of attributes to their parents. The lack of directed arcs among attributes represent a conditional independence relationship.

In Figure 2.6, an example of Bayesian Network is reported. The lack of arcs between attributes $Attr_8$ and $Attr_{11}$indicate that they are conditionally independent given $Attr_1$.

The parameters of the network are the local probability distributions attached to each variable. The structure and parameters taken together encode the joint

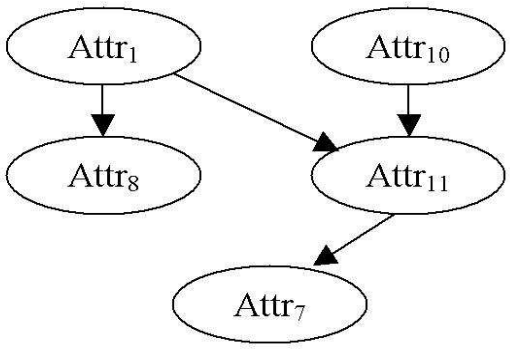

Figure 2.6: A Bayesian Network representing the conditional probabilities of attributes given the independent events described by $Attr_1$ and $Attr_{10}$

probability of the variables. In particular, given an attribute X_i, the structure implicitly defines $Pa(X_i)$, the set of attributes associated to the parent nodes of the node represented by X_i.

For each attribute X_i, it is possible to define a conditional probability distribution $P_B(X_i|Pa(X_i))$. A Bayesian Network B defines a unique joint probability distribution over the sample space, defined by the following equation.

$$P_B(X_1, X_2, ..., X_m) = \prod_{i=1}^{m} P_B(X_i|Pa(X_i)) \tag{2.12}$$

The problem of learning a Bayesian network from data can be broken into two components: learning the structure and learning the parameters. If the structure is known, then the problem reduces to learning the parameters. If the structure is unknown, the learner must first find the structure before learning the parameters (actually in many cases they are induced simultaneously). Learning the structure can itself be decomposed into searching for structures and evaluating structures.

In any case, the learned Bayesian Network can be profitably used for classification purposes, as proposed by Friedman and his colleagues [FGG97], who introduced tree-augmented naive Bayesian (TAN) Networks in which the class variable has no parents and each observed attribute has as parents the class variable and at most one other observed attribute. Thus, each attribute can have one augmenting edge pointing to it.

In Figure 2.7, an example of TAN model for the dataset "pima" taken from [FGG97] is reported. The attribute C is the target variable (class).

2.2 Naive Bayesian Classification

A different solution is provided by the Naive Bayesian classifier. The Naive Bayesian classifier[1] is a simple and computationally efficient learning algorithm with theoret-

[1]The term *Naive Bayesian Classifier* has been introduced by Kononenko [Kon90]. It refers to the *Simple Bayesian Classifier* [Lan93] and *idiot Bayes* [Bun90]

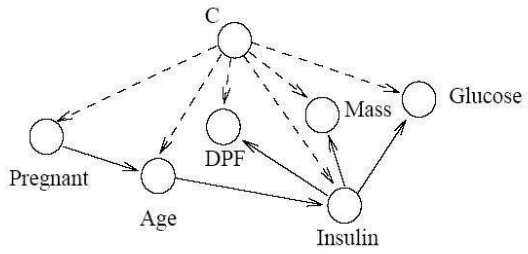

FIGURE 2.7: An example of TAN model for the dataset "pima"

ical roots in the Bayes theorem.

The Bayes theorem states that:

- Let $A_1, A_2, ..., A_L$ be mutually exclusive (disjoint) events whose union has probability one. That is, $\sum_{i=1}^{L} P(A_i) = 1$.
- Let the probabilities $P(A_i)$ be known.
- Let B be an event for which the conditional probability of B given A_i, $P(B|A_i)$ is known for each A_i.

Then:

$$P(A_i|B) = \frac{P(B|A_i)P(A_i)}{\sum_{j=1}^{L} P(B|A_i)P(A_i)} \tag{2.13}$$

The probabilities $P(A_i|B)$ reflect our updated or revised beliefs about A_i, in the light of the knowledge that B has occurred.

Making a notational change by identifying the events $A_1, A_2, ..., A_L$ with classes of a classification problem $C_1, C_2, ..., C_L$ and by identifying the event B as the union of events $(X_1 = x_1, X_2 = x_2, ...X_m = x_m)$, where, as specified in section 2.1, X_i are random predictor variables and $x_i (i = 1..m)$ are values, the Bayes formula can be written:

$$P(Y = C_i|X_1 = x_1, X_2 = x_2, ...X_m = x_m) = \frac{P(X_1 = x_1, X_2 = x_2, ...X_m = x_m|Y = C_i)P(Y = C_i)}{\sum_{j=1}^{L} P(X_1 = x_1, X_2 = x_2, ...X_m = x_m|Y = C_j)P(Y = C_j)} \tag{2.14}$$

$P(Y = C_i|X_1 = x_1, X_2 = x_2, ...X_m = x_m)$ is referred to as a posterior probability, $P(Y = C_i)$ as the prior probability, and $P(Y = C_i|X_1 = x_1, X_2 = x_2, ...X_m = x_m)$ as the likelihood. The result of the Bayes Theorem 2.14 expresses the fundamental task of *learning by experience* in terms of the relation of prior and posterior probability.

Once the probabilities have been estimated, the class is predicted by identifying the most probable one.

$$f(x) = argmax_i \; P(Y = C_i|X_1 = x_1, X_2 = x_2, ...X_m = x_m) \tag{2.15}$$
$$i = 1..L$$

In the estimation of the posterior probability, the denominator of formula 2.14 can be ignored, since it is the same for all classes. Thus, in Bayesian Learning, the problem is, given an unlabeled example $x = x_1, x_2..., x_n$, to individuate the most probable class to which x belongs.

$$f(x) = argmax_i \; P(X_1 = x_1, X_2 = x_2, ...X_m = x_m|Y = C_i)P(Y = C_i) \quad (2.16)$$
$$i = 1..L$$

Bayes theorem plays an important role in inducting inference since 1961, when Harold Jeffreys devised five essential rules of inductive inference under which the theorem could be subsumed as representing one important case of probabilistic inference [Jef61]. Afterward, the Bayes theorem has been used in several statistical approaches for learning [Got80](e.g. Bayesian Networks 2.1.7). Each approach tries to estimate the value of the likelihood in a different way.

One of the most common approaches is the Naive Bayesian Learner, where the estimation of the likelihood is performed by means of the simplistic (naive) assumption that an attributes is independent of each other, given the class. Formally:

$$P(X_1 = x_1, X_2 = x_2, ...X_m = x_m|Y = C_i) = \prod_{j=1}^{m} P(X_1 = x_1|Y = C_i) \quad (2.17)$$

Thus, the discriminant function is:

$$f(x) = argmax_i \; P(Y = C_i) \prod_{j=1}^{m} P(X_1 = x_1|Y = C_i) \quad (2.18)$$

The naive assumption is clearly false when the predictor variables are statistically dependent. Furthermore, in practical domains, the attributes are seldom independent given the class. For this reason, the Naive Bayesian Learner has been considered not much reliable and has been initially used as baseline for comparison with more sophisticated algorithms. Despite of this scepticism, in past years it has been shown that in many domains the prediction accuracy of the naive Bayesian classifier compares well with that of other more complex learning algorithms including decision tree learning, rule learning, and instance-based learning algorithms.

In particular, Cestnik et al. [CKB87], Clark and Nibblett [CN89] and Cestink [Ces90] compared the naive Bayesian classifier with rule learners and results empirically proved the effectiveness of the naive Bayesian classifier. Langley et al. [LIT92] compared the Naive Bayesian classifier with a decision tree learner and found that it is more accurate in most of cases. Finally, Domingos and Pazzani [DP96] compared several learners in information filtering tasks and found that the Naive Bayesian classifier is the most accurate.

In addition, the naive Bayesian classifier has been proved robust to noise and irrelevant attributes [ZW00] and Kononenko [Kon00] reported that domain experts (in medicine), found learned theories easy to understand.

In past years, several attempts to improve the Bayesian Classifier have been performed. Such improvements concerned three different and non-orthogonal directions, namely, improvements in the treatment of numeric attributes, improvements

in the treatment of statistically dependent attributes (with respect to the class) and, finally, improvements in the classifier's expressiveness by means of a more sophisticate representation.

2.2.1 Numeric attributes

The Naive Bayes classifier has been defined for discrete attributes. To extend the standard Naive Bayes classifier to both continuous and discrete attributes, several approaches have been proposed. The simplest approach consists in a preprocessing step that aims to discretize continuous attributes. A different solution consists in an embedded discretization method that discretize continuous attributes during the learning phase.

One of the approaches that follows the latter solution has been proposed by John and Langley [JL95], where two discretization algorithms are compared: a kernel density estimation of numeric attributes and an estimation based on a Gaussian distribution. The authors empirically proved that the kernel density estimation algorithm is, in general, the better solution.

An independent study by Dougherty et al. [DKS95] showed that the performance of the naive Bayes algorithm is significantly improved when features are discretized using an entropy-based method. On the contrary, when purity-based methods are applied, the performance of the naive Bayes algorithm is not significantly improved.

2.2.2 Relaxing the independence assumption

The assumption of conditional independence made by naive Bayes learners has often been regarded as unrealistic. Therefore, several attempts to relax this assumption can be found in the literature. Most of them have been proposed chronologically before the empirical and formal considerations on the optimality region of the naive Bayes classifier illustrated by Domingos and Pazzani [DP97] in 1997.

In 1991, Kononenko proposed an approach (namely, Semi-Naive Bayesian Classifier) [Kon91] that is based on the joining of dependent attribute values. The dependence/independence is checked accordingly to a statistical test. Experiments conducted to evaluate the effectiveness of this approach did not show significant improvements in accuracy.

Langley and Sage [LS94] responded to the problem by embedding the naive Bayesian scheme within an algorithm that carries out a greedy search through the space of features (so, resorting to a feature selection problem). They found that it is, in general, beneficial in domains that involve significant correlation among attributes. However, in most domains no advantage of the proposed approach is noticed.

Following the idea proposed by Kononenko, Pazzani [Paz96] proposed a method to join attributes, rather than values. The algorithm is based on an iterative procedure that joins two attributes at each step. The evaluation of the effectiveness of the join is based on a cross-validation approach. The method shows improvements with respect to the standard naive Bayesian classifier. However, the author

noticed that the main advantage concerned the case when the Bayesian classifier was substantially less accurate than decision trees learners.

In 2002, Webb et al. [WBW02] [WBWss] proposed a method named AODE, that overcomes the attribute independence assumption of naive Bayes by averaging over all models in which all attributes depend upon the class and a single other attribute. The resulting classification learning algorithm for nominal data is computationally efficient and achieves lower error rates. However, the learned theories are not easily understandable.

2.2.3 Optimality of Naive Bayes Classifier

Despite of this multitude of approaches, Domingos and Pazzani [DP97] showed that relaxing the independence assumption is not always the best way to improve naive Bayesian classifier. Indeed, the paper presents both formal and empirical results showing that even when the independance assumption is violated by a wide margin, the naive Bayesian classifier returns the same classification of the Bayesian classifier that takes into account the statistical dependence among attributes (non-naive).

The basic idea is that, even if the classifier does not return correct probabilities, the discriminant function 2.18 still returns correct estimations.

The authors proved a set of both sufficient and necessary conditions for the *global optimality* of the naive Bayes classifier. For global optimality, they intend that, for each example in the sample space, the zero-one loss (error rate) is the same of the Bayesian classifier that is not based on the independence assumption of attributes given the class. Among others, interesting results are that the naive Bayesian classifier is globally optimal for learning conjunctions and disjunctions of literals.

The authors also proved limitations of the naive Bayes classifier. In particular, when all attributes are nominal, the Bayesian classifier is not globally optimal for classes that are not discriminable by linear functions of linear features.

Another important aspect of the naive Bayesian classifier observed for two class problems, is that, even if the classifier does not return a correct estimation of the class, it tends to rank examples well [ZE01]. This is an important point because a simple modification of the naive Bayesian classifier will allow it to better discriminate all positive examples from negatives [DP97]. This modification is particularly useful in the case of learning $(m-of-n)$ *concepts* [MP91] as well as in the case of unbalanced data [ZE01].

2.2.4 Improving the classifier's expressiveness

A different research direction on naive Bayesian classifiers is improving the classifier's expressiveness. Langley [Lan93] proposes a method, named RBC (Recursive Bayesian Classifier), that recursively partitions the instance space in sub-regions by means of a hierarchical clustering algorithm. A Bayesian classifier is learned for each sub-region. Although the method paves the way in revising the naive Bayesian classifier, the results do not show significant improvements over it.

A different approach, proposed by Kohavi [Koh96] is NBTree that scales up the accuracy of Naive Bayesian classifier in large datasets as well as decision trees learners. This demand derives from the observation that, increasing the size of the training set, the Naive Bayesian classifier tends to degrade its performances in terms of classification accuracy [DP97][Koh96]. NBTree induces a hybrid of decision tree classifiers and Naive Bayesian classifiers. Each node of the decision tree classifier represents a split over the sample space, while the leaves represent a Naive Bayesian Classifier. An empirical analysis of NBTree shows that, in general, NBTree tends to outperform either approaches alone, namely the Naive Bayesian classifier and the decision trees classifier, in the case of large datasets. The performance of NBTree is comparable to that of the standard Naive Bayesian Classifier.

Sahami, in 1996 [Sah96], proposed a learning algorithm, named Kdb, very similar to Bayesian networks. Kdb learns Bayesian classifiers that allow each attribute to depend on at most k other attributes within a class for a given number k. When k is equal to 0, Kdb generates naive Bayesian classifiers, while when k is equal to the number of all attributes 1, Kdb creates full Bayesian classifiers without attribute independences.

Friedman et al., in 1997, [FGG97], compared naive Bayesian classifier with Bayesian Networks (see 2.1.7) that is, a much powerful representation that has the bayesian classifier as a special case and found that Bayesian Networks tend to produce no improvements and, sometimes, lead to large reduction in accuracy with respect to the naive Bayesian classifier. From this observation, they proposed a *intermediate* solution that allows each attribute to depend on at most one other attribute in addition to the class. This method has been named TAN and is a special case of Kdb (in terms of expressiveness). Experiments showed that TAN provides interesting results in terms of accuracy.

In 2000, Zheng and Webb proposed LBR (Lazy Bayesian Rule Learning algorithm) which differ from NBTree for the fact that avoids the problem of small disjunctions of tree learning algorithms. The small disjunct problem, called by Holte et al. [HAP89], is defined as the problem that occurs when a disjunct covers only a few training examples and, Holte et al. by examining previously learned concepts, showed that small disjuncts are much more error prone than large disjuncts. To overcome this problem, LBR adopts a lazy approach and generates a rule that is the most appropriate to the test example.

Among all methods presented above, both LBR and TAN show some real improvement with respect to the standard naive Bayesian classifier. However, both techniques obtain this result at a considerable computational cost. In order to face the computational complexity problem, Webb et al. [WBW02] [WBWss] proposed a different approach named AODE, which works by averaging over all models in which all attributes depend upon the class and a single other attribute. The resulting classification learning algorithm for nominal data is computationally efficient and achieves lower error rates. Experiments delivers comparable prediction accuracy to LBR and TAN with improved computational efficiency.

2.3 The roles of data: units of analysis and units of observation

In this thesis we try to further improve the expressiveness of the naive Bayesian classifier by facing the problem of naive Bayesian classification taking the "structure" of data into account.

The structure of data can be in various forms. In this work we consider two common interpretations of structured data: the occurrence of relations between categories of the units of analysis (categorization structure) or the occurrence of relations between the units of analysis and/or the observation units (unit structure).

Before formally describing the problems we face, some useful definitions are necessary. In particular it is important to formally define the concepts of *unit of analysis* and *unit of observation*.

A Unit of analysis is the observable entity that is *analyzed* in a statistical study and to which the generalizations made by a statistical analysis apply.

This definition contrasts with the definition of the Unit of observation that is the entity which is observed or about which information is systematically collected. The unit of observation is the same as the unit of analysis when the generalizations being made from a statistical analysis are attributed to the unit of observation (i.e., the objects about which data were collected and organized for statistical analysis).

The difference between units of analysis and units of observation is basically due to the difference between primary and secondary data. Primary data are originated by the researcher for the specific purpose of addressing the research problem. Typical methods for the collection of primary data are interviews (in social science) or experiments (in other scientific discipline).

Secondary data are collected for some purpose other than the problem at hand. They generally are census data (in social science) or data published on research reports (in other scientific disciplines). Secondary data are cheaper to collect and sometimes they are the only source of data where it is possible to conduct primary research on the topic of interest. The disadvantage related to secondary data are the validity (e.g collected questionnaires may not be ideally worded for the research question at hand) and the convergence of a population different from that targeted in a data mining task.

If units of observation always refer to primary research (and hence to primary data), the units of analysis strongly depend on the problem at hand. While the units of observation and analysis are often the same, the wealth of secondary data sources creates opportunities to conduct analysis with data from multiple units of observation. This is probably most recognizable in research fields, such as Bioinformatics and spatial Data mining where secondary data sources are considered of great relevance in the analysis of data.

Example 2.1 *For instance, suppose that, for a social survey, data about each person in a dwelling and information about the housing structure are collected. Therefore, this study collects data for two units of observation:*

- *persons*

- *housing structures*

From these data, different units of analysis may be constructed:

- *Household could be examined as a unit of analysis by combining data from people living in the same dwelling.*

- *Family could be treated as the unit of analysis by combining data from all members in a dwelling sharing a familial relationship.*

Example 2.1 shows how the unit of analysis can be constructed from units of observation consisting of some type of relationship constructed by time, space or social properties.

In traditional data mining relatively simple transformations are required to obtain units of analysis from the units of observation explicitly stored in the database. The unit of observation is often the same as the unit of analysis, in which case no transformation at all is required. Conversely, in this thesis we face the problem of taking the "structure" into account. Where the structure is defined in terms of relations between units of analysis and/or units of observation and the relations between the units of observation. In the following section we take into account the occurrence of relations between categories of the units of analysis (categorization structure) and we propose a framework that supports the induction of statistical models for hierarchical classification of entities.

2.4 Classification in a Hierarchy of categories

Most of research in predictive Data Mining has focused on classifying examples into a set of categories with no structural relationships among them (flat classification). However, in many application domains, instances are organized in a hierarchy of categories in order to support a thematic search by browsing topics of interests. This is the case of text classification or functional genomics.

Indeed, many popular search engines and text databases arrange instances (documents) in topic hierarchies, such as Yahoo, Google Directory, Medical Subject Headings (MeSH) in MEDLINE, Open Directory Project (ODP)[2] and Reuters Corpus Volume I (RCV1). In functional genomics, a typical application is to predict the functional class of a gene, where genes are organized hierarchically. For example in the Munich Information Center for Protein Sequences (MIPS) hierarchy [3], the top level of the hierarchy has classes such as: "Metabolism", "Energy", "Transcription" and "Protein Synthesis". Each of these classes is iteratively subdivided into more specific classes, so to obtain a hierarchy which is up to 4 levels deep. An example of a subclass of "Metabolism" is "amino-acid metabolism", and an example of a subclass of this is "amino-acid biosynthesis". An example of a gene in this subclass is YPR145w (gene name ASN1, product "asparagine synthetase") [BBD+02] [CK01].

[2] www.dmoz.org

[3] http://mips.gsf.de/proj/yeast/catalogues/funcat/

In all these applications pre-defined categories are organized in a hierarchical structure (tree-like structure). Such a structure reflects relations between concepts in the application domain covered by the classification.

This hierarchical arrangement is essential when the number of categories is quite high and the use of a non-hierarchical classifier (flat classifier) would lead to a fragmentation of the class, producing many classes with few members. Furthermore, the hierarchical classification arranges examples hierarchically, thus supporting a thematic search by browsing topics of interests. The structural relationship among categories can be taken into account when devising the classification process. While in flat classification a given example is assigned to a category on the basis of the output of one classifier, in hierarchical classification, the assignment of an example to a category can be done on the basis of the output of multiple sets of classifiers, which are associated to different levels of the hierarchy and distribute examples among categories in a top-down way. The advantage of this hierarchical view of the classification process is that the problem is partitioned into smaller subproblems, each of which can be effectively and efficiently managed. Another motivation is given by the observation that at different levels of the hierarchy the same example can be represented in a different way. In particular, it is possible to give different abstractions of the same object varying the level of the hierarchy (e.g. it is possible to emphasize some attributes rather than others at different levels of the hierarchy).

However, taking into account the hierarchy poses additional issues in the development of methods supporting hierarchical classification. First, instances can either be associated to the leaves of the hierarchy or to internal nodes. Second, the set of attributes selected to build a classifier can either be category specific or the same for all categories (corpus-based). Third, the training set associated to each category may include or not training examples of subcategories. Fourth, the classifier may take into account or not the hierarchical relation between categories. Fifth, some stopping criterion is required for hierarchical classification of new instances in non-leaf categories. Sixth, new performance evaluation criteria are required to take into account the different types of classification errors.

Although there are several approaches that face the problem of hierarchical classification [BBD+02] [KS97] [MRMN98] [Mla98b] [DMSK00] [DC00] [NGL97] [RS02] [WWP99], they are often strictly related to the application in hand and lack of a general approach that is independent of the domain.

Here, we propose a general framework for hierarchical classification. It supports the change of representation of examples at different levels of hierarchy. The framework includes a tree distance-based thresholding algorithm for classifying instances in internal categories of the hierarchy. Although it is applied to Naive Bayes classifiers, it can be also applied to any classifier that returns a degree of membership (e.g. probabilistic and distance based) of an example to a category. The framework can manage a variety of situations in terms of hierarchical structure: examples can be assigned to any node in the hierarchy, some nodes can have no associated examples and internal nodes can have only one child. In the next subsections the proposed framework is described and a complexity analysis that formally proves the its efficiency is provided. Afterward, the proposed approach is compared with related

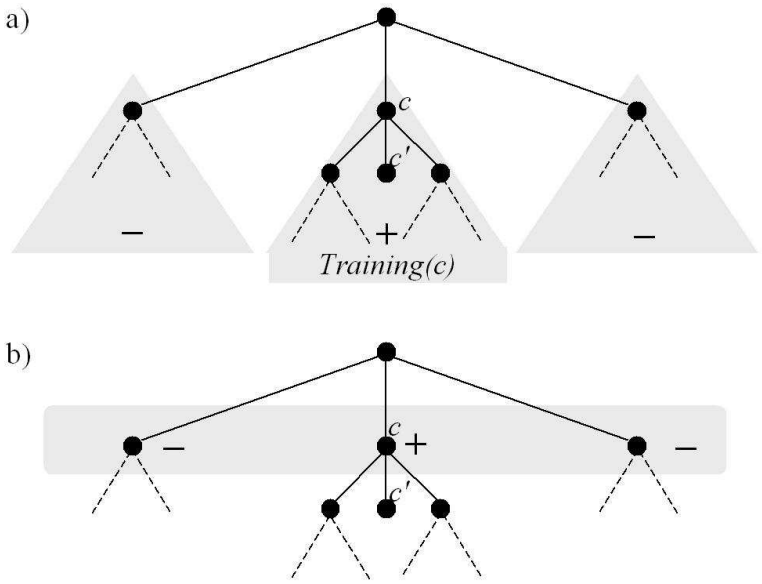

FIGURE 2.8: a) Hierarchical training set; b) proper training set.

works found in the literature. The empirical evaluation is reported in chapter 4 in the context of automatic classification of documents on the basis of their textual content.

2.4.1 Hierarchical classification: the framework

In our proposal, a classifier is learned for each each internal category c of the hierarchy. This classifier is used to decide, during the classification process, which category c_i, subcategory of c, is the most appropriate to receive the instance to be classified. Thus, in the learning process several classifiers have to be learned from a set of examples. An important aspect in the learning phase concerns the training set. Indeed, for each internal category, different opportunities to build a training set are possible. In [CM03], we identified two different approaches. In the first approach, called *Hierarchical Training Set*, the training set includes examples belonging to the subtree rooted in a category (positive examples) and examples of the sibling subtrees (negative examples). The second alternative is called *Proper training set* (See Figure 2.8), which include instances of a category (positive examples) and instances of the sibling categories (negative examples). In this thesis we only consider Hierarchical Training Sets for two reasons. In [CM03], we already showed that hierarchical training sets perform better than proper training sets in the text categorization domain. Second, when no training example is associated to internal categories, proper training sets cannot be used, since it would be impossible to build a classifier.

A different issue we face concerns the representation of training examples. In the text categorization domain, Apté et al. [ADW94] propose two different types of representations: the same feature vector for all categories or several specialized feature vectors for different categories. The former is obtained by selecting features

from a *universal* set of features built by examining all examples in the training set, while the latter is obtained by selecting a feature set from several *local* sets of features built for each category by examining only examples of that category.

It has been observed that increasing the number of categories leads to an increase in the number of necessary features [BL97a]. To keep the dimensionality problem under control, it is possible to use local feature sets. On the other hand, the uniqueness of the feature set permits the application of several statistical and machine learning algorithms (e.g. nearest neighbour or naive Bayes classifier) defined for multi-class problems. These algorithms are appropriate for single-class categorization and are theoretically founded on the assumption that all examples are points of the same (multidimensional) feature space.

In the context of hierarchical classification a different, somewhat intermediate, solution can be adopted. Examples of both an internal category c and its subcategories are represented by means of the same feature set in order to build a classifier that assigns examples in c to one of its direct subcategories. However, different internal categories may have different feature sets. In other terms, by taking into account the hierarchy, it is possible to define several representations (sets of features) for each example. Each representation is useful for the classification of an example at one level of the hierarchy.

For instance, examples of the general category "Mammals" can be well represented by general features like "has bones", while examples concerning specific classes (e.g., "Cats") are better represented by specific features like "can mew". In the case of hierarchies representing is-a relations between categories, this multiple representation of examples corresponds to having several abstractions of the same entity, each of which is appropriate for a particular decision problem.

In our hierarchical categorization framework 2.9, we use this multiple representation of examples, which permits the application of multi-class learning algorithms for the induction of classifiers associated at each internal node. The outputs of the classifier associated to the category c are the degrees of membership of the input example to all direct subcategories. When the degrees of membership (or scores) are all lower than the corresponding automatically detected thresholds, the example is assigned to the category c.

The classification of a new example is performed by searching the hierarchy of categories. Search proceeds top-down from the root to the leaves according to a greedy strategy. When the example reaches an internal category c, it is represented on the basis of the feature set associated to c. The classifier of category c returns a score for each direct subcategory. Then, among subcategories whose score is greater than the corresponding threshold, the one with the highest score is chosen. Search proceeds recursively from that subcategory, until no score is greater than the corresponding threshold or a leaf category is reached. The last crossed node in the hierarchy is returned as the candidate category for example classification (single-category classification). If search stops at the root, then the example is considered unclassified. It is noteworthy that the application of a classifier is always preceded by a change in the example representation according to the set of selected features. Since selected features are expected to be more specific for lower levels

categories, the example is represented at decreasing levels of abstraction during the classification process. This automated representation change is highly desirable in hierarchical categorization.

Another noteworthy observation is that the application of an exhaustive search strategy would be incorrect in this framework, since the different representations of a example make the classification scores incomparable across different nodes in the hierarchy. This is particularly evident for naive Bayes classifiers since posterior probabilities are defined on different probability spaces [4].

A special case is represented by categories with a unique direct subcategory. A probabilistic classifier would assign a unit probability to all examples that reach a category c with a single subcategory c', since no alternative to c' is given. In this case, the thresholding procedure cannot work properly: if the threshold is less than one, all examples that reach c would be passed down to c', thus committing some errors for those examples that actually belong to c; otherwise, no example would be passed down to c', thus committing some errors for those examples that should be actually classified in a subcategory of c. To avoid this problem, a dummy sibling category of c' is introduced during the learning process. The training examples associated to the dummy subcategory are only those associated to c. The effect is that examples of c can be considered as negative examples for all subcategories of c itself. Therefore, the prior probability of all direct subcategories of c do not sum to 1.0 since the possibility that the example belongs to no subcategory should be taken into account. While the dummy category is used during the learning process, it plays no role during the classification process, since scores associated to the dummy category are not considered. The assignment of the example to c is based only on the thresholds, whose bottom-up automated determination permits to take into account the final effect of local decisions taken by the classifier associated to c.

2.4.2 Automated threshold determination

As pointed out in the previous section, a classifier is learned for each internal category c of the hierarchy. This classifier is used to decide, during the classification of a new example, which category $c_i \in DirectSubCategories(c)$ is the most appropriate to receive the example. In general, however, an example should not be necessarily passed down to a subcategory of c. This makes sense in the case that:

1. The example to be classified belongs to a general category rather than a specific category, or

2. The example to be classified belongs to a specific category that is not present in the hierarchy and makes more sense to classify the example in the "general category" rather than in a wrong category.

To support the classification of examples also in the internal categories of the hierarchy, it is necessary to compute the thresholds that represent the "minimal

[4] A probability space is a triple (Space, $\boldsymbol{S}$, P) on the domain S, where (S, $\boldsymbol{S}$) is a measurable space, $\boldsymbol{S}$ are the measurable subsets of S, and P is a measure on $\boldsymbol{S}$ with P(S)=1. Different features sets associated to internal categories define different measurable spaces

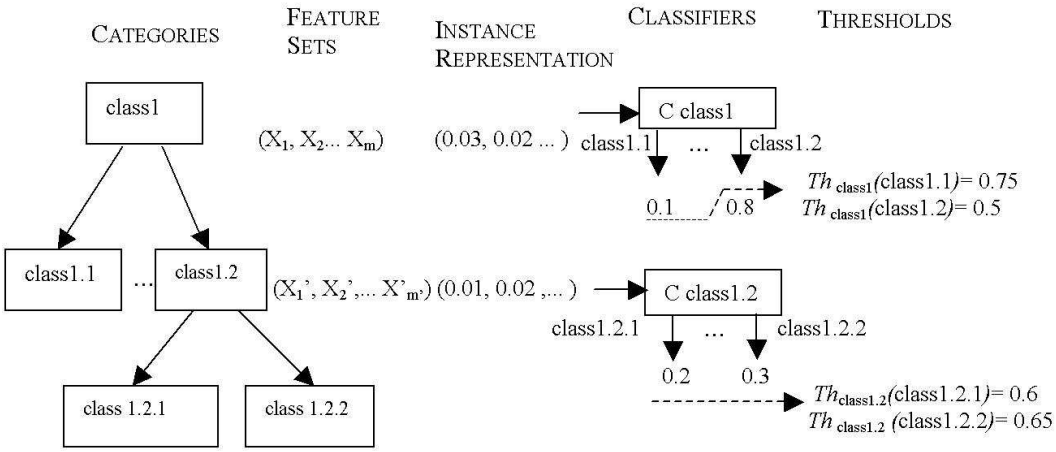

Figure 2.9: Classification of a new instance. On the basis of the scores returned by the first classifier (associated to the category *class1*) the example is passed down to *class1.2*. The scores returned by the second classifier (associated to the category *class1.2*), are not high enough to pass down the example to either *class1.2.1* or *class1.2.2*. Therefore, the example is classified in the *class1.2* category.

score" (returned by the classifier) such that an example can be considered belonging to a direct subcategory. More formally, let $\gamma_{c\rightarrow c'}(d)$ denote the score [5] returned by the classifier associated to the internal category c when the decision of classifying the example d in the subcategory c' is made. Thresholds are used to decide if a new testing example is characterized by a score that justifies the assignment of such a example to c'. Formally, a new example d temporary assigned to a category c will be passed down to a category c' if $\gamma_{c\rightarrow c'}(d) > Th_c(c')$, where $Th_c(c')$ represents "minimal score" such that an example assigned to c can be considered belonging to c'.

The algorithm for automated threshold determination is based on a bottom-up strategy and tries to minimize a measure that is based on a tree distance.

Before describing the algorithm and the used measure, some useful notations are introduced:

1. $Training(c)$ is the set of positive examples in the hierarchical training set of category c;
2. $Taining(c/c_i) = \Big(Training(c) \bigcup_{c_j \in DirectSubCategories(c)} Training(c_j)\Big) - Training(c_i)$ is the set of positive examples in $Training(c)$ but not in $Training(c_i)$;
3. $DirectSubCategories(c)$ is the set of direct subcategories of c in the hierarchy;
4. $\gamma_c(c_i) = \lfloor \gamma_{c\rightarrow c'}(d) | d \in Training(c_i) \rfloor$ is the list of values taken by the classifier for all examples of category c_i (or a subcategory);
5. $\gamma_c(\neg c_i) = \lfloor \gamma_{c\rightarrow c'}(d) | d \in Training(c/c_i) \rfloor$ is the list of values taken by the classifier for each example in c or a direct subcategory of c different from c_i;

[5] In the case of naive Bayes Classifier $\gamma_{c\rightarrow c'}(d) = P_c(c'|d)$

6. $V = \gamma_c(c_i) \cup \gamma_c(\neg c_i)$sorted in ascending order;

The algorithm (see Algorithm 2.1 [MCLA04]) takes in input the categories c and c', where c' is a child of c, and returns the threshold associated to c' ($Th_c(c')$). $Th_c(c')$ is determined by examining the sorted list V of classification scores and by selecting the middle point between two values in V such that the expected error is minimized. This error is estimated on the basis of the distance between two nodes in a tree structure (see Definition 2.1)

Algorithm 2.1 *Automated Threshold definition algorithm for a category* c_i

find_ thresholds*(c,c',thresholdSet) {*
if not leaf(c') then
$\forall c'' \in SubClasses(c')$ *//recursive bottom-up threshold determination*
thresholdSet←find_ thresholds(c',c'',thresholdSet);
compute_ and_ sort(V,c,c');
$Th_c(c') \leftarrow 0$;
bestError$\leftarrow \infty$;
$\forall k = 0, ..., |V|$ { *//choose a possible threshold*
if k=0 then threshold$\leftarrow V[1] - \epsilon$;
elseif k=|V| then threshold $\leftarrow$ *V[k];*
else threshold $\leftarrow$ *(V[k]+V[k+1])/2;*
error $\leftarrow$ *0;* $\forall v \in \gamma_c(c')$ { *//compute tree distance-based errors*
let $d \in Training(c')s.t.v = \gamma_{c \to c'}(d)$
if $v > threshold$ *then*
$error+ = \delta_{Hierarchy(c')}(class(d), classify(d, thresholdSet, Hierarchy(c')))$
else
$error+ = \delta_{Hierarchy(c')}(class(d), c)$;
}
$\forall v \in \gamma_c(\neg c')$ {
let $d \in Training(c/c')$ *s.t.* $v = \gamma_{c \to c'}(d)$
if $v > threshold$ *then*
$error+ = \delta_{Hierarchy(c)}(class(d), classify(d, thresholdSet, Hierarchy(c')))$
else
$error+ = 0$;
}
if error$<$ *bestError then //choose the best threshold*
$Th_c(c') \leftarrow threshold$; *bestError* $\leftarrow$ *error;*
}
$thresholdSet \cup \{< c', Th_c(c') >$;
}

Definition 2.1 *(tree distance)*

Let Categories be the set of all the categories, the tree distance $\delta_{Hierarchy}$ *is a function* $\delta_{Hierarchy} : Categories \rightarrow \Re$ *that associates two categories* $c_1, c_2 \in Categories$ *with a real value such that the following conditions are fulfilled:*

I $\forall c_1, c_2 \in Categories\ 0 = \delta_{Hierarchy}(c_1, c_1) \leq \delta_{Hierarchy}(c_1, c_2) = \delta_{Hierarchy}(c_2, c_1)$

II $\forall c_1, c_2 \in Categories : \delta_{Hierarchy}(c1, c2) = 0 \Longrightarrow c_1 = c_2$

III $\forall c_1, c_2, c_3, c_4 \in Categories : \delta_{Hierarchy}(c_1, c_2) + \delta_{Hierarchy}(c_3, c_4) \leq$
$\leq max\{\delta_{Hierarchy}(c_1, c_3) + \delta_{Hierarchy}(c_2, c_4), \delta_{Hierarchy}(c_1, c_4) + \delta_{Hierarchy}(c_2, c_3)\}$

In a tree distance, the dissimilarity between two categories is reproduced as the sum of the weights of all edges of the (unique) path connecting the two categories in the hierarchy [EMTB00]. When a unit weight is associated to each edge (as in our proposal) the dissimilarity is the length of the path. Intuitively, the automated thresholding algorithm tries to compute thresholds by minimizing the distance between the true class of an example and the class returned by the hierarchical classifier.

The computation proceeds bottom-up, from leaves to the root. In [CM03] a top-down approach was proposed. However, this approach suffers from two problems:

- It is conservative in the sense that it tends to classify examples in higher categories;
- When a threshold is defined, it is not possible to take into account the possibly wrong decisions taken by classifiers at lower levels of the hierarchy.

Another difference is that in our previous work thresholds were determined by maximizing the *FScore* [Seb02] of the hierarchical classification on training examples. Although this approach gives promising results, it presents the limitation that the distance between "target" and "assigned" categories in the hierarchy is not considered when a misclassification error occurs.

2.4.3 Learning Complexity

In the hierarchical classification framework, the original learning problem is partitioned into smaller subproblems each of which can be efficiently managed. This leads to an efficiency gain, with respect to the flat classifier, that depends on the number classes associated to each learned classifier. More formally, let

- f(number of classes, number of training examples, number of features) be the learning complexity of a generic classification algorithm.
- r be the total number of classes
- n be the number of training examples
- a be the number of features

- d be the depth of the hierarchy
- k number of children of a generic internal node (for simplicity, in this analysis we suppose that k is constant).

Then the complexity of a flat classifier is: $f(r, n, a)$. For what concerns the complexity of the hierarchical framework, it is:

- $f(k, n, a)$ for the first level;
- $k \cdot f(k, n/k, a)$ for the second level, in the worst case that all examples are classified in lower categories
- $k^2 \cdot f(k, n/k^2, a)$ for the third level.

By generalizing, the complexity of the hierarchical framework is:

$$\sum_{i=1}^{d} k^i f(k, \frac{n}{k^i}, a) \tag{2.19}$$

If we use a naive Bayes classifier, the complexity of the learning phase is linear in the number of training examples, in the number of features and in the number of classes [Mit97]. In such a case the time complexity of a flat classifier is

$$O(n \cdot a \cdot r) \tag{2.20}$$

while in the case of hierarchical framework, it is:

$$O\Big(\sum_{i=1}^{d} k^i \cdot (\frac{n}{k^i} \cdot k \cdot a)\Big) = O\Big(\sum_{i=1}^{d} (n \cdot k \cdot a)\Big) = O(d \cdot n \cdot k \cdot a) \tag{2.21}$$

Both are linear in the number of training examples and in the number of features. The difference is that the complexity of a flat classifier is linear in the number of classes, while the complexity of the hierarchical framework is linear in the product of the number of children of each node and the depth of the tree. Under the assumption of a balanced hierarchy with constant branching factor k, we have $d = log_k r$. Therefore the complexity of the hierarchical framework is

$$O(n \cdot a \cdot log_k r) \tag{2.22}$$

Comparing the 2.22 with 2.20, we note that, in case of naive Bayesian classification, the hierarchical framework is particularly efficient when the number of categories is quite high. If we consider that the Bayesian classifier is particularly accurate when the number of attributes is relatively small [DP97], we expect that the naive Bayesian classifier takes great advantage of the use of the Hierarchical framework. This is confirmed by results on the application of the proposed algorithm in text categorization (see Section 4.1).

2.4.4 Related Work

In the literature, several approaches have been proposed that face the problem of hierarchical classification. Most of them have been applied in the context of text categorization. They differ in terms of several aspects that principally involve the definition of the problem, the example representation and the learning strategy. As for the definition of the problem, a classifier that classifies examples into L categories can be formulated in two different ways: either a binary classifier is induced for each category, or a *1-of-L* (or multi-class) classifier is learned to determine whether a new example belongs to one of the L categories [Seb02]. Our approach is based on the second formulation.

As for the example representation, each example can be described by several sets of features, each of which is useful for the classification of the example at one level of the hierarchy. In this way, general features and specific features are not forced to coexist in the same feature set.

As for the learning process, it is possible to consider the hierarchy of categories either in the formulation of the learning algorithm or in the definition of the training sets. Training sets can be specialized for each internal node of the hierarchy by considering only examples of the sub-hierarchy rooted in the internal node (hierarchical training set). This is an alternative to using all examples for each learning problem like in flat classification.

Some of these aspects have been considered in related works. In particular, in the seminal work by Koller and Sahami [KS97] a different feature set is built for each node in the hierarchy. For the learning step, two Bayesian classifiers are compared, namely the naïve Bayes and KDB [Sah96]. A distinct classifier is built for each internal node (i.e., split) of the hierarchy. In the classification step, which proceeds top-down, it is used to decide to which subtree to send the new example. There is no possibility of recovering errors performed by the classifiers associated to the higher levels in the hierarchy. Two limitations of this study are the possibility of associating instances only to the leaves of the hierarchy and the effectiveness of the learning methods only for relatively small vocabularies ($<$100 features).

McCallum et al. [MRMN98] proposed a method based on the naive Bayes learner. A unique feature set is defined for the entire training set. Because of the uniqueness of the feature set, Bayesian classifiers associated at internal nodes are homogeneous, and, as formalized by Mitchell [Mit98] the hierarchical organization of homogeneous classifiers is equivalent to a single flat classifier. In other terms, the hierarchical structure would have no practical impact on the classification process. This explains why, in the learning step, McCallum et al. use a statistical technique known as shrinkage to smooth parameter estimates for lower-level categories with parameter estimates for their ancestors in the category hierarchy. For the classification step, the authors compare two techniques: exploring all possible paths in the hierarchy and greedily selecting the most probable one/two branches as done in [KS97]. Results show that greedy selection is more error prone but also more computational efficient. As in the previous work, all examples can be assigned only to the leaves of the hierarchy.

Mladenić [Mla98b] used the hierarchical structure to decompose a problem into a set of subproblems corresponding to categories (nodes in the hierarchy). For each subproblem, a naive Bayes classifier is built from a set of positive examples, which is constructed from examples in the corresponding category node and all examples of its subtrees, and a set of negative examples corresponding to all remaining examples. The set of features selected for each category can be different. The classification applies to all the classifiers (nodes) in parallel, using some pruning of unpromising nodes. In particular, an example is passed down to a category only if the posterior probability for that category is higher than a user-defined threshold. Contrary to the previous work, examples can be assigned to any node of the hierarchy.

In the work by D'Alessio et al. [DMSK00] examples are associated only to leaf categories of the hierarchy. Two sets of features are associated to each category, one is positive (features extracted from exampless of the category) while the other is negative (features extracted from examples of sibling categories in the hierarchy). In addition to contributing to feature extraction, the training set is also used to estimate feature weights and a set of thresholds, one for each category. Classification in a given category is based on a weighted sum of feature occurrences that should be greater than the category threshold. Both single and multiple classifications are possible for each testing example. The classification of a example proceeds top-down either through a single path (one-of-L classification) or through multiple-paths (binary classification). An innovative contribution of this work is the possibility of restructuring an initial hierarchy or building a new one from scratch.

Dumais and Chen [DC00] use the hierarchical structure for two purposes. First, to train several Support Vector Machines (SVM's), one for each intermediate node. The sets of positive and negative examples are constructed from examples of categories at the same level, and different feature sets are built, one for each category. Second, to classify examples by combining scores from SVM's at different levels. Several combination rules are compared, some requiring a category threshold to be exceeded to pass a test example down to descendant categories. Multiple classification of a examples is allowed for leaf categories, while the assignment of an example to intermediate categories is not considered.

In the system CLASSI by Ng et al. [NGL97], the hierarchical classification of examples is obtained by combining several linear classifiers according to a tree structure (hierarchical classifier). The tree structure corresponds to the hierarchy of categories, which means that a linear classifier is associated to each category. The output of the classifier defines a degree of membership of an example to a category. In the classification phase the hierarchical classifier receives an example and checks whether it belongs to any of the first level nodes. If the tested example activates any of the first level nodes, then the descendant categories of that node are tested recursively. Multiple classifications of examples is allowed while classification of examples in non-leaf categories seems not to be supported. Weights of each linear classifier are determined by means of the perceptron learning algorithm. The training set of each linear classifier includes all positive examples of the associated category (i.e., no hierarchical training set) and some selected examples of other categories.

In the work by Ruiz and Srinivasan [RS02] a variant of the Hierarchical Mixture of Experts (HME) model is used. A tree of backpropagation neural networks is used. Neural networks are of two types: experts and gates. The former take the feature-vector representation of an example as input and are trained to recognize whether the example belongs to a specific category. There are as many experts at the leaves of this tree-structured classifier as categories (leaves and non-leaves) in the hierarchy. Gating networks are the internal nodes of the tree-structured classifier and map the non-leaf categories of the hierarchy. They have two kinds of input: the feature-vector representation of an example and the output of the expert/gating networks below in the tree. Their role is that of restricting the number of experts to be activated for a given example. Indeed, the classification of an example proceeds top-down in the tree of neural networks, starting from the gate at the root towards the experts at the leaves. Multiple classification is supported. The gates are trained to recognize whether or not any of the categories of their descendants is present in the example. The experts are trained to recognize the presence or absence of particular categories. Therefore, the set of positive examples for an expert includes examples of the uniquely associated category while the set of positive examples for a gate includes all training examples of the set of associated categories. Some form of filtering is used for negative examples, since unbalanced data sets may affect the learning capability of backpropagation neural networks. Different feature sets are selected for each expert and gating network. The proposed method is tested on some MEDLINE records. Only categories with positive examples are selected, since this method cannot work when intermediate categories have no positive examples.

A hierarchical classifier combining several neural networks is also proposed by Weigend et al. [WWP99]. Neural networks at internal nodes are *meta-topic classifier* while those at the leaves are *individual classifiers*. The method has been devised and tested only on two-levels hierarchies, although the extension to more than two levels should be straightforward. The dimensionality reduction of the original feature space is obtained by means of two statistical techniques: Latent Semantic Indexing to transform the original feature set into a new set of features that are a linear combination of the original features, and c2 statistics to select the most discriminant features. Moreover, selected feature sets can either be specific for each category or unique for all categories. The former resulted in better performance on the Reuters dataset, thus empirically confirming Mitchell's finding [Mit98] also for classifiers based on neural networks.

Blockeel et al. [BBD+02] defined a specific decision tree induction algorithm for the case of hierarchical multi-category classification. In particular, the authors use Clus in order to build predictive clustering trees. They use a distance measure to support a Top-Down construction of trees. Examples can be assigned only to the leaves of the hierarchy.

Sun and Lim [SL01] have proposed the use of category-similarity measures and distance-based measures to consider the degree of misclassification in measuring the classification performance. Experiments were performed on the Reuters-22173 collection with a SVMlight Version 3.50 implemented by Joachims [Joa].

2.5 Conclusions

In this chapter we formally defined the classification problem in data mining. We focused the attention on the naive Bayesian classification that we deeply described. Furthermore, we illustrate the classification problem when hierarchical relations between target categories are taken into account. We also propose a general framework for hierarchical classification of examples. It can be applied to any classifier that returns a degree of membership, such as probabilistic or distance-based classifier.

Chapter 3

Naive Bayesian Multi-Relational Classification

In the previous chapter we considered a particular kind of "structure", namely relations between categories of the unit of analysis. In this chapter we consider a different kind of "structure", that is, the structure represented by the occurrence of relations between the units of analysis and/or the units of observation. At this aim, we resort to a new branch of data mining research, namely multi-relational data mining.

3.1 Multi-relational Data Mining

Multi-relational data mining is a new branch of data mining research that can deal with complex data where relations between units of analysis and/or the units of observation are taken into account. In particular, looking at the problem in a database prospective, multi-relational data mining overcomes the problem of *single table assumption* [Wro01] that assumes that the training set can be represented as a single relational table, where each row corresponds to an example and each column to a predictor variable or to the target variable. This assumption is made in classical data mining and seems quite restrictive in some data mining applications, where data are stored in a database and are organized into several tables for reasons of efficient storage and access. In this context, both predictor variables and the target variable are represented as attributes of distinct tables (relations) eventually related each other by means of foreign key constraints defining a structure in the data. Relational Data mining looks for patterns that involve multiple relations in a relational database. It does so directly, without transforming it in a single table first and then looking for patterns in it.

For example, suppose we have a database storing information about the access and the use of a website. It stores information about users, sessions, available services and requested services. An example of a database schema is shown in figure 3.1.

For analysis purposes, suppose we are interested in statistics concerning "ses-

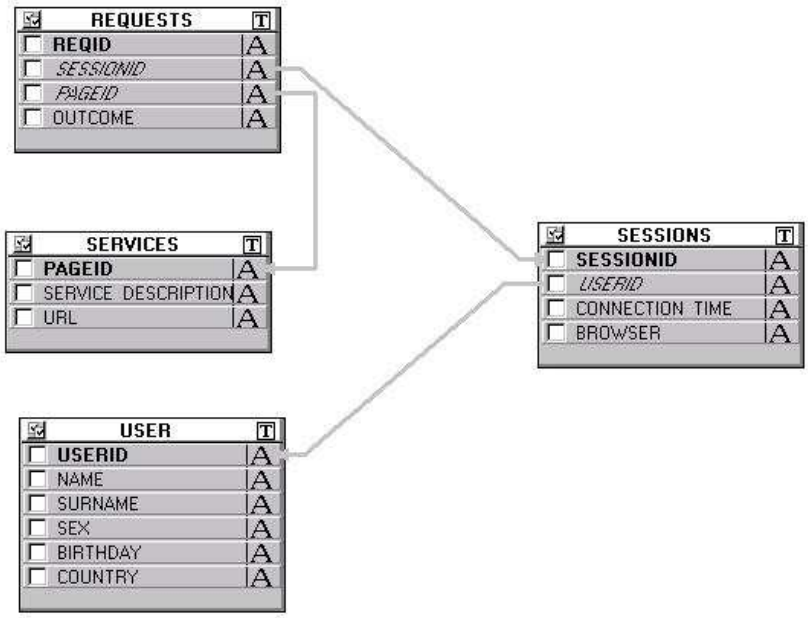

Figure 3.1: Simple schema of a database that stores information about the access and the use of a website

sions". This means that the "sessions" are our basic observable entities in this statistical study (Units of Analysis). The simplest approach takes into account data stored in the table *sessions*. From such data, traditional data mining algorithm can produce different kinds of knowledge: classifiers, association rules, clusters etc. However, all extracted knowledge concerns only information stored in the *sessions* table.

Suppose that, in this study, we are interested in taking into account other information i.e. users and type of requested services. These entities are not the principal entities in the study but are just collected useful information (units of observation). In order to take into account such information, we can add to the sessions table as many attributes as we want: we might add a column representing the user's country, we might add his sex and so on. Adding information about the user is easy and straightforward (it is a simple JOIN operation), but the situation changes when we add information about requested services. In this case, it is possible that, in the same session, the user uses more than one service.

In this case, the single table assumption turns out to be a severe limitation and two alternatives are possible. First, we could make one tuple for each requested service in the *sessions* table. Thus, if a user requests two services in the same session, the session will be represented by 2 tuples, each duplicating both session and user information. Obviously, this "redundancy" of data leads to several disadvantages: there is a waste of space and an error in training data is duplicated.

In addition, from a statistical point of view, we are moving our unit of analysis

from "sessions" to "requests" because each individual represents a request and not a session. Thus, our analysis will be about "requests", not "sessions", which is not what we might want.

The second alternative avoids the problems with redundancy and multiple rows, and thus allows analysis methods to operate properly. The method consists in creating a summarizing table that contains information about sessions and users, but also a simple summary of information about requested services (e.g. number of requested services). Obviously, this alternative avoids problems of the first one at the expense of information details. This approach is known in the literature as "propositionalization" and formally consists in the construction of features that capture relational properties of the learning examples [KLF01] [KRZ+03]. In the literature, propositionalization is often related to the problem of feature construction and predicate invention (in Inductive Logic Programming, ILP) [Sta96]. However, the output of a propositionalization algorithm is often a propositional table consisting of a multitude of attributes that are often useless for learning. For this reason, propositionalization is often followed by a feature selection process that aims to significantly reduce the number of features on which the propositional learning algorithm works [ACRF04].

An alternative to the combined use of propositionalization and feature selection is represented by the *Structural Approaches*. Structural approaches take into account the structure of the original data and use the database schema as it originally is, without transformations. Historically, structural approaches are not new in the Machine Learning research and, in particular, ILP community has been working for a number of years on a powerful representation that allows to represent such information. In fact, ILP make use of first order logic representation formalism that allows to represent both propositional and structure information. For example, a possible pattern an ILP system can discover is:

$$
\begin{aligned}
&session(SessionID, UserID, Connection\_Time, Browser) \leftarrow \\
&\quad Connection\_Time > 3', \\
&\quad user(UserID, Name, Surname, Sex, Birthday, Country), \\
&\quad Birthday{>}'01/01/1980', \\
&\quad request(REQID, SessionID, PageID, Outcome), \\
&\quad Outcome =' ok'.
\end{aligned}
$$

ILP approaches typically work on a set of main-memory Prolog facts and facts correspond to tuples stored on relational databases. In ILP systems, some pre-processing is required in order to transform tuples into facts. However, this has some disadvantages. First, only part of the original hypothesis space implicitly defined by foreign key constraints can be represented after some pre-processing. Second, much of the pre-processing may be unnecessary, since a part of the hypothesis described by Prolog facts space may never be explored, perhaps because of early pruning. Third, in applications where data can frequently change, pre-processing has to be frequently repeated. Finally, database schemas provide the learning system free

of charge with useful knowledge of data model that can help to guide the search process. This is an alternative to asking the users to specify a language bias, such as in ILP systems.

Typically, ILP methods are characterized by an approach which is purely logic-based. Main issues investigated in the formulation of these methods are generalization models, generalization/specialization operators, folding/unfolding logical theories, etc. In recent years the research interests in ILP have moved towards methods based on a statistical approach, where the way of dealing with uncertainty is a key issue. The main focus of this chapter is on these statistical approaches which can be considered an "upgrade" of statistical approaches in order to handle relational data by means of first-order logic representation.

In the following section we report some statistical methods that are able to handle multi-relational data and in the subsequent sections we propose two different approaches representing an upgrade of statistical methods and, in particular, naive Bayesian classification to the multi-relational setting.

3.2 Statistical approaches to multi-relational data mining

In the literature, several approaches have been proposed that face the problem of statistical classification in multi-relational data mining. They are mainly based on the upgrade of standard statistical approaches such as, Bayesian networks, decision trees, naive Bayesian, Markov networks and logistic regression to the multi-relational setting.

In this section we briefly review some of cardinal and well known approaches.

Probabilistic Relational Models (PRMs) [TSK01] [FGKP99b] extend the standard attribute-based Bayesian Network representation (see 2.1.7) to incorporate a much richer relational structure. In particular, these models allow properties of an entity to depend probabilistically on properties of other related entities. In the learning task, like Bayesian Networks, PRMs have two variants: parameter estimation and structure learning. In the parameter estimation task, the qualitative dependency structure is assumed to be known, so the input consists of the schema and the training database as well as a qualitative dependency structure. The parameter learning task consists in filling in the parameters that define the conditional probability distributions of the attributes. In the structure learning task, the goal is to extract an entire PRM from the training database alone. This learning problem is at least as hard as Bayesian Networks, thus, to keep the computational complexity under control, a heuristic search is used. The search is based on a Bayesian score defined as the probability that the structure is adequate, given the data. The search is structured so that it first explores dependencies within entities, then between entities that are directly correlated, then between entities that are two links apart and so on. An improved version of PRMs is represented by SRMs (Statistical Relational Models) [Get01a]. Differently from PRMs, SRMs have a different semantics and are able to capture tuple frequencies in the database.

Taskar et al., in 2002 [TAK02] proposed Relational Markov networks (RMNs). RMNs are a relational extension of discriminatively trained Markov networks. In particular, RMNs compactly define a Markov network over a relational dataset. The graphical structure of an RMN is based on the relational structure of the domain. In this approach, the use of undirected graphical models avoids the difficulty of defining a coherent generative model for graph structures in directed models, increasing in flexibility. The parameter estimation uses conjugate gradient combined with approximate probabilistic inference based on the belief propagation used in Bayesian Networks. In RMNs, however, the structure of learning domain, determining which direct interactions are explored, is prespecified by the relational template. This precludes the discovery of deeper and more complex regularities than other approaches.

Relational Bayesian Classifier (RBC) [NJG03] is a simple modification of the naive Bayesian classifier that allows to deal with relational data. The approach is based on the transformation of the original database schema in a single table containing multisets (which is not in first normal form). The resulting table is used in the classical naive Bayesian probability estimation. Multisets conditional probabilities are estimated by simple approaches, namely "Average value", "Independent Value" and "Average Probability". The "average value" estimator corresponds to flattening the data by averaging. The "Independent Value" estimator provides to duplicate instances assuming that each value of a multiset is independently drawn from the same distribution. The "average probability" estimator uses probabilities estimated with the previous estimator and computes the mean of the conditional probabilities over the multiset. This approach, however, has a limited representation power, in fact, it cannot represent more than two links apart.

Ngo and Haddawy proposed a language for representing context-sensitive probabilistic knowledge and provided a declarative semantics for their Probabilistic Logic Programs (PLPs) [NH97]. PLPs are a first order extension of Bayesian networks. A similar approach has been proposed by Kersting and De Raedt who introduced Bayesian logic programs [KD00]. Bayesian logic programs are a reformulation and a simplification of probabilistic logic programs [NH97]. The added value in Bayesian logic programs concernes the fact that they can serve as a kind of common kernel to other approaches that combine first order logic with Bayesian networks (as PRMs and PLPs) because they can essentially be used to represent the same knowledge.

Blockeel and De Raedt proposed the system TILDE (Top-Down induction of first order decision trees) [BD98]. TILDE is an upgrade of a propositional decision tree learner to first order logic. In particular, C4.5 [Qui86] for binary classification is a special case of TILDE. TILDE is the result of a study that aimed to make use of top down induction of decision trees in ILP. This study shows that first order logical decision trees (induced by TILDE) are more expressive than the flat non-recursive logic programs induced by typical ILP systems for classification tasks.

Neville and her colleagues proposed Relational Probability Trees (RPTs) [NJFH03]. RPTs extend standard probability estimation trees to a relational setting. The proposed algorithm learns the structure and the parameters of an RPT by searching over a space of relational features that use aggregate functions to dynamically propositionalize relational data and create binary splits.

Pompe and Kononenko [PK95] proposed a method based on a two-step process. The first step uses the ILP-R system [PK94] to learn a hypothesis in the form of a set of first-order rules and then, in the second step, the rules are probabilistically analyzed. During the classification phase, the conditional probability distributions of individual rules are combined naively according to the naïve Bayesian formula.

Flach and Lachiche proposed a similar two-step method, however, unlike the previous one, there is no learning of first-order rules in the first step. Alternatively, a set of patterns (first-order conditions) is generated that are used afterwards as attributes in a classical attribute-value naive Bayesian classifier [FL04]. 1BC, the system implementing this method, views individuals as structured objects and distinguishes between structural predicates referring to parts of individuals (e.g. atoms within molecules), and properties applying to the individual or one or several of its parts (e.g. a bond between two atoms). An elementary first-order feature consists of zero or more structural predicates and one property.

An evolution of 1BC is represented by the system 1BC2 [FL04] [LF03a], where no preliminary generation of first-order conditions is present. Predicates whose probabilities have to be estimated are dynamically defined on the basis of the individual to classify. Therefore, this is a form of lazy learning, which defers processing of its inputs (i.e., the estimation of the posterior probability according to the Bayesian statistical framework) until it receives requests for information (the class of the individual). Computed probabilities are discarded at the end of the classification process. Probability estimates are recursively computed.

Popescul et al. 2003 proposed Structural Logistic Regression. In particular, their approach integrates classical logistic regression (See section 2.1.3) with feature generation from relational data. The feature generation process is defined as the search in the space of relational database queries. The search is based on a top-down approach that applies refinement operators. The search is performed by means of a breadth-first algorithm and is guided by heuristics based on statistical criteria. In refinements of queries, cyclic paths are not supported and aggregating operators are also used. The approach has been used in link prediction. An interesting aspect of such a method is that it provides a general framework in which different continuous outcome methods can be included (For example, Poisson regression, linear regression etc.).

In the literature several multi-relational data mining approaches for regression tasks have also been proposed. Some examples are: FORS [Kar95] [KB97], FFOIL [Qui96], SRT [Kra96], S-CART [Kra99] [KW01a] and TILDE-RT [Blo98] and finally, our proposal Mr-SMOTI [ACM03]. However, it is out of the scope of this thesis to describe these approaches in depth.

3.3 A Multi Relational approach for Naive Bayesian Classification: Mr-SBC

In classical classification stetting, data are generated independently and with an identical and unknown distribution P on some domain X and are associated with

a value in some domain Y according to an unknown function g. The domain of g is spanned by m independent (or predictor) random variables X_i (both numerical and categorical), that is $X = X_1 \times X_2 \times \cdots \times X_m$, the goal is to predict the dependent (or response or targer) symbolic variable Y ($Y = C_1, C_2, \ldots, C_L$). An inductive learning algorithm takes a training sample $S = \{(x, y) \in X \times Y | y = g(x)\}$ as input and returns a function f which is hopefully close to g on the domain X.

According to section 2.2, a well-known solution to classification is represented by the Naive Bayesian Classifiers, which aim to classify any $x \in X$ is the class maximizing the posterior probability $P(C_i|x)$ that the observation x is of class C_i, that is:

$$f(x) = argmax_i \; P(Y = C_i | X_1 = x_1, X_2 = x_2, ...X_m = x_m)$$
$$i = 1..L$$

By applying the Bayes theorem, $P(C_i|x)$ can be reformulated as follows:

$$P(Y = C_i | X_1 = x_1, X_2 = x_2, ...X_m = x_m) =$$
$$\frac{P(X_1 = x_1, X_2 = x_2, ...X_m = x_m | Y = C_i) P(Y = C_i)}{\sum_{j=1}^{L} P(X_1 = x_1, X_2 = x_2, ...X_m = x_m | Y = C_j) P(Y = C_j)}$$

where the term $P(x|C_i)$ is in turn estimated by means of the naive Bayes assumption:

$$P(X_1 = x_1, X_2 = x_2, ...X_m = x_m | Y = C_i) = \prod_{j=1}^{m} P(X_1 = x_1 | Y = C_i) \qquad (3.1)$$

Thus, the discriminant function is:

$$f(x) = argmax_i \; P(Y = C_i) \prod_{j=1}^{m} P(X_1 = x_1 | Y = C_i) \qquad (3.2)$$

We already discussed this assumption in section 2.2.3 and, citing some seminal works, we observed that, even in the case that the independence assumption is violated by a wide margin, the naive Bayesian classifier can give good results [DP97].

In this chapter, we present a new approach to the problem of learning classifiers from relational data. In particular, we intend to extend the naive Bayes classifier to the case of relational data. Our proposal is based on the induction of a set of first-order classification rules in the context of naive Bayesian classification. Studies on first-order naive Bayes classifiers have already been reported in the literature (see section 3.2). In particular, Pompe and Kononenko [PK95] proposed their own method and Flach and Lachiche [FL04] proposed two methods: 1BC and 1BC2.

An important aspect of both Pompe and Kononenko's approach and 1BC, is that they keep the phases of first-order rules/conditions generation and of probability estimation separate. In particular, Pompe and Kononenko use ILP-R to induce first-order rules [PK94], while 1BC uses TERTIUS [FL00] to generate first order

features. Then, the probabilities are computed for each first-order rule or feature. In the classification phase, the two approaches are similar to a multiple classifier because they combine the results of two algorithms. However, most first-order features or rules share some literals and this approach takes into account the related probabilities more than once. To overcome this problem it is necessary to rely on an integrated approach, so that the computation of probabilities on shared literals can be separated from the computation of probabilities on the remaining literals.

Systems implementing one of the three approaches above work on a set of main-memory Prolog facts. In real-world applications, where facts correspond to tuples stored on relational databases, some pre-processing is required in order to transform tuples into facts. However, this has some disadvantages. First, only part of the original hypothesis space implicitly defined by foreign key constraints can be represented after some pre-processing. Second, much of the pre-processing may be unnecessary, since a part of the hypothesis described by Prolog facts space may never be explored, perhaps because of early pruning. Third, in applications where data can frequently change, pre-processing has to be frequently repeated. Finally, database schemas provide the learning system free of charge with useful knowledge of data model that can help to guide the search process. This is an alternative to asking the users to specify a language bias, such as in 1BC or 1BC2.

A different approach has been proposed by Getoor [Get01a] where the Statistical Relational Models (SRM) (see section 3.2) are learned taking advantage of the tight integration with a database. However, SRMs are models based on Bayesian Networks. The main difference is that the input of a SRM learner is both the relational schema of the database and the tuples of the tables in the relational schema.

In this chapter the system Mr-SBC (Multi-Relational Structural Bayesian Classifier) [CAM03] is presented. It implements a new learning algorithm based on an integrated approach of first-order classification rules with naive Bayesian classification, in order to separate the computation of probabilities of shared literals from the computation of probabilities for the remaining literals. Moreover, Mr-SBC is tightly integrated with a relational database as in the work by Getoor, and handles categorical as well as numerical data through a discretization method.

In the next subsection the problem is introduced and defined. The induction of first-order classification rules is presented in the following subsection, afterwards the discretization method is explained and the classification model is illustrated. Finally, experimental results are reported in section 4.1, in the context of document engineering and in section 5.1 in other application domains.

3.3.1 Formal Definition of the problem

In traditional classification systems that operate on a single relational table, an observation (or individual) is represented as a tuple of the relational table. Conversely, in Mr-SBC, which induces first-order classifiers from data stored in a set $T = T_0, T_1, \ldots, T_h$ of tables of a relational database, an individual is a tuple t of a target relation TR joined with all the tuples in T which are related to t following a

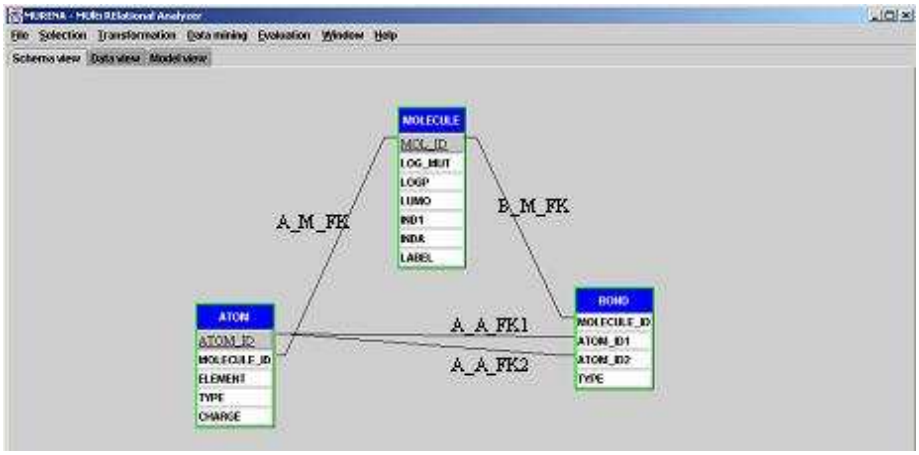

FIGURE 3.2: An example of a relational representation of training data of the Mutagenesis database

foreign key path. Formally, a foreign key path is defined as follows:

Definition 3.1 *A **foreign key path** is an ordered sequence of tables* $\vartheta = (T_{i_1}, T_{i_2}, \ldots, T_{i_s})$*, where*

- $\forall j = 1, ..., s, T_{i_j} \in T$
- $\forall j = 1, ..., s-1, T_{i_{j+1}}$ *has a foreign key to the table* T_{i_j}

In Figure 3.2 an example of foreign key paths is reported. In this case, S= {MOLECULE, ATOM, BOND} and the foreign keys are: A_M_FK, B_M_FK, A_A_FK1, A_A_FK2. If the target relation T is MOLECULE then five foreign key paths exists. They are:

- (MOLECULE)
- (MOLECULE, ATOM)
- (MOLECULE, BOND)
- (MOLECULE, ATOM, BOND)
- (MOLECULE, ATOM, BOND)

The last two are equal because the bond table has two foreign keys referencing the table atom.

A formal definition of the learning problem solved by MR-SBC is the following: Given:

- A training set represented by means of h relational tables $T = T_0, T_1, \ldots, T_h$ of a relational database D.
- A set of primary key constraints on tables in T.
- A set of foreign key constraints on tables in T.
- A target relation $TR(x_1, \ldots, x_n) \in T$

- A target discrete attribute y in TR, different from the primary key of TR.

Find:

A naive Bayesian classifier which predicts the value of y for some individual represented as a tuple in TR (with possibly UNKNOWN value for y) and related tuples in T according to foreign key paths.

3.3.2 Generation of first-order rules

Let R' be a set of first-order classification rules for the classes $\{C_1, C_2, \ldots, C_L\}$, and I an individual to be classified and defined as above. The individual can be logically represented as a set of ground facts, the only exception being the fact associated to the target relation TR, where the argument corresponding to the target attribute y is a variable Y. A rule $R_j \in R'$ covers I, if a substitution Θ exists, such that $R_j\Theta \subseteq I\Theta$. The application of the substitution to I is required to ground the only variable Y in I to the same constant as that reported in R_j for the target attribute. Let R be the subset of rules in R' that cover I, that is $R = \{R_j \in R' | R_j \; covers \; I\}$. The first-order naive Bayes classifier for the individual I, $f(I)$, is defined as follows:

$$f(I) = argmax_i P(C_i|R) = argmax_i \frac{P(C_i)P(R|C_i)}{P(R)} \tag{3.3}$$

The value $P(C_i)$ is the prior probability of the class C_i. Since $P(R)$ is independent of the class C_i, it does not affect $f(I)$, that is,

$$f(I) = argmax_i \; P(C_i)P(R|C_i) \tag{3.4}$$

The computation of $P(R|C_i)$ depends on the structure of R. Therefore, it is important to clarify how first-order rules are built in order to associate them with a probability measure. As already pointed out, Pompe and Kononenko use the first-order learning system ILP-R to induce the set of rules R'. This approach is very expensive and does not take into account the bias automatically determined by the constraints in the database. On the other hand, Flach and Lachiche use Tertius to determine the structure of first-order features on the basis of the structure of the individuals. The system Tertius deals with learning first-order logic rules from data lacking an explicit classification predicate. Consequently, the learned rules are not restricted to predicate definitions as in supervised inductive logic programming. Our solution is similar to that proposed by Flach since the structure of classification rules is determined on the basis of the structure of the individuals. The main difference, in the construction of rules, is that the classification predicate is considered.

All predicates in classification rules generated by Mr-SBC are binary and can be of two different types.

Definition 3.2 *A binary predicate p is a **structural predicate** associated to a table $T_i \in T$ if a foreign key FK in T_i exists that references a table $T_{i_1} \in T$. The first argument of p represents the primary key of T_{i_1} and the second argument represents the primary key of T_i.*

Definition 3.3 *A binary predicate p is a property predicate associated to a table $T_i \in S$, if the first argument of p represents the primary key of T_i and the second argument represents another attribute in T_i which is neither the primary key of T_i nor a foreign key in T_i.*

Definition 3.4 *A first order classification rule associated to the foreign key path ϑ is a clause in the form:*

$$p_0(A_1, y) : -p_1(A_1, A_2), p_2(A_2, A_3), \ldots, p_{s-1}(A_{s-1}, A_s), p_s(A_s, c)$$

. where

1. *p_0 is a property predicate associated to the target table TR and to the target attribute y.*
2. *$\vartheta = (T_{i_1}, T_{i_2}, ..., T_{i_s})$ is a foreign key path such that for each $k = 1, ..., s-1$, p_k is a structural predicate associated to the table T_{i_k}*
3. *p_s is a property predicate associated to the table T_{i_s}.*

An example of a first-order rule is the following:

$$molecule\_Label(A, active) : -molecule\_Atom(A, B), atom\_Type(B,' [22..27]').$$

Mr-SBC searches all possible classification rules by means of a breadth-first strategy and iterates over some refining steps. A refining step is biased by the possible foreign key paths and consists of the addition of a new literal, the unification of two variables and, in the case of a property predicate, in the instantiation of a variable. The search strategy is biased by the structure of the database because each refining step is made only if the generated first-order classification rule can be associated to a foreign key path. However, the number of refinement steps is upper bounded by a user-defined constant MAX_LEN_PATH.

3.3.3 Discretization

In Mr-SBC continuous attributes are handled through supervised discretization. Supervised discretization methods utilize the information on the class labels of individuals to partition a numerical interval into bins. The proposed algorithm sorts the observed values of a continuous feature and attempts to greedily divide the domain of the continuous variable into bins, such that each bin contains only instances of one class. Since such a scheme could possibly lead to one bin for each observed real value, the algorithm is constrained to merge bins in a second step. Merging of two contiguous bins is performed when the increase of entropy is lower than a user-defined threshold (MAX_GAIN). This method is a variant of the one-step method 1RD by Holte [Hol93] for the induction of one-level decision trees, that proved to work well with the Naive Bayes Classifier [DKS95]. It is also different from the one-step method by Fayyad and Irani [FI94] that recursively splits the initial interval according to the class information entropy measure until a stopping criterion based on the Minimum Description Length (MDL) principle is verified.

3.3.4 Computation of Probabilities

According to the naive Bayes assumption, the attributes are considered independent. However, this assumption is clearly false for the attributes that are primary keys or foreign keys. This means that the computation of $P(R|C_i)$ in equation 3.4 depends on the structures of rules in R. For instance, if R_1 and R_2 are two rules of class C_i, that share the same structure and differ only for the property predicates in their bodies

$$R1 : \beta_{1,0} : -\beta_{1,1}, ..., \beta_{1,k_1-1}\beta_{1,k_1}$$
$$R2 : \beta_{2,0} : -\beta_{2,1}, ..., \beta_{2,k_1-1}\beta_{2,k_2}$$

where

$$K_1 = K_2 \quad and \quad \beta_{1,1} = \beta_{2,1}, \beta_{1,2} = \beta_{2,2}, ..., \beta_{1,k_1-1} = \beta_{2,k_2-1}$$

then

$$P(\beta_{1,K_1} \cap \beta_{2,K_2} | \beta_{1,0} \cap (\beta_{1,1}, ..., \beta_{1,K_1-1}) \cap C_i) =$$

$$P(\beta_{1,K_1} | \beta_{1,0} \cap (\beta_{1,1}, ..., \beta_{1,K_1-1}) \cap C_i) \cdot P(\beta_{2,K_2} | \beta_{1,0} \cap (\beta_{1,1}, ..., \beta_{1,K_1-1}) \cap C_i)$$

According to this approach the conditional probability of the structure is computed only once. This approach differs from that proposed in the works of Pompe and Kononenko [PK95] and Flach and Lachiche [FL04] where the factorization would multiply the structure probability twice.

By generalizing to a set of classification rules we have:

$$P(C_i)P(R|C_i) = P(C_i)P(structure)\prod_j P(R_j|structure) \tag{3.5}$$

where the term *structure* takes into account the class C_i and the structures of the rules in R.

If the classification rule $R_j \in R$ is in the form $\beta_{j,0} : -\beta_{j,1}, ..., \beta_{j,K_j-1}, \beta_{j,K_j}$ where $\beta_{j,0}$and $\beta_{j,K_j}$are property predicates and $\beta_{j,1}, \beta_{j,2}, ..., \beta_{j,K_j-1}$ are structural predicates, then:

$$P(R_j|structure) = P(\beta_{j,K_j}|\beta_{j,0}, \beta_{j,1}, ..., \beta_{j,K_j-1}) = P(\beta_{j,K_j}|C_i, \beta_{j,1}, ..., \beta_{j,K_j-1})$$

where C_i is the value of the target attribute in the head of the clause ($\beta_{j,0}$). To compute this probability, we use the Laplace estimation:

$$P(\beta_{j,K_j}|C_i, \beta_{j,1}, ..., \beta_{i,K_j-1}) = \frac{\#(\beta_{j,K_j}, C_i, \beta_{j,1}, ..., \beta_{j,K_j-1}) + 1}{\#(C_i, \beta_{j,1}, ..., \beta_{j,K_j-1}) + F}$$

where F is the number of possible values of the attribute to which the β_{j,K_j}property predicate is associated. Laplace's estimate is used in order to avoid null probabilities in the equation 3.5. In practice, the value at the nominator is the number of individuals which satisfy that conjunction $\beta_{j,K_j}, C_i, \beta_{j,1}, ..., \beta_{j,K_j-1}$, in other words,

the number of individuals covered by the rule $\beta_{j,0} : -\beta_{j,1}, ..., \beta_{j,K_j-1}, \beta_{j,K_j}$. It is determined by a "select count (*)" SQL instruction. The value of the denominator is the number of individuals covered by the rule $\beta_{j,0} : -\beta_{j,1}, ..., \beta_{j,K_j-1}$.

The term *P(structure)* in the equation 3.5 is computed as follows:
Let B=$\{(\beta_{j,1},\beta_{j,2},\ldots,\beta_{j,t})|\ j$=1..$s$ and t=1, ...,$K_j - 1\}$ the set of all distinct sequences of structural predicates in the rules of R. Then

$$P(structure) = \prod_{seq \in \mathrm{B}} P(seq) \tag{3.6}$$

To compute *P(seq)* it is necessary to introduce the definition of the probability JP that a join query is satisfied [Get01a]. Let ϑ=(T_{i_1}, T_{i_2},..., T_{i_s}) be a *Foreign Key Path*, then:

$$\mathrm{JP}(\vartheta) = \mathrm{JP}(\mathrm{T}_{i_1}, ..., \mathrm{T}_{i_s}) = \frac{|\triangleright\triangleleft(\mathrm{T}_{i_1} \times ... \times \mathrm{T}_{i_s})|}{|\ \mathrm{T}_{i_1}| \times ... \times |\mathrm{T}_{i_s}|}$$

where $\triangleright\triangleleft(\mathrm{T}_{i_1} \times ... \times \mathrm{T}_{i_s})$ is the result of the join between the tables $\mathrm{T}_{i_1}, ..., \mathrm{T}_{i_s}$.

We must remember that each sequence *seq* is associated to a foreign key path ϑ. If *seq*=($\beta_{j,1}$,$\beta_{j,2}$, ...,$\beta_{j,t}$) there are two possibilities: either a prefix of *seq* is in B or not. By denoting as T_{j_h} the table related to $\beta_{j,h}$, h=1,..., t, the probability P(seq) can be recursively defined as follows:

$$P(seq) = \begin{cases} \mathrm{JP}(\mathrm{T}_{j_1}, ..., \mathrm{T}_{j_t}) & \text{if } seq \text{ has no prefix in B} \\ \frac{\mathrm{JP}(\mathrm{T}_{j_1}, ..., \mathrm{T}_{j_t})}{P(seq')} & \text{if } seq' \text{ is the longest prefix of } seq \text{ in B} \end{cases}$$

This formulation is necessary in order to compute the formula 3.6 considering both dependent and independent events. Since *P(structure)* takes into account the class, *P(seq)* is computed separately for each class.

3.3.5 Learning Complexity

In order to evaluate the time complexity of the proposed algorithm, we first define some useful variables. Let

- k be the number of tables that are related to another by means of a foreign key.
- h be the number of attributes per table.
- n be the number of tuples in a table
- q be the number of different values per attribute

For simplicity, in this analysis we suppose that k, h, n and q are constant and do not depend on the table. We are aware that this is a strong assumption, but in the worst case analysis we can take the values of that variables such that the resulting cost complexity function is an upper bound of the real one (e.g. we can assume

that k is the maximum number of tables that are related to another by means of a foreign key).

In the computation of $P(structure)$, if we consider the worst case, that is, when the all intermediate tables have no attributes and structural intermediate probabilities have not been already computed before, we have that the time complexity is:

- 0 for the target table
- $k \times join\_complexity$ for the tables at distance 1 from the target table.
- $k^2 \times join\_complexity$ for the tables at distance 2 from the target table.
- ...
- $k^i \times join\_complexity$ for the tables at distance i from the target table.

In the worst case, a join among p tables, is computed in time n^p, thus the time complexity is:

$$\sum_{i=1}^{MAX\_LEN\_PATH} k^i \cdot n^{i+1}$$

this means that the complexity of computing $P(structure)$ is

$$O(k^{MAX\_LEN\_PATH} \cdot n^{MAX\_LEN\_PATH+1}) \tag{3.7}$$

In the computation of $\prod_j P(R_j|structure)$ we have:

- $h \cdot q$ for the target table
- $h \cdot q \cdot k$ for the for the tables at distance 1 from the target table.
- $h \cdot q \cdot k^2$ for the for the tables at distance 2 from the target table.
- ...
- $h \cdot q \cdot k^i$ for the for the tables at distance i from the target table.

The complexity of the computation of $\prod_j P(R_j|structure)$ is:

$$O(h \cdot q \cdot k^{MAX\_LEN\_PATH}) \tag{3.8}$$

By summarizing and combining the two components, we have that the complexity is:

$$O(h \cdot q \cdot k^{MAX\_LEN\_PATH} \cdot n^{MAX\_LEN\_PATH+1}) \tag{3.9}$$

This means that the complexity strongly depends on the MAX_LEN_PATH constant. When the maximum number of tables involved in a foreign key path is

three, the complexity is $O(h \cdot q \cdot k^2 \cdot n^3)$, that is, quadratic in the number of foreign keys between tables and cubic in the number of tuples composing each table.

However, this is the most pessimistic case and, in general, the complexity mainly derives from queries similar to:

```
select count(*) from table1, table2 where table1.id=table2.id1 and...
```

where in the join, at least a primary key is involved. Such queries are efficiently managed by DBMSs that automatically create indexes based on hash tables. Therefore, the use of a DBMS strongly increases efficiency of the learning algorithm.

In the next section we propose to substitute the breadth-first strategy with a search biased by association rules in order to strongly reduce the beam of the search.

3.4 Associative Classification in Multi-relational Data Mining

Another form of structural learning is represented by the possibility to take into account both the structure in the attribute domains and the structure implicitly defined by relations between units of analysis and units of observation.

This problem is particularly salient in a particular branch of data mining, namely Spatial Data Mining. In Spatial Data Mining, training data consists of multiple target spatial objects (units of analysis), possibly spatially-related with other non-target spatial objects (units of observation). The goal is to learn the concept associated with each class on the basis of the interaction of two or more spatially-referenced objects or space-dependent attributes, according to a particular spacing or set of arrangements [Kop99a].

Indeed, mining classification models in spatial data mining presents two main sources of complexity, that is, the implicit definition of spatial relations and the granularity of the spatial objects. The former is due to the fact that the geometrical representation (e.g. point, line, and region in a 2D context) and the relative positioning of spatial objects with respect to some reference system, define implicitly spatial relations of different nature, such as directional or topological. Modeling these spatial relations is a key challenge in classification problems that arise in spatial domains [SSV+02]. Indeed, both the attribute values of the object to be classified and the attribute values of spatially related objects may be relevant for assigning an object to a class from a given set of classes. The second source of complexity refers to the fact that spatial objects can be described at multiple levels of granularity. For instance, UK census data can be geo-referenced with respect to the hierarchy of areal objects:

$$ED\ (enumeration\ district) \rightarrow Ward \rightarrow District \rightarrow Country$$

based on the inside relationship between locations. Therefore, some kind of taxonomic knowledge of task-relevant geographic layers may also be taken into account

to obtain descriptions at different granularity levels (multiple-level classification).

In this session we propose a classification method [CAM04] based on a multi-relational approach that takes spatial relations into account. It can perform the classification at different levels of granularity and takes advantage from domain specific knowledge expressed in form of rules to support qualitative spatial reasoning. In this way, the proposed method can deal with both sources of complexity presented above.

Differently from the approach proposed in section 2.4, here we do not only consider the hierarchical structure in the domain of categories of the units of analysis, but also the hierarchical nature of domains in units of observation. Furthermore, differently from Mr-SBC (see section 3.3), where classification rules are extracted by means of a breadth first strategy, in this section we propose to generate classification rules by means of a spatial association rule discovery system characterized by the capability of generating association rules at multiple levels of granularity. As in Mr-SBC, classification is based on the extension of the naive Bayesian classifier to multi-relational data.

3.4.1 Associative Classification for Spatial Data Mining

The problem of classifying spatial objects has been investigated by some researchers. Ester et al. [EKJ97] proposed a neighbourhood graph based extension of decision trees that considers both non-spatial attributes of the classified objects and relations with neighbouring objects. However, the proposed method does not take into account hierarchical relations defined on spatial objects as well as non-spatial attributes (e.g. number of residents) of neighbouring objects. In contrast, Kopersky [Kop99b] described an efficient method that classifies spatial objects by considering both spatial and hierarchical relations between spatial objects and takes into account non-spatial attributes for neighbouring objects. However this method suffers from severe limitations due to the restrictive representation formalism known as single-table assumption (see section 3.1). More specifically, it is assumed that data to be mined are represented in a single table of a relational database, such that each row (or tuple) represents an independent unit of the sample population and columns correspond to properties of units. This requires that non-spatial properties of neighboring objects be represented in aggregated form causing a consequent loss of information and a change in the units of analysis.

In [MEL+03], the authors proposed to exploit the expressive power of predicate logic to represent both spatial relations and background knowledge, such as spatial hierarchies. In addition the logical notions of generality order and of downward refinement operator on the space of patterns may be profitably used to define both the search space and the search strategy. For this purpose, the ILP system ATRE [Mal03] has been integrated in the data mining server of a prototypical Geographical Information System (GIS), named INGENS, which allows, among other things, to mine classification rules for geographical objects stored in an object-oriented database. Training is based on a set of examples and counterexamples of geographic concepts of interest to the user (e.g., ravine or steep slopes). The first-order logic

representation of the training examples is automatically extracted from maps, although it is still controlled by the user who can select a suitable level of abstraction and/or aggregation of data by means of a data mining query language [MAC03].

Similarly, the discovery of spatial association rules, that is, spatial and a-spatial relationships among spatial objects, has been investigated both in propositional and multi-relational setting. A spatial association rule is a rule of the form $P \rightarrow Q$ (s, c) such that both P (body) and Q (head) are sets of literals, some of which refer to spatial properties, and $P \cap Q = \phi$. $P \cup Q$ is named pattern. The support s estimates the probability $p(P \cup Q)$, while the confidence c estimates the probability $p(Q|P)$.

Koperski and Han [KH95] implemented the module Geo-associator of the spatial data mining system GeoMiner that mines rules from data represented in a single relation (table) of a relational database. In contrast, in [LM04], the authors proposed an ILP approach to spatial association rules discovery. The algorithm SPADA (Spatial Pattern Discovery Algorithm) reported in their work, allows the extraction of multi-level spatial association rules, that is, association rules involving spatial objects at different granularity levels. SPADA has been implemented as a module of the system ARES (Association Rules Extractor from Spatial data) [ACL+03], which also supports users in the complex processes of extracting spatial objects from the spatial database, specifying the background knowledge on the application domain and defining a search bias.

Despite the fact that spatial association rule mining is a descriptive task, while classification of spatial objects is a predictive task, recent studies in Data Mining and Machine Learning have investigated the opportunity of combining association rules discovery and classification, by taking advantage of employing association rules for classification purpose [DZWL99b] [BG03b]. This approach is named associative classification [LHM98] and several advantages are reported in the literature for this approach. First, differently from most of classifiers as decision trees, association rules consider the simultaneous correspondence of values of different attributes, hence allowing to achieve better accuracy [BG03b]. Second, it makes association rule mining techniques applicable to classification tasks. Third, the user can decide to mine both association rules and a classification model in the same data mining process [LHM98]. Fourth, the associative classification approach helps to solve understandability problems [CM93b] [PMS97b] that may occur with some classification methods. Indeed, many rules produced by standard classification systems are difficult to understand because these systems often use only domain independent biases and heuristics, which may not fulfil user's expectation. With the associative classification approach, the problem of finding understandable rules is reduced to a post-processing task [LHM98]; filtering based on user-defined rule template may help in extracting understandable rules.

Although associative classification methods present several interesting aspects, they also suffer from some limitations. First, most of methods reported in the literature work under the single-table assumption, which is a strong limitation in those application domains characterized by a spatial dimension. Second, they have a categorical output which convey no information on the potential uncertainty in classification. Small changes in the attribute values of an object being classified

may result in sudden and inappropriate changes to the assigned class. Missing or imprecise information may prevent a new object from being classified at all. In alternative, to overcome these deficiencies, we propose to use a statistical classifier that returns, in addition to the result of the classification, the confidence of the classification. This is an important aspect because of the increasing attention on the ROC curve analysis [FF03] that defines an evaluation measure to take into account the confidence of the classification. Third, reported methods require additional heuristics to identify the most effective rule at classifying a new object. Alternatively, in the proposed approach, the evaluation of the class is based on the computation of probabilities taking into account all the rules.

3.4.2 Multi-level spatial association rules

In [ACL+03] the problem of mining spatial association rules has been formalized as follows:
Given

- a spatial database (SDB),
- a set S of reference objects tagged with a class label $c_j \in C_1, C_2, ..., C_L$,
- some sets R_k, $1 \leq k \leq m$, of task-relevant objects,
- a background knowledge BK including some spatial hierarchies H_k on objects in R_k,
- M granularity levels in the descriptions (1 is the highest while M is the lowest),
- a set of granularity assignments ψ_k which associate each object in H_k with a granularity level,
- a couple of thresholds $minsup[l]$ and $minconf[l]$ for each granularity level,
- a language bias LB that constrains the search space;

Find strong multi-level spatial association rules, that is, association rules involving spatial objects at different granularity levels.

The reference objects are the main subject of the description (units of analysis), that is, the observation units, while the task relevant objects are spatial objects that are relevant for the task in hand and are spatially related to the former (units of oservation). The sets R_k typically correspond to layers of the spatial database, while hierarchies H_k define is-a (i.e., taxonomical) relations of spatial objects in the same layer (e.g. river is-a water body). Objects of each hierarchy are mapped to one or more of the M user-defined description granularity levels in order to deal uniformly with several hierarchies at once. Both frequency of patterns and strength of rules depend on the granularity level l at which patterns/rules describe data. Therefore, a pattern P ($s\%$) at level l is *frequent* if $s \geq minsup[l]$ and all ancestors of P with respect to H_k are frequent at their corresponding levels. An association rule $Q \rightarrow R(s\%, c\%)$ at level l is *strong* if the pattern $Q \cup R$ ($s\%$) is frequent and $c \geq minconf[l]$.

The problem above is solved by the algorithm SPADA [LM04] that operates in three steps for each granularity level:

1. pattern generation;
2. pattern evaluation;
3. rule generation and evaluation.

SPADA takes advantage of statistics computed at granularity level l when computing the supports of patterns at granularity level $l + 1$.

In the system ARES [1], SPADA has been loosely coupled with a spatial database, since data stored in the SDB Oracle Spatial are pre-processed and then represented in a deductive database (DDB). For instance, spatial intersection between two objects X and Y is represented by the extensional predicate *crosses(X,Y)*. In this way, the expressive power of first-order logic in databases is exploited to specify both the background knowledge BK, such as spatial hierarchies and domain specific knowledge, and the language bias LB. Spatial hierarchies allow to face with one of the main issues of spatial data mining, that is, the representation and management of spatial objects at different levels of granularity, while the domain specific knowledge stored as a set of rules in the intensional part of the DDB supports qualitative spatial reasoning. On the other hand, the LB is relevant to allow the user to specify his/her bias for interesting solutions, and then to exploit this bias to improve both the efficiency of the mining process and the quality of the discovered rules. In SPADA, the language bias is expressed as a set of constraint specifications for either patterns or association rules. Pattern constrains allow to specify a literal or a set of literals that should occur one or more times in discovered patterns. During the rule generation phase, patterns that do not satisfy a pattern constraint are filtered out. Similarly, rule constraints are used to specify literals that should occur in the head or body of discovered rules.

In a more recent release of SPADA (3.1) a new rule constraint has been introduced in order to specify the maximum number of literals that should occur in the head of a rule. In this way users may define the head structure of a rule requiring the presence of exactly a specific literal and nothing more. In the case this literal describes the class label, multi-level spatial association rules discovered by ARES may be used for classification purposed.

3.4.3 Multi-level spatial association rules mining

We denote the DDB in hand $D(S)$ to mean that it is obtained by adding the data extracted from SDB, regarding the set of reference objects S, to the previously supplied BK. The ground facts[2] in $D(S)$ can be grouped into distinct subsets: each group, uniquely identified by the corresponding reference object $s \in S$, is called *spatial observation* and denoted $O[s]$. We define the set:

$$R(s) = \{r_i | \quad \exists k : r_i \in R_k \textit{ and a ground fact } \alpha(s, r_i) \textit{ exists in } D(S)\}$$

[1] http://www.di.uniba.it/∼malerba/software/ARES/index.htm

[2] In this work we assume that ground facts concern either taxonomic "is_a" relationships or binary spatial relationships $\alpha(s, r)$ or object properties.

as the set of task-relevant objects spatially related to s. The set $O[s]$ is given by

$$O[s] \;=\; O[s|R(s)] \cup \bigcup\nolimits_{r_i \in R[s]} O[r_i|S],$$

where:

- $O[s|R(s)]$ contains properties of s and spatial relations between s and r_i
- $O[r_i|S]$ contains properties of r_i and spatial relations between r_i and some $s\prime \in S$.

In an extreme case, $O[s]$ can coincide with $D(S)$. This is the case in which s is spatially related to all task-relevant objects. The unique reference object associated to a spatial observation allows us to define the support and the confidence of a spatial association rule (see the definition of spatial association rule below). Note that the notion of spatial observation in SPADA adapts the notion of *interpretation*, which is common to many relational data mining systems [DL01], to the case of spatial databases.

Let A=$\{a_1, a_2, \ldots, a_t\}$ be a set of Datalog atoms whose terms are either variables or constants [CGT89]. Predicate symbols used for A are all those permitted by the user-specified declarative bias, while the constants are only those defined in $D(S)$. The atom denoting the reference objects is called *key atom*. Conjunctions of atoms on A are called *atomsets* [DR97] like the itemsets in classical association rules. In our framework, a language of patterns $L[l]$ at the granularity level l is a set of well-formed atomsets generated on A. Necessary conditions for an atomset P to be in $L[l]$ are the presence of the key atom, the presence of taxonomic "is_a" atoms exclusively at the granularity level l, the linkedness [Hel87], and safety [CGT89]. In particular, the last property guarantees the correct evaluation of patterns when the handling of negation is required. To a pattern P we assign an existentially quantified conjunctive formula $eqc(P)$ obtained by turning P into a Datalog query.

Definition 3.5 *A pattern P covers an observation $O[s]$ if $eqc(P)$ is true in $O[s] \cup BK$.*

Definition 3.6 *Let O be the set of spatial observations in $D(S)$ and O_P denote the subset of O containing the spatial observations covered by the pattern P. The support of P is defined as $\sigma(P) = |O_P \mid / |O|$.*

Definition 3.7 *A spatial association rule in $D(S)$ at the granularity level l is an implication of the form*

$$P \rightarrow Q \;(s\%,\, c\%)$$

where $P \cup Q \in L[l]$, $P \cap Q = \emptyset$, P includes the key atom and at least one spatial relationship is in $P \cup Q$. The percentages $s\%$ and $c\%$ are respectively called the *support* and the *confidence* of the rule, meaning that $s\%$ of spatial observations in $D(S)$ is covered by $P \cup Q$ and $c\%$ of spatial observations in $D(S)$ that is covered by P is also covered by $P \cup Q$. The support and the confidence of a spatial association rule $P \rightarrow Q$ are given by $s = \sigma(P \cup Q)$ and $c = \varphi(Q|P) = \sigma(P \cup Q) \;/\; \sigma(P)$.

In multi-level association rule mining, an *ancestor* relation between two patterns at different granularity levels $P \in L[l]$ and $P' \in L[l']$, $l < l'$, exists if and only if P' can be obtained from P by replacing each spatial object $h \in H_k$ at granularity level $l = \psi_k(h)$ with a spatial object $h' < h$ in H_k, which is associated with the granularity level $l' = \psi_k(h')$.

The frequency of a pattern depends on the granularity level of task-relevant spatial objects.

Definition 3.8 *Let minsup[l] and minconf[l] be two thresholds setting the minimum support and the minimum confidence respectively at granularity level l. A pattern P is large (or frequent) at level l if $\sigma(P) \geq$ minsup[l] and all ancestors of P with respect to the hierarchies H_k are large at their corresponding levels. The confidence of a spatial association rule $P \rightarrow Q$ is high at level l if $\varphi(Q|P) \geq$ minconf[l]. A spatial association rule $P \rightarrow Q$ is strong at level l if $P \cup Q$ is large and the confidence is high at level l.*

The definition of the strong spatial association rule given above suggests that the generation of association rules at different granularity levels should proceed from the most general towards the most specific granularity levels. This is the approach followed in the ILP system SPADA, which has been developed for mining multi-level association rules in spatial databases. In the following subsection we explain how SPADA performs its search in the space of patterns at a given granularity level l, that is, in the space of patterns defined by the language $L[l]$ (*intra-level search*). In the subsequent subsection we illustrates how SPADA takes advantage of statistics computed at a level l when it searches in the 'more specific' space at level $l+1$ *(inter-level search)*.

Intra-level search of the pattern space

Given a granularity level l and a pattern language $L[l]$, the task of mining spatial association rules can be split into two sub-subtasks:

1. Find *large* (or *frequent*) spatial patterns in the space defined by $L[l]$;
2. Generate highly-confident spatial association rules at level l.

Algorithm design for frequent pattern discovery (step 1) has turned out to be a popular topic in data mining. The blueprint for most algorithms proposed in the literature is the levelwise method [MT97], which is based on a breadth-first search in the lattice spanned by a generality order $\geq$ between patterns. Given two patterns P_1 and P_2, we write $P_1 \geq P_2$ to denote that P_1 is more general than P_2 or equivalently that P_2 is more specific than P_1. The space is searched one level at a time, starting from the most general patterns and iterating between the candidate generation and candidate evaluation phases. The intra-level search algorithm of SPADA implements the afore-mentioned levelwise method (see Algorithm 3.1).

Algorithm 3.1 *Intra-level search implemented in SPADA*
Find large 1-atomsets at level l

Cycle on the depth (k > 1) of search in the pattern space
1. *Generate candidate k-atomsets at level l from large (k-1)-atomsets by applying the refinement operator ρ*
2. *Prune candidates that θ-subsume infrequent patterns*
3. *Prune candidates equivalent under θ-subsumption*
4. *Evaluate candidates and generate large k-atomsets at level l from candidate k-atomsets*

Until the user-defined maximum depth

The pattern space is structured according to the θ-subsumption [Plo70]. Many ILP systems adopt θ-subsumption as the generality order for clause spaces. In this context we need to adapt the framework to the case of atomsets. More precisely, the restriction of θ-subsumption to *Datalog queries* (i.e. existentially quantified conjunctions of Datalog atoms) is of particular interest.

Definition 3.9 *Let Q_1 and Q_2 be two queries. Then Q_1 θ-subsumes Q_2 if and only if there exists a substitution θ such that $Q_2\theta \subseteq Q_1$.*

We can now introduce the generality order adopted in SPADA.

Definition 3.10 *Let P_1 and P_2 be two patterns. Then P_1 is more general than P_2 under θ -subsumption, denoted as $P_1 \geq_\theta P_2$, if and only if P_2 θ-subsumes P_1.*

$\theta-$subsumption is a quasi-ordering, since it satisfies the reflexivity and transitivity property but not the anti-symmetric property. The quasi-ordered set spanned by $\geq_\theta$ can be searched by a *refinement operator*, namely a function that computes a set of refinements of a pattern.

Definition 3.11 *Let $\langle G, \geq_\theta \rangle$ be a pattern space ordered according to $\geq_\theta$. A downward refinement operator under θ-subsumption is a function ρ such that $\rho(P) \subseteq \{Q \mid P \geq_\theta Q\}$.*

In SPADA, the following operator ρ' is used.

Definition 3.12 *Let P be a pattern in $L[l]$. Then $\rho'(P) = \{ P \wedge a_i \mid a_i$ is an atom in $L[l]$ $\}$.*

It can be easily proven that $\rho'(P)$ is a downward refinement operator under θ-subsumption, that is $P \geq_\theta Q$ for all $Q \in \rho'(P)$. Indeed, $Q = P \wedge a_i$ for an atom a_i in $L[l]$. By adopting the set notation we can also write Q= $P \cup \{a_i\}$. The inequality $P \geq_\theta P \cup \{a_i\}$ holds if $P \cup \{a_i\}$ θ-subsumes P, that is, a substitution θ exists such that $P\theta \subseteq P \cup \{a_i\}$. Obviously, θ is the empty substitution. The refinement operator $\rho'(P)$ allows the generation of k-atomsets, that is atomsets of k literals, from (k-1)-atomsets.

It is noteworthy that $\geq_\theta$ on patterns represented as Datalog queries is monotone with respect to support.

Property of θ-subsumption monotony Let $\langle G, \geq_\theta \rangle$ be a pattern space ordered according to $\geq_\theta$. For any two patterns P_1 and P_2 such that $P_1 \geq_\theta P_2$ we have that $\sigma(P_1) \geq \sigma(P_2)$.

Therefore, the refinement operator ρ drives the search towards patterns with decreasing support. If a pattern P is infrequent, all its refinements in $\rho'(P)$ are also infrequent. This is the first-order counterpart of one of the properties holding in the family of the Apriori-like algorithms [AS94], on which the pruning criterion is based. Indeed, the generation of patterns obtained as refinements of infrequent patterns can be avoided, since those patterns have certainly a support lower than the user-defined threshold. This is what happens at step 1) in the algorithm 3.1.

Given a frequent pattern P of k-1 atoms, it may happen that some pattern Q$\in \rho'(P)$ θ-subsumes another infrequent pattern P' of k' atoms, with $k'<k$. This means that Q is certainly infrequent because of the above monotony property, and its evaluation can be avoided (step 2 in the algorithm 3.1). Additional candidates not worth being evaluated are those equivalent under θ-subsumption to some other candidate (step 3 in the algorithm 3.1).

Finally, unpruned candidates are evaluated to check whether they are large (i.e., frequent) or not (*candidate evaluation* phase, step 4). The evaluation of each generated pattern P requires a θ-subsumption test against some spatial observations $O[s]$. Indeed, if $O[s] \cup BK$ θ-subsumes P, then $eqc(P)$ is true in $O[s] \cup BK$, that is P covers $O[s]$, according to the definition given in the previous section. Actually, in SPADA the test of a pattern Q$\in \rho'(P)$ is performed only against those spatial observations covered by P, since, if a spatial observation $O[s]$ is not covered by P, it cannot be covered by Q without violating the transitive property of θ-subsumption.

Inter-level search of the pattern space

As specified in Section 3.4.3, to be able to define a pattern P as *large* (or *frequent*) at level l two conditions must be satisfied, namely

i) $\sigma(\mathrm{P}) \geq minsup[l]$ and

ii) all ancestors of P with respect to the hierarchies H_k are large at their corresponding levels.

The second condition suggests an additional pruning strategy. Let P and Q be two frequent patterns at levels l and $l+1$ respectively, such that P is an ancestor of Q. Suppose that P has been refined into the infrequent pattern P' while searching in the pattern space at level l. When the space of patterns at level $l+1$ is explored and Q is refined, it is possible to generate a candidate pattern Q' whose ancestor is P'. In this case, Q' can be safely pruned, since it cannot be a large pattern without violating condition *ii*). In order to support this additional pruning strategy, the refinement operator implemented in SPADA uses a graph of backward pointers to be updated while searching. Backward pointers keep track of both intra-space and inter-space search stages. Fig. 3.3 gives an example of such a graph, where nodes,

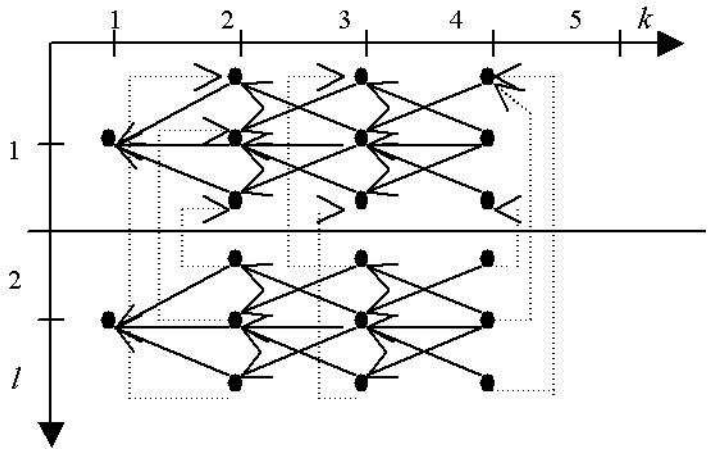

Figure 3.3: Graph of intra-space and inter-space backward pointers.

dotted edges and dashed edges represent patterns, intra-space generality and inter-space parenthood, respectively. The effectiveness of this computational solution is illustrated in [LM02].

From patterns to association rules

Once large patterns have been generated, it is possible to generate strong spatial association rules. For each pattern P, SPADA generates antecedents suitable for rules being derived from P. The consequent corresponding to an antecedent is simply obtained as a complement of atoms in P and not in the antecedent. It is noteworthy that the generation of "good" rule antecedents is crucial. A naïve implementation would consist of a combinatorial computation step followed by a pruning step. The former would output combinations of atoms occurring in P, while the latter would discard those that are not well-formed, e.g. without the key atom in the antecedent or not respecting the constraints of linkedness and safety. Backward pointers can also be exploited to speed up the generation of association rules instead. In particular, SPADA recursively retrieves the predecessors of a frequent pattern and returns only those yielding strong rules. Backward pointers are profitably exploited in the pattern generation phase in order to prevent the generation of some infrequent patterns [LM02]. In a more recent release of SPADA (3.0), backward pointers are also exploited in the pattern evaluation phase. Indeed, by associating each pattern with the list of support objects, it is possible to perform the evaluation of each pattern only on the support objects of its intra-space parent and not on the whole set S of reference objects. An additional caching technique compensates the overhead in looking for the parent of each pattern, since it has a cost which increases with the number of stored patterns.

Filtering patterns and association rules

The efficiency improvements reported above are all based on the monotonicity property of the generality order defined for spatial patterns with respect to the support

of the patterns themselves. This is a nice example of an "intelligent" exploitation of general properties to prune the search space and to reduce the number of expensive tests. However, this uninformative approach does not take into account user preferences and expectations. In real-world applications, such as the characterization of the area crossed by a motorway [MLAS02], a large number of spatial patterns can be generated even for a few hundred spatial objects, most of them proving useless for the application at hand. Therefore, it is important to allow the user to specify his/her bias for some solutions, and then to exploit this bias to improve both the efficiency of the system and the quality of the discovered rules with respect to user's interests. In SPADA, the bias is expressed as a set of constraint specifications for either patterns or association rules. Altogether, they define the language bias (LB) reported in the formulation of the spatial association rule mining problem.

For patterns, the user can specify the constraint *pattern_constraint(AtomList, Min_occur)* where *AtomList* is a list of atoms (for atomic constraints) or a list of atom lists (for conjunctive constraints), while *Min_occur* is a positive number which specifies the minimum number of constraints in the list that must be satisfied. For instance, the following pattern constraint:

pattern_constraint([not_crossed_by_green_area(_,_),
crossed_by_urban_area(_,_)],1).

specifies that at least one of the (spatial) predicates *not_crossed_by_green_area/2* and *crossed_by_urban_area/2* must occur in the patterns filtered by SPADA, while the following pattern constraint:

pattern_constraint([[not_crossed_by_green_area(_,_),
crossed_by_urban_area(_,_)] , [crossed_only_by_road(_)]], 1).

specifies that either the (spatial) predicates *not_crossed_by_green_area/2* and *crossed_by_urban_area/2* or the predicate *crossed_only_by_road/1* must occur in the patterns filtered by SPADA. It is noteworthy that this simple specification allows users to define both conjunctive and disjunctive constraints.

Patterns that do not satisfy a pattern constraint are filtered out during the *rule generation* phase. This means that they are generated and evaluated anyway. This late exploitation of the constraint is due to the fact that if a pattern P does not satisfy a constraint (e.g. because of the lack of the predicate *not_crossed_by_green_area/2*), it is still possible that descendants of P (i.e., more specific patterns) do satisfy it. Therefore, pattern constraints do not prune the pattern space but improve the efficiency of the mining process since they prevent the generation of useless rules, and hence their evaluation.

A further pattern constraint takes into account the typing mechanism of the variables to be included in the rules. A variable X is (un-)typed when it (does not) appear as first argument of a *is-a/2* atom in the rule. In some applications, the occurrence of untyped variables in a rule is undesirable; therefore the user can specify the constraint *max_rules_untyped_vars(n)*, where n denotes the maximum

number of un-typed variables in the rules being generated. As in the previous case the specification of this constraint affects the rule generation phase.

For spatial association rules the user can define constraints either on the antecedent or on the consequent by specifying one of the following facts in LB:

body_ constraint(AtomList, Min_ occur). head_ constraint(AtomList, Min_ occur).

where *AtomList* and *Min_ occur* have the same meaning as in the pattern constraint. For instance, the constraint *head_ constraint ([high_ mortality(_)], 1)* specifies that the predicate *high_ mortality/1* must occur in the head of the rules to be discovered by SPADA. Since association rules discovered by SPADA can have several conditions in the head, additional predicates are also allowed in the head.

As for pattern constraints, head and body constraints do affect the rule generation phase. The main difference is that these constraints do not prevent the generation of candidate rules but only the evaluation of their confidence.

Discretizing Numerical Features

Since SPADA, like many other association rule mining algorithms, cannot process numerical data properly, it is necessary to perform a discretization of numerical features with a relatively large domain. For this purpose we have implemented the relative unsupervised discretization algorithm RUDE [LW00], which discretizes an attribute of a relational database in the context defined by other attributes. Formally, the problem can be stated as follows:

Given

- a database table T consisting of n tuples,
- a continuous attribute in T to be discretized (*target attribute*),
- a set of continuous attributes (*source attributes*) in T that define the context for the discretization of the target attribute,
- a relative tolerance between split points (minimal difference) s

Find a set of split points that minimize loss of correlation between attributes.

The algorithm RUDE is based on two general procedures: a *prediscretization* procedure, used to pre-process the *source attributes*, and a *clustering* procedure, used to group target attribute values corresponding to some source attribute value or interval. Therefore, several different "specializations" of the RUDE algorithm can be generated by varying the two procedures. The implementation of RUDE in ARES supports two prediscretization algorithms, namely equal width and equal frequency and two clustering algorithms, namely *EM* [WF99] and *AutoClass* [CS96]. RUDE proves to be suitable for dealing with numerical data in the context of association rule mining. An experimental study not reported in this thesis showed that the best performance can be obtained by using the equal width prediscretization procedure and the Autoclass algorithm.

3.4.4 Classification using Discovered association rules

Once a set of rules has been extracted for each level, the construction of the naive Bayesian classifier mainly follows the Mr-SBC approach (see section 3.3), which aims to classify any target object $o \in S$ by maximizing the *posterior probability* $P(C_i|o)$ that o is of class C_i, that is:

$$class(o) = arg\ max_i P(C_i|o)$$

By applying the Bayes theorem, $P(C_i|o)$ can be reformulated as follows:

$$P(C_i|o) = \frac{P(C_i)P(o|C_i)}{P(o)} \tag{3.10}$$

The term $P(o|C_i)$ is estimated by means of the *naive Bayes assumption*:

$$P(o|C_i) = P(o_1, o_2, \ldots, o_m|C_i) = P(o_i|C_i) \times P(o_2|C_i) \times \ldots \times P(o_m|C_i)$$

where $o_1, o_2, \ldots, o_m$ represent the set of the properties, different from the class, used to describe the object.

In 3.10 the value $P(C_i)$ is the prior probability of the class C_i. Since $P(o)$is independent of the class C_i, it does not affect $f(o)$, that is,

$$class(o) = arg\ max_i \quad P(C_i)P(o|C_i) \tag{3.11}$$

In order to take into account the relations of the target object, we consider the set of rules to guide the computation of $P(o|C_i)$.

Given the object $o \in S$, we consider the subset of the extracted rules that can be used to classify o. More formally, we consider the subset R of rules whose body is satisfied by the object to be classified both in terms of the values of properties of involved spatial objects and in terms of the spatial relations between objects. For example, if S is the set of wards in a district, a ward w satisfies the rule:

$$mortality\_rate(A,\ low) \leftarrow wards\_relatedTo\_waters(A,\ B),$$
$$waters\_typewater(B,\ river),\ cars\_per\_person(A,\ high)$$

when w is spatially related (intersects) to a river and is characterized by a high average number of cars per person.

We use R to estimate $P(o|C_i)$. In particular, we estimate $P(o|C_i)$by means of the probabilities associated to both spatial relations (e.g. *wards_ relatedTo_ waters(A,B)*) and properties (e.g. *waters_ _ typewater(B,RIVER)*, *cars_ per_ person(A,high)*) associated to each rule in R.

For instance, if $R = \{R_1, R_2\}$, where R_1 and R_2 are two association rules of class C_i extracted by SPADA:

$$R_1\colon\ \beta_{1,0} : -\beta_{1,1}, \beta_{1,2} \quad R_2\colon\ \beta_{2,0} : -\beta_{2,1}, \beta_{2,2}$$

where $\beta_{1,1}$ and $\beta_{2,1}$ are spatial relations,$\beta_{1,2}$and$\beta_{2,2}$ are properties and $\beta_{1,0} = \beta_{2,0}$(class) then $P(\{R_1, R_2\}|C_i) = P(\beta_{1,0} \cap \beta_{1,1} \cap \beta_{2,1} \cap \beta_{1,2} \cap \beta_{2,2}|C_i)=$

$$P(\beta_{1,0} \cap \beta_{1,1} \cap \beta_{2,1}|C_i) \cdot P(\beta_{1,2} \cap \beta_{2,2}|\beta_{1,0} \cap \beta_{1,1} \cap \beta_{2,1} \cap C_i)$$

The first term takes into account the relations of the rules while the second term refers to the conditional probability of satisfying the property predicates in the rules given the relations. According to the the naive Bayes independence assumption, the probabilities can be factorized as follows:

$$P(\beta_{1,0} \cap \beta_{1,1} \cap \beta_{2,1}|C_i) = P(\beta_{1,1}|C_i) \cdot P(\beta_{2,1}|C_i)$$

$$P(\beta_{1,2} \cap \beta_{2,2}|\beta_{1,0} \cap \beta_{1,1} \cap \beta_{2,1} \cap C_i) = P(\beta_{1,2}|\beta_{1,1} \cap \beta_{2,1} \cap C_i) \cdot P(\beta_{2,2}|\beta_{1,1} \cap \beta_{2,1} \cap C_i)$$

Since $\beta_{1,2}$ and $\beta_{2,2}$ do not depend from $\beta_{2,1}$and $\beta_{1,1}$ respectively, then:

$$P(\beta_{1,2} \cap \beta_{2,2}|\beta_{1,0} \cap \beta_{1,1} \cap \beta_{2,1} \cap C_i) = P(\beta_{1,2}|\beta_{1,1} \cap C_i) \cdot P(\beta_{2,2}|\beta_{2,1} \cap C_i)$$

By generalizing to a set of rules we have:

$$P(C_i)P(o|C_i) = P(C_i) \prod_{k \in |R|} (P(relations_k|C_i) \prod_j P(property_{k,j}|relations_k, C_i)) \tag{3.12}$$

where the term $relations_k$ represents the event that the set of spatial relations expressed in the k-th rule is satisfied, while the term $property_{k,j}$ represents the event that the j-th property of the k-th rule is satisfied.

If $relations_k$= { $relation(Set_1,Set_2)|$ $Set_1,Set_2 \in \{S\} \cup \{R_k,\ 1 \leq k \leq m\}$, $Set_1 \neq Set_2$ } is a set of binary relations between spatial objects (either task relevant or reference) involved in the k-th rule, the probability $P(relations_k|C_i)$ is computed by means of the naive Bayes assumption:

$$P(relations_k|C_i) \ = \prod_{l \in |relations_k|} P(relation(Set_{l_1}, Set_{l_2})|C_i)$$

where:

$$P(relation(Set_{l_1}, Set_{l_2})|C_i) = P(relation(Set'_{l_1}, Set'_{l_2})) = \frac{|relation(Set'_{l_1}, Set'_{l_2})|}{|Set'_{l_1}| \cdot |Set'_{l_2}|} \tag{3.13}$$

where Set'_l is a subset of objects in Set_l that are related, by means of spatial relations, with objects in S of class C_i, while $|relation(Set'_{l_1}, Set'_{l_2})|$ is the number of relations between objects of Set'_{l_1} and objects of Set'_{l_2}.

To compute the probability $P(property_{k,j}|relations_k, C_i)$ in (3), we use the Laplace estimation:

$$P(property_{k,j}|relations_k, C_i) = \frac{|relations_k \wedge property_{k,j} \wedge C_i| + 1}{|relations_k \wedge C_i| + F} \quad (3.14)$$

where F is the number of possible admissible values of the property. Laplace's estimate is used in order to avoid null probabilities in equation 3.11. In practice, the value at the nominator is the number of target objects of class C_i that are related to other spatial objects by means of spatial relations expressed in $relations_k$ and for which $property_{k,j}$ is satisfied. The value of the denominator is the number of target objects of class C_i that are related to other spatial objects by means of spatial relations expressed in $relations_k$ plus F.

In order to avoid the problem that the same relation or the same property is considered more than once in the computation of probabilities in formula 3.12, the values computed in formula 3.13 and 3.14 are effectively determined and included in formula 3.12 only if the values have not been computed before.

3.5 Conclusions

In this chapter we considered a different kind of "structure", that is, the structure represented by the occurrence of relations between the units of analysis and/or the units observation. For this purpose, we resort to the multi-relational data mining.

We present two classifiers that work in the multi-relational setting. In particular, we extend the naive Bayes classification to the case of relational data. The first solution is represented by a multi-relational data mining system which is tightly integrated with a relational DBMS. It is based on the induction of a set of first-order classification rules in the context of naive Bayesian classification. It presents several differences with respect to related works. First, it is based on an integrated approach, so that the contribution of literals shared by several rules to the posterior probability is computed only once. Second, it works both on discrete and continuous attributes. Third, the generation of rules is based on the knowledge of a data model embedded in the database schema. The proposed method has been implemented in the new system Mr-SBC.

The second solution is inspired by recent studies on the usage of association rules for classification purposes (Associative Classification). In particular, we have presented a spatial associative classifier that combines spatial association rule discovery with naive Bayesian classification. Domain specific knowledge may be defined as a set of rules that makes possible the qualitative spatial reasoning. In addition, hierarchies on spatial objects are expressed by a collection of ground atoms and are exploited to mine classification models at different granularity levels. For each granularity level, extracted rules concur in building the spatial classification model by exploiting a multi-relational naive Bayesian classifier integrated with the Database.

Chapter 4

Applications of Naive Bayesian Classification to Document Engineering

In this chapter we show the application of proposed solutions to the field of Document Engineering. Document Engineering is the computer science discipline that investigates systems for documents in any form and in all media. Document engineering is concerned with principles, tools and processes that improve our ability to create, manage and maintain documents. It shares many concepts with Software Engineering, which is concerned with the creation, management and maintenance of a special kind of documents, the programs. However it also presents several differences, due to the different semantics of documents, the diverse use of layout and logical structures, the different emphasis given to graphical aspects as well as the different design methods (or writing processes) [VQ90]. It might be debatable that Document Engineering is a true engineering discipline, for the same reasons that some researchers attributed at Software Engineering at the early '90 [Sha90]. Nonetheless, computer-based systems for creating, distributing and analysing documents are one of the centerpieces of the new "Information Society" and it is very likely that the meeting of economic and scientific interests will soon lead to the development of a professional engineering.

The notion of document adopted in Document Engineering is quite extensive. A document is a representation of information designed for reading by, or played-back to, a person. It may be presented on paper, on a screen, or played through a speaker and its underlying representation may be in any form and include data from any medium. A document may be stored in final presentation form or it may be generated on-the-fly, undergoing substantial transformations in the process. A document may include extensive hyperlinks and be part of a large web of information. Furthermore, apparently independent documents may be composed, so that a web of information may itself be considered a document.

Among conceptual topics relevant to the field of Document Engineering are

document structure and content analysis, document categorization and classification, document storage, indexing, and retrieval, performance of document systems, markup languages (e.g., XML), and optical character recognition (OCR). In this chapter we are interested in both printed documents and text documents, and we consider some conceptual topics reported above. More precisely, in the first part of this chapter we show the application of the hierarchical classification framework proposed in chapter 2 to the problem of *text document categorization and classification*. Whereas, in the second part, we show the application of the multi-relational naive Bayesian classifier Mr-SBC to the problem of *document image understanding* (or interpretation), which is defined as the formal representation of the abstract relationships indicated by the two-dimensional arrangement of the symbols [Nag00].

4.1 Hierarchical Text Classification

Text classification or text categorization is the process of automatically assigning one or more predefined categories to text documents. A wide range of supervised learning algorithms has been applied to this problem, using a training set of categorized documents to build a classifier that maps arbitrary documents to relevant categories. Most of learning methods reported in the literature deal with classifying text into a set of categories without structural relationships among them (flat classification). More recently, increasing attention has been given to hierarchical classification [KS97] [MRMN98] [Mla98b] [DMSK00] [DC00] [NGL97] [RS02] [WWP99], where the pre-defined categories are organized in a hierarchical structure (tree-like structure). Such a structure reflects relations between concepts in the application domain covered by the classification. Indeed, as already specified, many popular search engines and text databases arrange documents in topic hierarchies, such as Yahoo, Google Directory, Medical Subject Headings (MeSH) in MEDLINE, Open Directory Project (ODP)[1] and Reuters Corpus Volume I (RCV1) [LYRL04]. This hierarchical arrangement is essential when the number of categories is quite high, since it supports a thematic search by browsing topics of interests.

The advantage of this hierarchical view of the classification process is that the problem is partitioned into smaller subproblems, each of which can be effectively and efficiently managed. Another motivation, strictly related to the problem in hand, is given by the observation that both precision and recall decrease as the number of categories increases [ADW94] [Yan96] due to the increasing effect of term polysemy for large corpora.

As pointed out in chapter 2, taking into account the hierarchy poses additional issues in the development of methods for automated document classification.

- documents can either be associated to the leaves of the hierarchy or to internal nodes.
- the set of features selected to build a classifier can either be category specific or the same for all categories (corpus-based).

[1] www.dmoz.org

- the training set associated to each category may include or not training documents of subcategories.
- the classifier may take into account or not the hierarchical relation between categories.
- some stopping criterion is required for hierarchical classification of new documents in non-leaf categories.
- new performance evaluation criteria are required to take into account the different types of classification errors.

All these issues are systematically investigated in this chapter, which presents the hierarchical classification framework proposed in chapter 2 in the text categorization domain. The hierarchy of categories is used in all phases of text categorization, namely feature extraction, learning, and classification of a new document.

In this chapter we use the naive Bayesian learner and compare it with two of the most widely investigated methods for (flat) text classification, namely centroid-based and support vector machines (SVM), and we investigate the performance of these methods on three datasets (Yahoo, DMOZ, RCV1). These datasets present a variety of situations in terms of hierarchical structure: documents can be assigned to any node in the hierarchy, some nodes can have no associated documents and internal nodes can have only one child. The baseline of the empirical evaluation is the flat classification, so that it is possible to analyse the actual contribution of the hierarchy in text classification performance. Another aspect considered in this framework is the construction of feature sets, which can be performed by merging the dictionaries of all subcategories (hierarchical feature set) or by taking the union of dictionaries of direct subcategories (proper feature set). Pros and cons of hierarchical feature sets are discussed and interactions with learning methods are empirically evaluated.

To test alternative hierarchical text classification methods, the system WebClassIII has been implemented. This is a client-server application that has been designed to support the search activity of a geographically distributed group of people with common interests [MEC02]. It works as an intermediary when users browse the Web through the system and classify documents into a hierarchy of categories by means of one of the classification techniques available. Automated classification of Web pages is performed on the basis of their textual content and may require a preliminary training phase in which document classifiers are built on the basis of a set of training examples.

4.1.1 Document Representation and Feature Selection

In WebClassIII, the feature set is unique for each internal category and is automatically determined by means of a set of positive and negative training examples (extracted from the Hierarchical training set, see section 2.4.1 and Figure 2.8). More specifically, in WebClassIII, all training documents are initially tokenized, and the set of tokens (words) is filtered in order to remove HTML tags, punctuation marks,

numbers and tokens of less than three characters. Only relevant tokens are used in the feature set. Before selecting relevant features, standard text pre-processing methods are used to:

1. Remove stopwords, such as articles, adverbs, prepositions and other frequent words taken from Glimpse[2], a tool used to index files by means of words.

2. Determine equivalent stems (stemming), such as 'topolog' in the words 'topology' and 'topological', by means of Porter's algorithm for English texts [Por97].

Despite these preprocessing steps reduce the number of extracted tokens, the feature set can be still large even in the case of small document collections. In many learning algorithms, reduction of the set of features is essential for both complexity and accuracy issues. In particular, centroid-based methods compute the distance of a document from a centroid on the basis of all features used to describe the documents. If the attribution of a document to a category depends on only a few of the many available features, than the documents that are truly "close" to the centroid may well be a large distance apart. Galavotti et al. [GSS00] and Ruiz and Srinivasan [RS02] have independently proved that the Rocchio classifier, which is a particular centroid-based classifier, benefits from feature selection. Also for naive Bayesian classifiers it has been proved that they benefit of irrelevant feature removal [Mla98a]. The situation is different in the case of SVM classifiers, which work well with high dimensional feature spaces and eliminate the need for feature selection [Joa98]. In this work, where these three different learning methods are considered, feature selection is always performed for the purpose of having a fair comparison. The exploration of the effect of considering all features is postponed for future research.

The problem of feature selection has been widely explored in machine learning. Feature selection approaches may be categorised into wrapper, filter and embedded approaches [JKP94] [BL97b]. The wrapper approach attempts to identify the best feature subset to use with a particular algorithm, that is, the induction algorithm that will be used to learn the final target concept is part of the evaluation function. In the filter approach, the goal is to filter the irrelevant and/or redundant features on the basis of the characteristics of the training data without involving any learning algorithm. Finally, in the embedded approaches the feature selection process is done inside the learning algorithm, preferring some features to others, and possibly not including all the available features in the final model induced by the learning algorithm (a clear example is represented by decision trees). The wrapper approach tends to produce better accuracy than the filtering approach, but this is possible to the disadvantage of the computational complexity. Because of the abundance of features (and documents) in automated text categorization, filtering approach remains the most widely used. Moreover, it is very flexible, since any target learning algorithm can be used, while both the wrapper approach and the embedded approach are strictly dependent on the learning algorithm.

[2] glimpse.cs.arizona.edu

Most of filtering methods for information retrieval simply score words according to some feature selection measure and select the best firsts. However, techniques proposed for information retrieval purposes are not always appropriate for the task of text categorization. Indeed, we are not interested in words characterizing each single document, but we look for words that distinguish a document category from other categories. Generally speaking, the set of words required for classification purposes is much smaller than the set of words required for indexing purposes.

There are two distinct approaches to feature selection for text categorization:

a) local, for each category c_i a set of features is chosen for classification of documents in c_i;

b) global, a set of terms is chosen for classification under all categories [Seb02].

They are related to document representation [ADW94] by means of several specialized feature vectors for different categories or by a unique feature. No special recommendation for local vs. global feature selection is reported in the literature. Typically the approach adopted in a work depends on the type of classifier. Local feature sets are used together with binary classifiers, which decide to assign a document to a category c_i or not, while global feature sets are used together with multi-class classifier, which assign a document to one (single-category classification) or more (multi-category classification) categories in $\{c_1, c_2, \ldots, c_L\}$.

Independently of the approach, several feature selection measures have been reported in the literature. They can be classified on the basis of four dependency tuples between a term w and a category c_i [ZWS04]:

1. (w, c_i): w and c_i co-occurs,
2. $(w, \neg c_i)$: w occurs without c_i;
3. $(\neg w, c_i)$: c_i occurs without w;
4. $(\neg w, \neg c_i)$: neither w nor c_i occur.

The first two tuples concern the presence of a term, while the last two are related to its absence. The first and the last tuples represent the positive dependency between w and c_i, while the other two represent the negative dependency. Although all feature selection measures try to capture the intuition that the best terms for c_i are the ones that distributed most differently in the sets of positive and negative examples of c_i[3], they consider different dependency tuples. For instance, Correlation Coefficient [NGL97] considers all the four tuples, Mutual Information [YP97] considers the first three, while Odds ratio [Mla98b] is based only on the first two. The variety of results reported in the literature does not allow us to make any claim on what should be the dependencies to involve in the definition of a good feature selection measure. As observed by Mladenic and Grobelnik [MG99] "the most important characteristics of a good feature scoring measure for text are: favoring common features and considering domain and algorithm characteristics".

[3] A notable exception is the frequency of a term in a document collection, where only positive examples are considered.

Following this indication, in this work we focus our interest on the global approach, which seems best suited for multi-class classifiers, as well as on the first two tuples, since the classifiers that will be presented in the next sections increase their confidence on classification on the basis of present terms rather than absent terms. In the design of the feature selection measure reported in this work we do take into account another important factor: the observation unit for all classifiers is the document, hence the "common features" Mladenić and Grobelnik refers to, should not only be "frequent for a category" but also shared by most of documents of the same category. A term that occurs frequently in very few documents of a category can be frequent for the category but can hardly be considered a common feature. Surprisingly, a closer look at the feature selection measures reported in the literature reveals that most of them consider a term (and not a document) as observation unit. By looking at formulas of the most widely investigated feature selection measure reported in [MG99] at Table 1, we find that the ingredients of various formulas are:

1. $P(w)$, the prior probability that the term w occurs
2. $P(c_i)$, the prior probability of the i-th class or category
3. $P(c_i|w)$, the conditional probability of the i-th class value given that w occurs
4. $P(w|c_i)$, the conditional probability of w given the i-th class value
5. $TF(w)$, the term frequency.

None of them do actually refer to the document as observation unit. For instance, the absolute frequency of a term in a document, $TF(w, d)$, which is used in the naive Bayes classifier (see Section 4.1.2), is not considered. In the centroid-based classification, where it is important to select a set of features that increase the intra-class document similarity and decrease the inter-class document similarity, the distribution of a term across training documents of the same category is important, but it does not appear in the list above.

For multi-class problems, as those considered in the framework proposed in this chapter, Malerba et al. [MEC02] developed a feature selection procedure that do take into account these observations. In this work, we develop an extension to the case of hierarchical training sets.

Let c be a category and c' one of its children in the hierarchy of categories, that is, $c' \in DirectSubCategories(c)$. Let d be a training document (after the tokenizing, filtering and stemming steps) from c', w a feature extracted from d and $TF_d(w)$ the relative frequency of w in d. Then, the following statistics can be computed:

- the maximum value of $TF_d(w)$ on all training documents d of category c',

$$TF_{c'}(w) = max_{d \in Training(c')} TF_d(w)$$

- the document frequency, that is, the percentage of documents of category c' in which the feature w occurs,

$$DF_{c'}(w) = \frac{|\{d \in Training(c') | \quad w \; occurs \; in \; d\}|}{|Training(c')|}$$

- the category frequency $CF_c(w)$, that is, the number of subcategories $c'' \in DirectSubCategories(c)$ such that w occurs in a document $d \in Training(c'')$.

We observe that only documents considered as positive examples of c' are used to compute both $TF_{c'}(w)$ and $DF_{c'}(w)$, while the estimation of $CF_c(w)$ also takes into account documents considered as negative examples of c'.

For each category c', a list of pairs $\langle w_i, v_i \rangle$ is computed, such that w_i is a term extracted from some document $d \in Training(c')$ and

$$v_i = TF_{c'}(w_i) \times DF^2_{c'}(w_i) \times \frac{1}{CF_c(w_i)}$$

By taking words that maximize the product $maxTF \times DF^2 \times ICF$, where ICF stands for "inverse CF", we reward common words used in documents of category c', but we penalize words common to both c' and its sibling categories. The category dictionary of c', $Dict_{c'}$, is the set of the best n_{dict} terms with respect to v_i, where n_{dict} is a user defined parameter.

The measure $maxTF \times DF^2 \times ICF$ scores high features that appear (possibly frequently) in many relevant documents and in documents of few alternative categories. In contrast with correlation coefficient, it does not suffer from problems of unreliability for low frequency terms, so we are not forced to remove rare features as done by Ruiz and Srinivasan [RS02] in their study on hierarchical text categorization. Moreover, it is not influenced by the marginal probability of terms as in the case of mutual information [YP97], which makes score incomparable across terms of widely differing frequency.

The feature set associated to a category c is defined on the basis of the dictionaries of its subcategories[4]. More precisely, the *proper feature set* $FeatSet_c$ is defined as the union of the dictionaries of all direct subcategories of c (see Figure 4.1):

$$FeatSet_c = \bigcup_{c' \in DirectSubCategories(c)} Dict_{c'}$$

It contains features that appear frequently in many documents of one of the subcategories but seldom occur in documents of the other subcategories (orthogonality of category features). In other terms, selected features decrease the intra-category dissimilarity and increase the inter-category dissimilarity. Therefore, they are useful to classify a document (temporarily) assigned to c as belonging to a subcategory of c itself. It is noteworthy that this approach returns a set of quite general features (like "math" and "mathemat") for upper level categories, and a set of specific features (like "topolog") for lower level categories.

An alternative proposal is the hierarchical feature set, which is defined as the union of the dictionaries of all subcategories (similarly to Mladenic [Mla98b]) where, in addition, weights are used to give less importance to subcategories that are further down in the hierarchy):

[4]McCallum et al. [MRMN98] use the term hierarchical feature selection to denote the selection of an equal number of features at each internal node of the tree, using the node's immediate children as the classes.

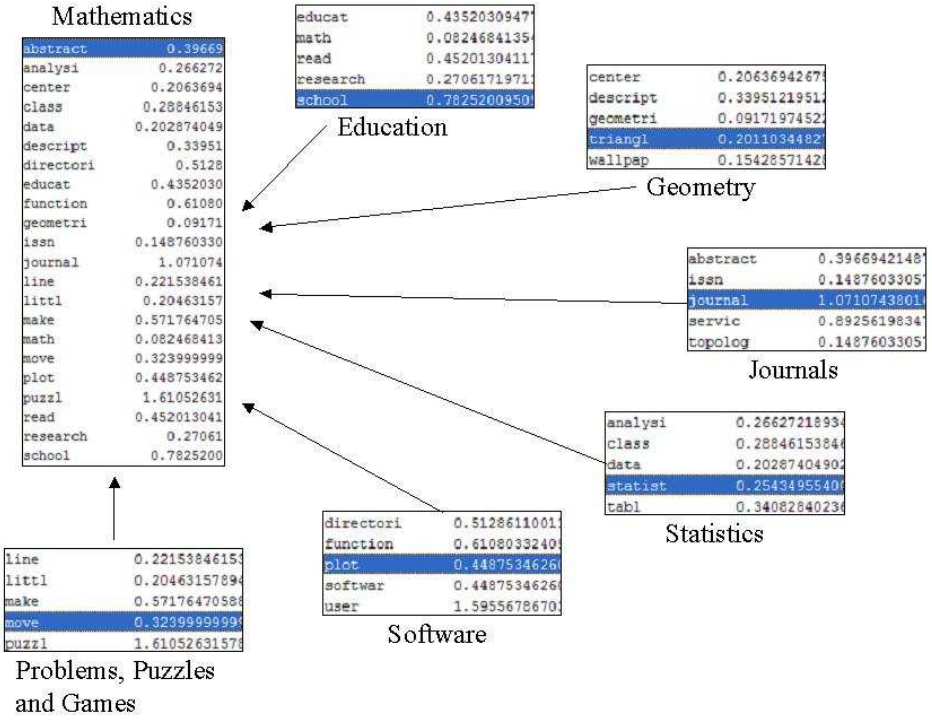

Figure 4.1: Category dictionaries extracted by WebClassIII for all subcategories of "Mathematics" in an experiment on Yahoo dataset ($n_{dict} = 5$) and proper feature set selected for "Mathematics".

$$HierFeatSet_c = \bigcup_{c' \in SubCategories(c)} Dict_{c'}$$

The rationale behind the hierarchical feature set is that if classifiers at the top level do take into account only general terms (such as "math" and "mathemat") typically extracted from documents of general topics (e.g., Mathematics), they might have some difficulties to correctly route along the right path those documents belonging to leaf categories (e.g., Geometry), because of the rarer occurrence of general terms. Once the set of features has been determined for an internal category c, training documents in $Training(c)$ can be represented as feature vectors, where each feature value is the frequency of a word.

4.1.2 Learning algorithms

In the context of the hierarchical text categorization framework described in section 2.4.1, the definition of the same feature set to represent documents of a category c and all its subcategories permits the application of a multi-class learning algorithm to induce a classifier that categorizes a document (temporarily) assigned to c as belonging to a subcategory c' of c. In this work we consider the naive Bayes learning approach [Mit97] modified in order to correctly handle documents of different length. We compare this approach with other two learning approaches:

- a centroid-based method [HK00], where each centroid (or class prototype) is the center of cluster of documents of the same category;
- SMO, which is an optimized algorithm for training SVM on very large data sets [Pla99].

Therefore, the classification of a new document to a category c' is obtained as follows:

1. By estimating the Bayesian posterior probability for that category (naive Bayes).
2. By computing the similarity between the document and the centroid of that category.
3. By estimating the posterior probability for that category according to an SVM probabilistic classifier.

The three learning algorithms are briefly described in the next subsections.

Naive Bayesian classifier

Let d be a document temporarily assigned to a category c. We intend to classify d into one of the subcategories of c. The Bayes optimal classification can be achieved by assigning d to the category $c_i \in DirectSubCategories(c)$ that maximizes the posterior probability $P_c(c_i|d)$.

In the literature, several Bayesian models have been proposed for text categorization. The naive Bayes classifier is the simplest of these models, in that it assumes that all the features used to describe the document are independent of each other given the context of the class (class conditional feature independence). We discussed this assumption in section 2.2.3 and, citing some seminal works, we deduced that, even in the case that the independence assumption is violated by a wide margin and the approximation of conditional probability is poor, the classification accuracy remains high [DP97].

In the text categorization literature, two different models based on the naive Bayes assumption have been proposed: the *multivariate Bernoulli model* and the *multinomial model* [MN98]. The former specifies that a document be represented by a vector of binary attributes indicating which terms occur and do not occur in the document. The "event" is the document, and both the presence and the absence of a term contribute to the estimation of the posterior probability, which is modelled as multivariate Bernoulli. In the context of hierarchical text categorization it has been used by Koller and Sahami [KS97]. The multinomial model specifies that a document be represented by the set of term occurrences in the document. In this case the "event" is the term and the number of occurrences of each term affects the posterior probability, which is based on a multinomial model. In hierarchical text categorization this model has been used by Mladenic [Mla98b]. A review of naive Bayes classifiers and their usage in information retrieval is reported in [Lew98], where the Bernoulli model is named *binary independence model*.

McCallum and Nigam [MN98] have shown that, over a number of different text categorization problems, the multinomial model is capable of categorizing documents more accurately than the multivariate Bernoulli model. Eyheramendy and his collegues [ELM03] have considered three alternatives to the multinomial model that still incorporate term frequencies, and have empirically shown that the multinomial model often outperforms these alternatives. Therefore, in this work we consider the naive Bayesian classifier based on multinomial model. This choice is also coherent with the feature selection process where only the presence (and not the absence) of a feature is considered, and the number of occurrences of a term is an important factor in feature selection.

In its general formalization, the multinomial model accommodates very naturally the document length. The posterior probability $P_c(c_i|d)$ can be defined as the sum over posterior probabilities of documents of different length [Joa97]:

$$P_c(c_i|d) = \sum_{l=1}^{\infty} P_c(c_i|d,l)P_c(l|d) \tag{4.1}$$

where $P_c(l|d) = 1$ for the length l_d of document d and is zero otherwise. In other terms, $P_c(c_i|d) = P_c(c_i|d, l_d)$. By applying Bayes' theorem to $P_c(c_i|d)$ we have:

$$P_c(c_i|d) = \frac{P_c(d|c_i,l_d)P_c(c_i|l_d)}{\sum_{c_j \in DirectSubCategories(c)} P_c(d|c_j,l_d)P_c(c_j|l_d)} \tag{4.2}$$

$P_c(c_i|l_d)$ is the prior probability that a document of length l_d is in class c_i. By assuming that *the category of a document does not depend on its length*, we can write $P_c(c_i|l_d) = P_c(c_i)$. The prior probability $P_c(c_i)$ is estimated as the fraction of training documents of c assigned to class c_i:

$$p_c(c_i) = \frac{|Training(c_i)|}{\sum_{c' \in DirectSubCategories(c)} |Training(c')|} \tag{4.3}$$

The estimation of the likelihood $P_c(d|c_i, l_d)$ is based on the multinomial model:

$$P_c(d|c_i,l_d) = \frac{l_d!}{\prod_{w \in FeatSet_c} TF(w,d)!} \prod_{w \in FeatSet} P_c(w|c_i,l_d) \tag{4.4}$$

where $TF(w, d)$ denotes the absolute frequency of w in d.

The first term depends only on the document d and multiplies both the numerator and the denominator of formula 4.2, hence it can be dropped. The subsequent terms are the probabilities of observing a term w of the feature set in documents of length l_d and of class c_i. Unfortunately, the estimation of this conditional probability is quite difficult, since we should consider only documents of length l_d in the training set. Therefore, a further simplifying assumption is usually made, that the occurrence of a term is only dependent on the membership class of a document [Joa97]. By combining this assumption with the original feature independence assumption we have:

$$P_c(d|c_i, l_d) \propto \prod_{w \in FeatSet} P_c(w|c_i)^{TF(w,d)} \tag{4.5}$$

In conclusion, under the assumptions that each term in d occurs independently of other terms, as well as independently of the text length, it is possible to estimate the posterior probability as follows:

$$P_c(c_i|d) = \frac{P_c(c_i) \prod_{w \in FeatSet} P_c(w|c_i)^{TF(w,d)}}{\sum_{c' \in DirectSubCategories(c)} P_c(c') \prod_{w \in FeatSet} P_c(w|c')^{TF(w,d)}} \tag{4.6}$$

To make our probability estimate of $P_c(w|c_i)$ more robust with respect to infrequently used terms, we use a smoothing method to modify the estimates that would have been obtained by simple event counting. Smoothing, whose main effect is that of assigning a small, non-null probability to unobserved events, is important in naïve Bayes classifiers, since probability estimates are multiplied. If only one of them were zero at numerator, the posterior probability in 4.6 would be zero, independently of the values of the other estimates. In this work smoothing is based on Laplace's law of succession, that is:

$$P_c(w|c_i) = \frac{1 + PF(w, c_i)}{|FeatSet_c| + \sum_{w' \in FeatSet_c} PF(w', c_i)} \tag{4.7}$$

where $PF(w, c)$ denotes the absolute frequency w in documents of category c. An alternative to Laplace estimator is Witten-Bell smoothing, that has been used in the work by Craven and his collegues on text categorization [CDF+00].

The main weakness of this naive Bayesian classifier is that it presents problems when one wants to interpret the score for each class as an estimate of uncertainty. If for some word w, the value of $P_c(w|c_i)$ differs by one order of magnitude between different classes c_i, then the final probabilities will differ by as many orders of magnitude as there are words in the document. As a consequence, scores for the winning class tend to be close to 1.0 while scores for the losing classes tend toward 0.0. For instance, Bennet [Ben00] shows this phenomenon on two classes (Earn and Corn) of the well-known Reuters 21578 dataset. These extreme values are an artefact of the independence assumption. Class-conditional word probabilities would be much more similar across classes if word dependencies were taken into account [CDF+00]. An additional problem in the above formalization is strictly related to the probability estimation in formula 4, which regards all documents belonging to ci as one huge document. In other words, this estimation method does not take into account the fact that there may be important differences among term occurrences from documents with different lengths [KRYL02] and estimation could be affected by significant length discrepancy among documents belonging to the same class [Seb02]. As observed by Eyheramendy et al. [ELM03], "directly incorporating document length into the multinomial model has little effect due to

the extreme probability estimates produced by the naive Bayes-type models. One possibility would be to correct for the bias before introducing length".

In our proposal we adopt a normalization of the value $TF(w,d)$ in formula 4.6 in order to avoid these problems. In particular, we normalize TF according to the following formula:

$$NormalizedTF(w,d) = \frac{TF(w,d)}{\|TF(\bullet,d)\|_2} \tag{4.8}$$

where

$$\|TF(\bullet,d)\|_2 = \sqrt{\sum_{w' \text{ in } d} TF(w',d)^2}.$$

By substituting $TF(w,d)$ with $NormalizedTF_c(w,d)$ in 4.7, we have:

$$P_c(c_i|d) = \frac{P_c(c_i) \cdot \prod_{w \in FeatSet_c} P_c(w|c_i)^{NormalizedTF(w,d)}}{\sum_{c' \in DirectSubCategories(c)} P_c(c') \cdot \prod_{w \in FeatSet_c} P_c(w|c')^{NormalizedTF(w,d)}} \tag{4.9}$$

We observe that this normalization does not change the assignment of a document to a class: it only contributes to smooth the values of the posterior probabilities and to make the thresholding algorithm more effective, since choosing a threshold when probability values are all 0 or 1 would not help in hierarchical text classification. A similar normalization, but to L1-norm, has been proposed in [SJ03].

Centroid-Based classifier

Linear classifiers are a family of learning algorithms that learn a feature weight vector (or prototype)

$$\vec{c_i} = \langle w_{i1}, w_{i2}, \ldots, w_{i|FeatSet_c|} \rangle$$

for every category c_i. In our framework, where a document d temporarily assigned to a category c has to be possibly assigned to a category $c_i \in$*DirectSubCategories(c)*, the dimensionality of the feature weight vector of c_i corresponds to the size of $FeatSet_c$. The score returned by a linear classifier for a document d and a category c_i is the dot product between the feature vector describing d and $\vec{c_i}$(hence the *linearity* of the classifier). Generally, the dot product (or equivalently, both the document and the class vectors) is normalized to unit as follows:

$$\frac{\vec{d} \cdot \vec{c_i}}{\left\|\vec{d}\right\|_2 \|\vec{c_i}\|_2}$$

This normalization represents the cosine of the angle spanned by the two vectors d and $\vec{c}_i$. It is a similarity measure (also known as *cosine similarity*), therefore, the higher the value, the more similar the document d and the category prototype $\vec{c}_i$.

The most well-known linear classifier is an adaptation to text categorization of Rocchio's formula originally proposed for relevance feedback in the context of information retrieval [Roc71]. The learning method, denoted as Rocchio method, computes the weights of $\vec{c}_i$ as follows:

$$w_{ij} = \beta \sum_{d \in Training(c_i)} \frac{d_j}{|Training(c_i)|} - \gamma \sum_{d \in Training(c/c_i)} \frac{d_j}{|Training(c/c_i)|}$$

where d_j denotes the j-th component of the document vector, *Training*(c_i) is the set of positive documents of category c_i, and *Training*(c/c_i) in our framework is the set of negative examples for c_i. The control parameters β and γ define the relative impact of positive and negative examples in the definition of the class prototype. Dumais et al. [DPHS98], Joachims [Joa97], Han and Karypis [HK00], Lertnattee and Theeramunkong [LT04] set β to 1 and γ to 0, so that the prototype of a class coincides with the *centroid* of its positive training examples. In this work we follow this mainstream and compute the classification score as the cosine similarity between the document vector and the centroid of a class. The main difference is that document vectors contain the term frequencies, that is $d_j = TF_d(w_j)$, while all mentioned works do operate on *tfidf* representations, that is, the weight associated to the j-th feature is the product of the term frequency of the term w_j in d, $TF_d(w_j)$,and the logarithm of the inverse document frequency, *IDF*(w_j). The document frequency is defined as the percentage of documents in the collection where the term w_j occurs.[5] The *tfidf* representation embodies the intuition that

- the more often a term occurs in a document, the more it is representative of its content, and
- the more documents a term occurs in, the less discriminating it is [Seb02].

The second intuition is appropriate for document indexing, that is, the task of information retrieval for which Salton and Buckley [SB88] defined the *tfidf* representation. However, for text categorization tasks, the usage of the IDF factor seems counterintuitive. In the feature selection phase, the most discriminant features are selected, such that they correspond to terms that occur frequently in documents of the same category. The IDF factor would penalize mainly the best discriminative features, while it would weight more those terms that occur frequently in a single document. A confirmation of our observation is indirectly given by Debole and Sebastiani [DS03] who suggest replacing the IDF factor with the value taken by the feature selection measure. Therefore, in this work the weight associated to the j-th feature of a document is based exclusively on the TF factor. In this case, the value associated to the feature w for the *centroid* of the category c_i is defined as follows:

[5] The document frequency used in the *tfidf* representation should not be confused with $DF_{c'}(w)$ defined in Section 4, which depends on the category c'.

$$P_c(w, c_i) = \frac{\sum_{d \in Training(c_i)} TF_d(w)}{|Training(c_i)|} \tag{4.10}$$

and the mathematical formulation of the *cosine correlation* is the following:

$$Sim_c(c_i, d) = \frac{\sum_{w \in FeatSet_c} P_c(w, c_i) \times TF_d(w)}{\sqrt{\sum_{w \in FeatSet_c} P_c(w, c_i)^2 \times \sum_{w \in FeatSet_c} TF_d(w)^2}} \tag{4.11}$$

It is noteworthy that the cosine correlation returns a particularly meaningful value when vectors are highly dimensional and features define orthogonal directions. As pointed out in Section 4.1.1 our feature selection algorithm guarantees a kind of orthogonality property which applies to the group of features extracted from each category dictionary rather than to the individual features. Therefore, the procedure adopted for feature selection seems to be coherent with this classifier as well.

We conclude by highlighting another difference with respect to related papers by Joachims [Joa97], [HK00] and [LT04] where all features[6] are used in their experiments. In this work, features are preliminarily filtered and only those deemed most discriminant do actually contribute to the classification. This seems to improve the accuracy of Rocchio classifiers [RS02] which can achieve quite competitive performance if properly trained [SSS98].

SVM-probabilistic classifier

Recently, a new learning technique has emerged and become quite popular in text categorization because of its good performance and its theoretical foundations in the computational learning theory: support vector machines (SVMs), proposed by Vapnik [Vap95]. Given a set of positive and negative examples (SVMs are defined for two-classes problems)(See section 2.1.6)$\{(\vec{x}_1,y_1), (\vec{x}_2,y_2), \ldots, (\vec{x}_N,y_N)\}$, where $\vec{x}_i \in \mathrm{R}^m$ ($\vec{x}_i$is a document vector) and $y_i \in \{-1,+1\}$, an SVM identifies the hyperplane in R^m that linearly separates positive and negative examples with the maximum margin (*optimal separating hyperplane*). In general, the hyperplane can be constructed as the linear combination of all training examples, however, only some examples, called *support vectors*, do actually contribute to the optimal separating hyperplane, which can be represented as:

$$f(x) = \sum_{i=1}^{N^*} y_i \alpha_i \vec{x}_i^* \cdot \vec{x} + b \tag{4.12}$$

where $\vec{x}_i^*$, i=1,2, ... N^*, are the support vectors. The coefficients α_i and b are determined by solving a large-scale quadratic programming problem for which efficient algorithms exist, which are guaranteed to find the global optimum.

[6] Joachims [Joa97] actually filters out all features that occur less then three times in the training documents.

SVMs are based on the *Structural Risk Minimization* principle: a function that can classify training data accurately and which belongs to a set of functions with the lowest capacity (particularly in the VC-dimension) [Vap95] will generalize best, regardless of the dimensionality of the feature space m. Therefore, SVMs can generalize well even in large feature space, such as those used in text categorization. In the case of the separating hyperplane, minimizing the VC-dimension corresponds to maximizing the margin.

The linear separability appears to be a strong limitation, however, as experimentally observed by Joachims [Joa98], most text categorization problems are linearly separable. In any case, SVMs can be generalized to non-linearly separable training data by mapping the data into another *feature space* F via a non-linear map:

$\Phi : \mathrm{R}^m \quad \rightarrow \quad F$

and then performing the above linear algorithm in F. Generally the map introduce new features that do take into account the p-order correlation between the input features. Since the solution has the form:

$$f(x) = \sum_{i=1}^{N^*} y_i \alpha_i \Phi(\vec{x})_i^* \cdot \Phi(\vec{x}) + b \tag{4.13}$$

it is non linear in the original feature set. Yang and Liu [YL99] report that they tested the linear and non-linear models offered by the SVM^{light} system [Joa][Joa98], and obtained "a slightly better result with the linear SVM than with the non-linear models". Therefore, in our experiments we will use only linear models.

The SVM embedded in WebClassIII is a modified version of the Sequential Minimal Optimization classifier (SMO) [Pla98]. The method developed by Platt is very fast and is based on the idea of breaking the large quadratic programming (QP) problem down into a series of smaller QP problems that can be solved analytically. The same system has been used by Dumais et al. [DPHS98] in an empirical comparison of five different learning algorithms for text categorization.

Modification of Platt's original method is necessary in our framework, since the classifier learned for each internal node of the hierarchy is of the kind one-of-r (multi-class problem). More precisely, a binary classifier is learned for each couple of classes and afterwards, the probability $P_c(c_i|d)$ is computed by means of a probabilistic pair-wise coupling classification [HT98]. Once again, the decision taken by the classifier for each training document is associated with a (probabilistic) score, which is processed by the automated thresholding algorithm as explained in Section 2.4.1.

Learning complexity

To evaluate the learning complexity of the learning algorithms, we have to consider the analysis of complexity reported in section 2.4.3. In particular, we showed that the complexity of the hierarchical framework is (Equation 2.19):

$$\sum_{i=1}^{d} k^i f(k, \frac{n}{k^i}, a) \tag{4.14}$$

where d is the depth of the hierarchy, a is the number of features, f(number of classes, number of training examples, number of features) is the learning complexity of a generic classification algorithm, r is the total number of classes, n be the number of training examples and k is the number of children of a generic internal node (we suppose that k is constant).

In the case of both naive Bayes and centroid based classifiers, the complexity of the learning phase is linear in the number of training documents, in the number of features and in the number of classes [HK00], [Mit97]. In such a case the time complexity of a flat classifier is O(n·a·r), while in the case of hierarchical framework, it is:

$$O(\sum_{i=1}^{d} k^i \cdot ((\frac{n}{k^i} \cdot k \cdot a)) = O(\sum_{i=1}^{d} (n \cdot k \cdot a)) = O(d \cdot n \cdot k \cdot a)$$

Both are linear in the number of training examples and in the number of features. The difference is that the complexity of a flat classifier is linear in the number of classes, while the complexity of the hierarchical framework is linear in the product of the number of children of each node and the depth of the tree. Under the assumption of a balanced hierarchy with constant branching factor k, we have d=$log_k r$. Therefore the complexity of the hierarchical framework is O($n \cdot a \cdot log_k r$).

In the case of SVM classifier, the complexity is linear in the number of training documents, features and classes [Pla98]. However, the SMO has been modified to deal with multi-class problems and to estimate the probability $P_c(c_i|d)$. This probability is computed by means of a probabilistic pair-wise coupling classification [HT98]. This modification makes the algorithm linear in the number of examples and cubic in the number of classes. Therefore the time complexity of a flat classifier is O(n·a·r^3), while in the case of hierarchical framework it is:

$$O(\sum_{i=1}^{d} k^i \cdot ((\frac{n}{k^i} \cdot k^3 \cdot a)) = O(\sum_{i=1}^{d} (n \cdot k^3 \cdot a)) = O(d \cdot n \cdot k^3 \cdot a)$$

Under the same assumptions given for naive Bayes and centroid-based classifiers, the complexity of the hierarchical framework is O($n \cdot a \cdot log_k r$).

This analysis can be refined by taking into account that the value of a (i.e. number of features) may change level by level. More precisely:

- $a = n_{dict} \cdot r$ in the flat classifier,
- $a = n_{dict} \cdot k$ in the hierarchical framework with proper feature set,
- $a < n_{dict} \cdot r$ in the hierarchical framework with hierarchical feature set.

Actually, in the case of hierarchical framework with hierarchical feature set, the number of features depends on the level of the hierarchy to which the classifier is associated. For the first level $a = n_{dict} \cdot r$, in the second level $a = n_{dict} \cdot (r - k)$, in the third level $a = n_{dict} \cdot (r - k - k^2)$ and so on.

4.1.3 Experimental Results

In this section we seek answers to the following questions with empirical evidence:

- Does the hierarchical classifier built with the proposed framework improve the performance when compared to a flat classifier?
- Does the proposed framework minimize the (tree) distance between the correct class and the returned one when the document is not correctly classified?
- Does the proposed framework actually improve the computational efficiency of the learning algorithms?
- What feature selection strategy is the most promising for hierarchical categorization?
- Which classifier has the best performance within the proposed framework?

Before describing results, we illustrate the three corpora used for this study and the performance evaluation measures considered for performance evaluation.

Datasets

The three corpora chosen for this study are the recently published benchmark dataset Reuters Corpus Volume I (RCV1) [LYRL04], and two collections of HTML documents (WebClass is specifically designed to classify HTML pages) referenced either in the Yahoo! Search Directory [7] or in a web directory developed in the Open Directory Project (ODP) [8]. The three corpora differ considerably in the training set size, in the hierarchical structure of categories as well as in the procedure adopted for the classification of documents. For the sake of completeness, a brief description of the document collections is reported in the following.

Reuters Corpus Volume 1

Reuters Corpus Volume I (RCV1) is a benchmark dataset widely used in text categorization and in document retrieval. It consists of over 800,000 newswire stories, collected by the Reuters news and information agency, that have been manually coded using three orthogonal category sets. Therefore, category codes from three sets (Topics, Industries, and Regions) are assigned to stories:

- Topic codes capture the major subject of a story.
- Industry codes are assigned on the basis of the types of business discussed in the story.
- Region codes include both geographic locations and economic/political groupings.

[7] http://dir.yahoo.com/
[8] www.dmoz.org

In our study, similarly to other authors [ZJYH03], we use topic codes for categorization.

The main characteristic that makes RCV1 particularly suitable in our study is the adopted coding policy. In particular, topics are organized hierarchically. The hierarchy of topics consists of a set of 104 categories organized in a 4-levels hierarchy.

We pre-processed documents as proposed by Lewis et al. and, in addition, we considered only documents associated to a single category. This selection is due to the fact that in this study we are interested in investigating single category assignment (feature selection method, learning algorithms, categorization framework, and performance evaluation functions are all based on the assumption that a document can be assigned to one category at the most). The removal of documents associated with multiple classes has been also adopted by other authors on different datasets in the evaluation of single-label corpora [SS00].

We separate the training set and the testing set using the same split adopted by Lewis et al. In particular, documents published form August 20, 1996 to August 31, 1996 (document IDs 2286 to 26150) are included in the training set while documents published from September 1, 1996 to August 19, 1997 (document IDs 26151 to 810596) are considered for testing. The result is a split of the 804,414 documents into 23,149 training documents and 781,265 test documents. After multiple-label documents removal, we have 150,765 documents, (4,517 training documents and 146,248 testing documents).

Yahoo dataset

The second data set used in this experimental study is obtained from the documents referenced in the Yahoo! Search Directory.[9] We extracted all 907 actual Web documents referenced at the top three levels of the Web directory http://dir.yahoo.com/Science. Empty documents and documents containing only scripts have been removed.

There are 6 categories at the first level, 27 categories at the second level and 35 categories at the third level. A document assigned to the root of the hierarchy is considered "rejected" since its content is not related to any of the 68 subcategories.

The dataset is analyzed by means of a 5-fold cross-validation, that is, the dataset is first divided into five *folds* of near-equal size, and then, for every fold, the learner is trained on the remaining folds and tested on it. The system performance is evaluated by averaging some performance measures (see below) on the five cross-validation folds.

dmoz dataset

The third data set used in this experimental study is obtained from the documents referenced by the Open Directory Project (ODP) (www.dmoz.org)[10]. We extracted all actual Web documents referenced at the top five levels of the Web directory

[9] Documents have been downloaded on the 15^{th} of July 2003. The dataset is electronically available at http://lacam.di.uniba.it:8000/phd/micFiles/yahoo_science_docs.zip.

[10] Documents have been extracted in April 2004. The dataset is electronically available at http://lacam.di.uniba.it:8000/phd/micFiles/dmoz_health_conditions_and_diseases_docs.zip.

rooted in the branch Health\Conditions_and_Diseases\. Empty documents and documents containing only scripts have been removed.

The dataset contains 5,612 documents in 221 categories organized in a five level hierarchy as follows:

- In the first level there are 21 categories and 340 documents.
- In the second level there are 81 categories and 1,514 documents.
- In the third level there are 85 categories and 2604 documents.
- In the fourth level there are 32 categories and 1099 documents.
- In the fifth level there are 2 categories and 55 documents.

The dataset is analyzed by means of a 5-fold cross-validation. The system performance is evaluated by averaging performance measures on the five cross-validation folds.

Both yahoo and dmoz datasets have been used in order to evaluate the performances of the system in presence of "noisy" documents and in presence of documents with no clearly predefined structure. This is not the case of the corpus RCV1 whose documents respect a well-defined XML structure.

Evaluation measures

Performances of the system have been evaluated on the basis of several measures. The first measure is the standard *accuracy* defined in machine learning to evaluate the performances of *1-of-r* classifiers. It represents the number of testing documents correctly classified over all testing documents. It is noteworthy that in the *1-of-r* classifiers context, this "narrowly" defined accuracy is indeed equivalent to the standard recall and is not equivalent to the standard definition of accuracy in text categorization literature that is given for classifiers based on binary decisions. In this case it is the proportion of correct assignments among the binary decisions over all category/document pairs. The standard text categorization accuracy measure is well-defined for documents with multiple categories; the narrowly defined accuracy is not. [YL99]. In our analysis, we use the narrowly defined accuracy because, as observed by Sebastiani [Seb02], in single-label text categorization, precision and recall are not independent of each other and in this case either precision or recall (machine learning accuracy) can be used as a measure of effectiveness.

Furthermore, we define other four evaluation measures in order to provide a more detailed evaluation of results. Intuitively, if a text categorization method misclassifies documents into categories similar to the correct categories, it is considered better than another method that misclassifies the documents into totally unrelated categories. Therefore, we define other four evaluation measures, namely:

1. the *misclassification error*, which computes the percentage of documents misclassified into a category not related to the correct category in the hierarchy.
2. the *generalization error*, which computes the percentage of documents misclassified into a supercategory of the correct category;

3. the *specialization error*, which computes the percentage of documents misclassified into a subcategory of the correct one;

4. the *unknown ratio*, that measures the percentage of rejected documents.

The sum of the accuracy, the generalization error, the specialization error, the misclassification error and the unknown ratio equals one.

Flat vs Hierarchical classifiers

The first question we investigate is the effectiveness of the hierarchical categorization framework with respect to flat classification. For a fair comparison, the thresholding algorithm has been used both for hierarchical and flat classification. In this way, both algorithms are able "reject" documents.

For evaluation purposes, several feature sets (proper or hierarchical) of different size have been extracted for each internal category in order to investigate the effect of this factor on the system performance. The feature set size ranges from 5 to 60 features per category in the case of RCV1 and Yahoo dataset, while it ranges from 5 to 40 in the case of dmoz dataset. Collected statistics concern the three classifiers.

Figure 4.2 shows the accuracy of different classifiers for the three datasets. Among flat classifiers, *SVM performs the best across the three datasets.* It is also noteworthy that in all datasets the SVM or centroid-based classifiers built according to the flat approach are more accurate than the corresponding two hierarchical classifiers built on proper or hierarchical feature sets. The situation is different for NB classifiers, which do benefit of the hierarchical framework in all three datasets. This is particularly evident for NB classifiers built from hierarchical feature sets. If we consider that the naive Bayesian classifier is particularly accurate when the number of attributes is relatively small [DP97], we can explain that the naive Bayesian classifier takes great advantage of the use of the Hierarchical framework.

Therefore, our second conclusion is that *there is an interaction, in terms of accuracy, between the hierarchical framework and the type of classifier.*

From a closer analysis of the percentage of errors (reject, misclassification, generalization and specialization) performed by the various classifiers (see Figures 4.3, 4.4, 4.5), we observe that the flat classifiers commit more rejection and misclassification errors (in percentage) than the corresponding hierarchical classifiers. Therefore, with reference to the second question, we conclude that, *even though SVM or centroid-based flat classifiers are more accurate than the corresponding hierarchical classifiers, they tend to commit "more serious" errors.*

This difference in error type is particularly significant for NB classifiers. Figure 4.6 shows the distribution of misclassification, specialization and generalization errors with respect to the (tree) distance of the wrong category from the correct one. Statistics refer to the dmoz dataset, which is the most complex in terms of number of categories and depth of the hierarchy. In general, errors are distributed quite "close" to the correct category, also thanks to the automated threshold definition algorithm that minimizes the sum of tree distances between the correct and the predicted categories. Nevertheless, results are better for the hierarchical classifier, since the distribution is more skewed towards low distance values.

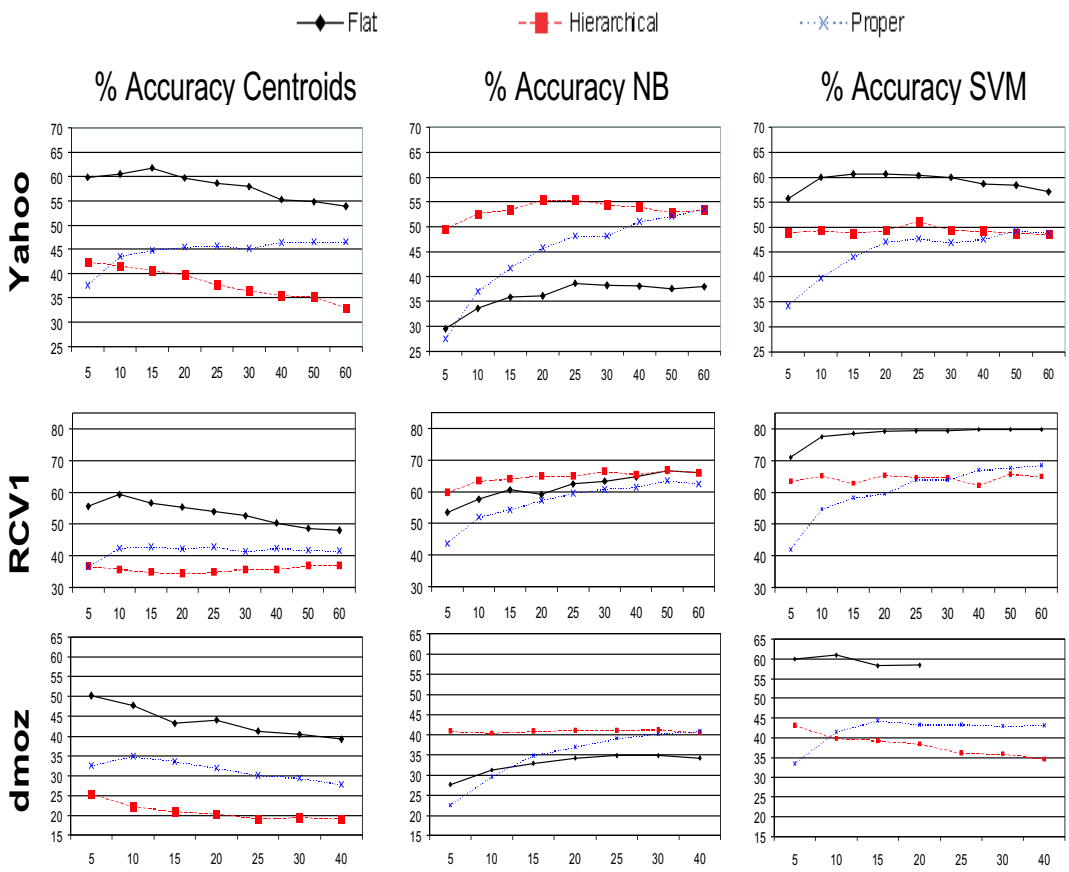

Figure 4.2: Accuracy for the three datasets: Flat vs Hierarchical with hierarchical feature set vs Hierarchical with proper feature set. Experimental results for dmoz with SVM are available up to 20 features per category.

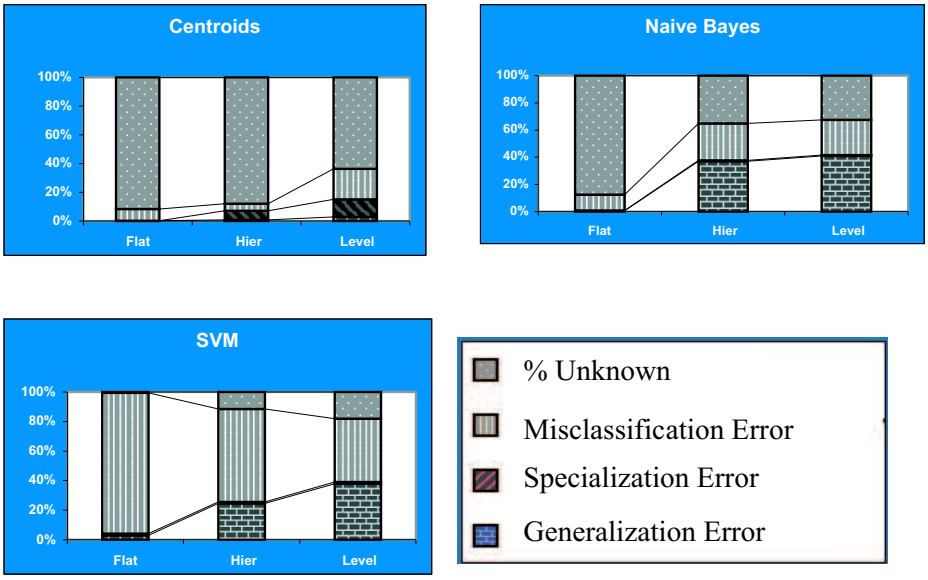

Figure 4.3: Distribution of errors for Reuters dataset (n_{dict}=60).

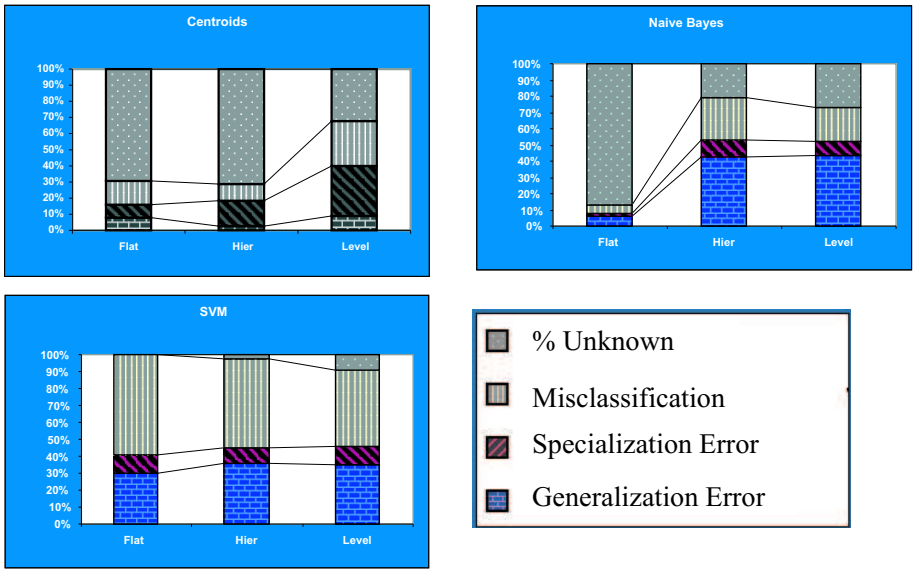

Figure 4.4: Distribution of errors for Yahoo dataset (n_{dict}=60).

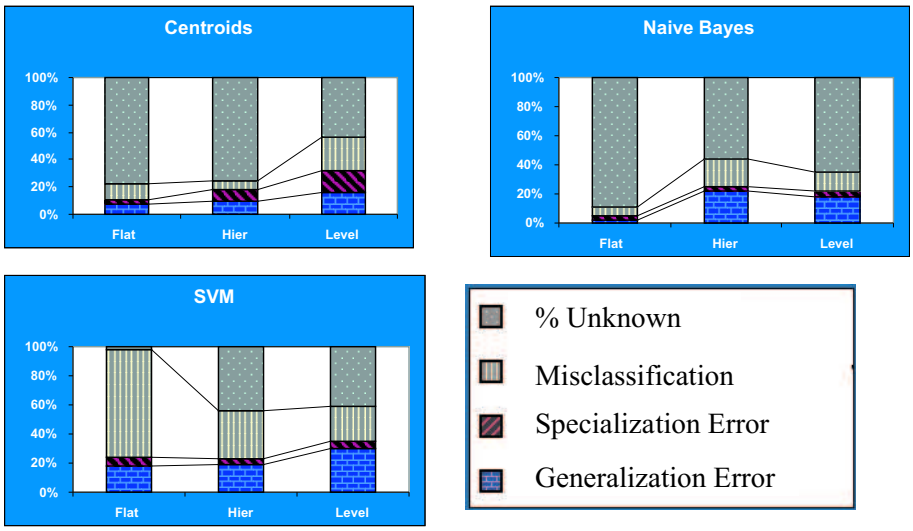

Figure 4.5: Distribution of errors for dmoz dataset (n_{dict}=20).

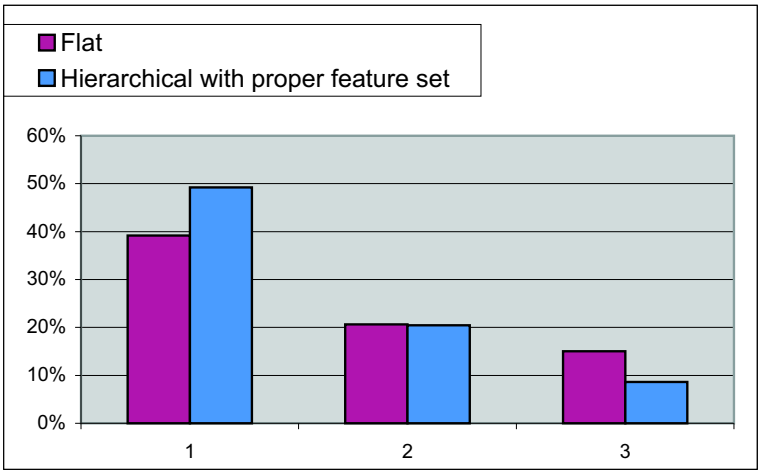

Figure 4.6: Distribution of errors. Percentage of misclassification, specialization and generalization errors classified at distance 1, 2 and 3 from the correct class. Statistics for larger distances are not shown. Results are obtained on the dmoz dataset, with Naive Bayes classifier, feature set size = 20.

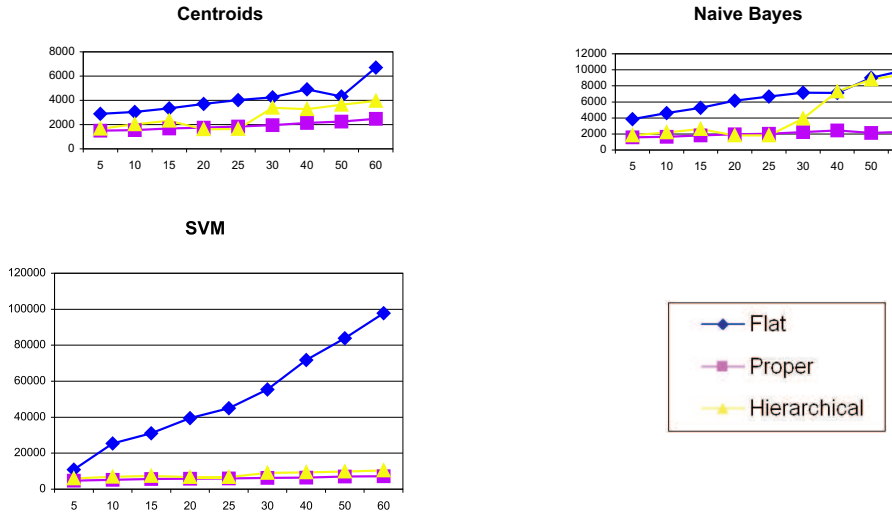

Figure 4.7: Learning running times on the RCV1 Dataset. Results are expressed in seconds varying the number of selected features. Results show the comparison between the Flat technique, hierarchical with a proper feature set and hierarchical with a hierarchical feature set. WebClassIII has been executed on a Pentium 4 PC 1.4GHz running a Windows 2000 Operating System.

To answer the question on the actual improvement of the computational efficiency of the learning algorithms, we collected statistics on the running time (see Figure 4.7). *Results are substantially in favor of the hierarchical framework*. The difference is particularly evident in the case of the SVM classifier. This confirms the analysis of complexity reported in section 2.4.3 and in section 4.1.2

The results also show the better performances of the hierarchical framework with a proper feature set, with respect to the hierarchical framework with a hierarchical feature set. This also confirms the formal analysis of complexity reported in section 4.1.2 and, in particular, the role of the number of features, which grows proportionally to the total number of classes in the case of a hierarchical feature set.

Comparing hierarchical classifiers

In the previous section we answered questions on the pros and cons of hierarchical classifiers when compared to flat classifiers. In this section, we investigate aspects specifically related to the hierarchical classifiers, namely, which is the best strategy for feature selection and what is the best classifier to use in combination with the hierarchical categorization framework

From the results shown in Figure 4.2 we observe that for smaller feature sets the hierarchical approach performs better than the proper approach. However, as the number of features increases, the classifier trained with a proper feature set asymptotically tends to the performances of the classifier trained with a hierarchical feature set. This can be explained by the observation that, with a limited number of features, the lower categories are not represented and it is necessary to use a

hierarchical feature set. By increasing the number of features, the deeper categories are better represented and the benefits of a hierarchical approach vanish.

For the comparison of classifiers, we limit our study to proper feature sets. Once again, several feature sets of different size have been extracted for each internal category, in order to study the effect of this factor on the classifier performance. Sizes range from 5 to 60 features per category in the case of the RCV1 and the Yahoo dataset, while it ranges from 5 to 40 in the case of dmoz dataset. Collected statistics concern centroid-based, naïve Bayes (NB) and SVM classifiers.

Figures 4.8, 4.9 and 4.10 show the performances of different classifiers for each document collection and for different sizes of the proper feature sets. For the RCV1 and the dmoz datasets, which are characterized by a complex hierarchy (both in the number of categories and in the depth of the tree structure), the best results in terms of accuracy are obtained by the SVM classifier, while for the Yahoo dataset, the naive Bayes classifier performs best for sufficiently large feature sets. The centroid-based classifier shows the worst performance, particularly when the size of the feature set increases.

Looking at the errors committed in detail, it is interesting to note that:

- NB and SVM show the same trend, which is different from the trend of centroids. For example, while for SVM and NB the specialization error is low and the generalization error tends to be quite high, the situation is reversed for centroids.

- Increasing the number of the features, the percentage of misclassifications for NB and SVM increases, while the percentage of "rejected" documents (unknown error) decreases. This behavior is reversed for centroids.

The different behaviour can be explained by the fact that the thresholding algorithm tends to be generally conservative (i.e. high thresholds and few documents passed down) for SVM and NB, while in the case of centroids the thresholds become more selective only for larger feature sets. Indeed, the scores computed by centroid-based classifiers are unevenly distributed at the extremes of the unit interval when only a few features determine the result of the classification. In this situation of binary-like classification, the thresholding algorithm cannot work properly. On the contrary, the scores are less extreme in large feature spaces and the thresholding algorithm can work properly by reducing the high number of misclassifications, at the cost of increasing the rejection rate.

Comparing NB and SVM it is noticeable that SVM has a higher misclassification rate, while NB has a higher rejection rate. This means that even when they do not perform best, NB classifiers can be a valid alternative to SVM in those application contexts where a “commission error” is considered more serious than an “omission error”.

4.1.4 Related work

Some of the related works have been presented in section 2.4.4 in the context of hierarchical classification. Here, we integrate the description already reported fo-

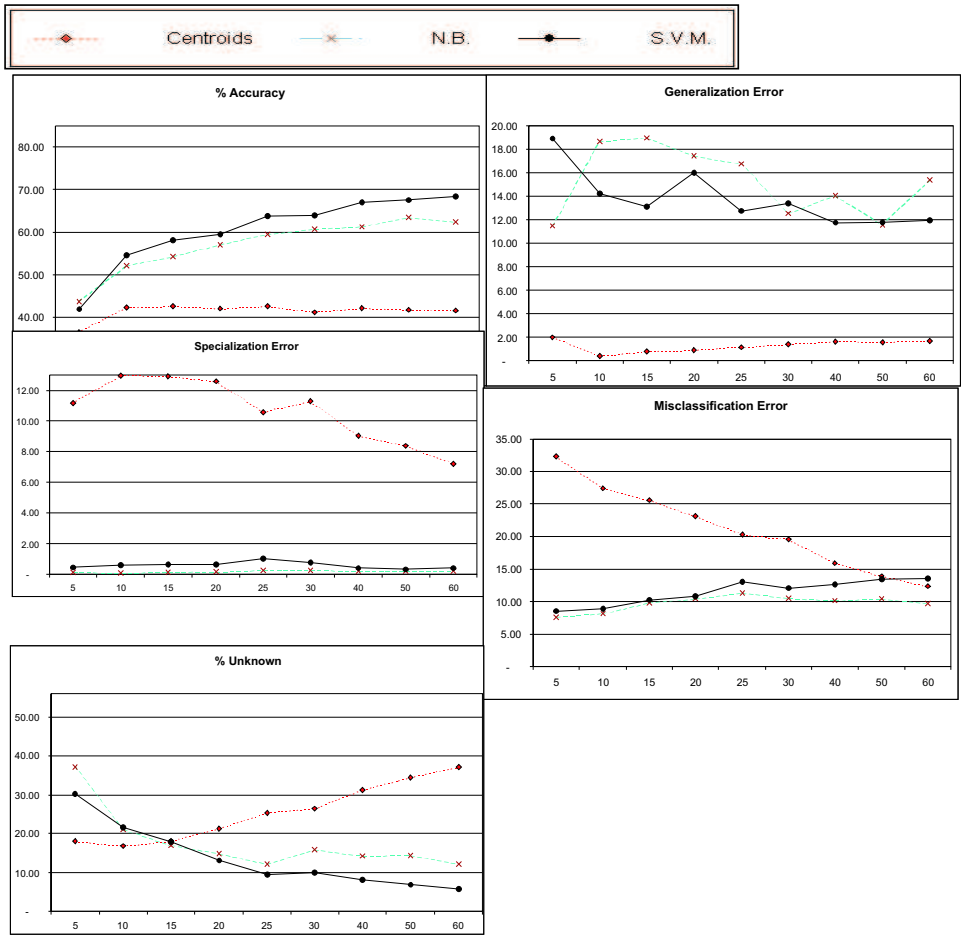

Figure 4.8: Classifier comparison on the RCV1 collection. Features are extracted using proper feature sets.

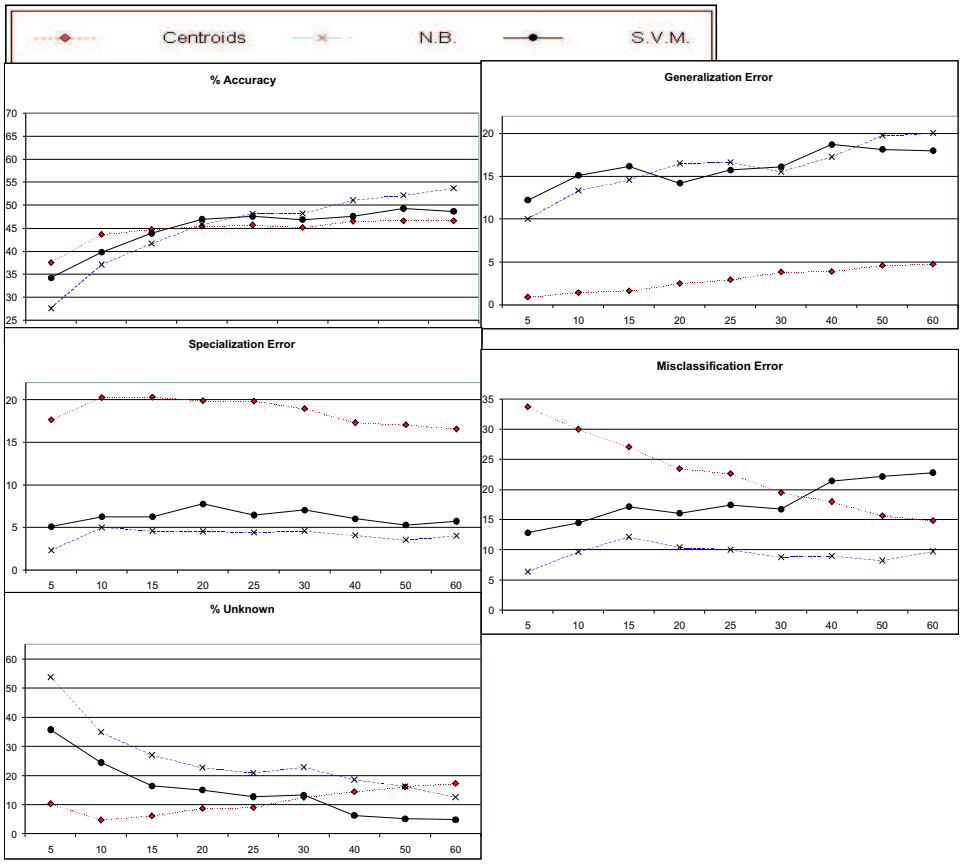

Figure 4.9: Classifier comparison on the RCV1 collection. Features are extracted using proper feature sets.

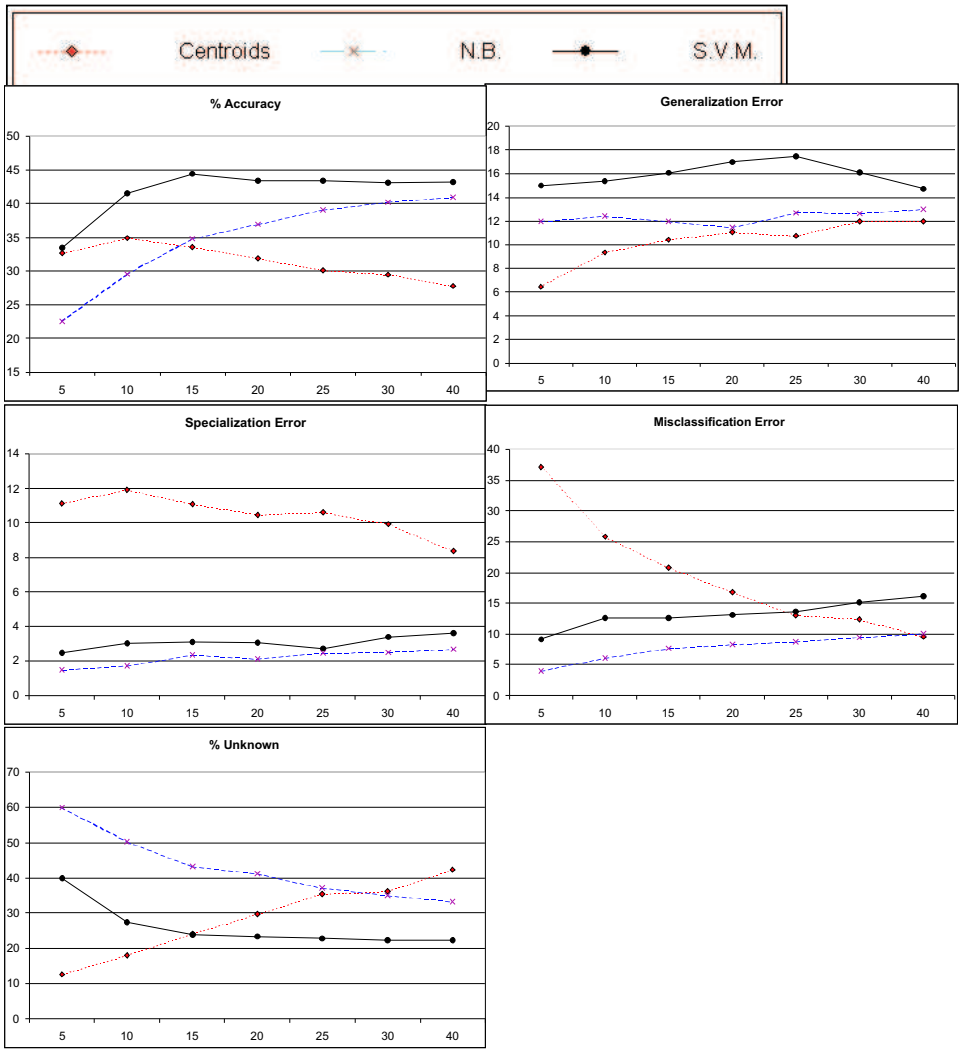

Figure 4.10: Classifier comparison on the dmoz dataset. Features are extracted using a proper feature set.

cusing on particular aspects of the task in hand, that is, text categorization. We also investigate differences of our approach with respect to existing approaches both in terms of the method and in terms of experimental results.

In the seminal work by Koller and Sahami [KS97] the hierarchy of categories is used in every processing step. For the feature extraction step a category dictionary is built for each node in the hierarchy. Feature extraction is based on an information theoretic criterion that eliminates both irrelevant and redundant features. For the learning step, two classifiers are used, namely the naive Bayes and KDB [Sah96].

McCallum et al. [MRMN98] proposed a method based on the naive Bayes learner. A unique feature set is defined for all documents by taking the union of all category vocabularies. Features for a given category are selected by means of mutual information at each internal node of the tree, using the node's immediate children as classes.

In the work by D'Alessio et al. [DMSK00] documents are associated only to leaf categories of the hierarchy. Two sets of features are associated to each category, one is positive (features extracted from documents of the category), while the other is negative (features extracted from documents of sibling categories in the hierarchy).

Dumais and Chen [DC00] use the hierarchical structure for two purposes. First, to train several SVMs, one for each intermediate node. The sets of positive and negative examples are constructed from documents of categories at the same level, and different feature sets are built, one for each category. Second, to classify documents by combining scores from SVMs at different levels. An empirical comparison based on a large heterogeneous collection of pages from LookSmart's web directory showed small advantages in accuracy for hierarchical models over flat models.

In the system CLASSI by Ng et al. [NGL97], the hierarchical classification of documents is obtained by combining several linear classifiers according to a tree structure (hierarchical classifier). Weights of each linear classifier are determined by means of the perceptron learning algorithm. Two peculiarities of this work are the use of WORDNET [Mil90] to replace each word with its morphological root form and the use of the correlation coefficient to select the best subset of words. However, F1-score values reported on the Reuters dataset are well below those reported by Yang [Yan99] on the same dataset.

A summary of the referenced papers is reported in Table 4.1 and in Table 4.2. We are aware that the list of related works summarized in the table is not exhaustive, although it is representative of the most well-known contributions. For the sake of completeness, we report a brief note on three additional works. Sun and Lim [SL01] have proposed the use of category-similarity measures and distance-based measures to consider the degree of misclassification in measuring the classification performance. Experiments were performed on the Reuters-22173 collection with an SVMlight Version 3.50 implemented by Joachims [Joa]. Chuang et al. [CTYG00] have tested a Rocchio-based classifier on a collection of approximately 200 documents on professional baseball and basketball news. Finally, Tikk and Biró [TB03] tested a centroid-based classifier on the WIPO-alpha (World Intellectual Property

Organization, Geneva, Switzerland, 2002)[11] English patent database that consists of about 75000 XML documents distributed over 5000 categories in four levels. Unfortunately, studies on the WIPO-alpha collection are not publicly available because of the strongly business sensitive nature of the research. As future work, we plan to extend our experimental results to this dataset as well.

Comparison with related work: the method

Our work differs from previous studies in several respects. First, documents can be associated to both internal and leaf nodes of the hierarchy. Surprisingly, this aspect is explicitly considered and tested only in [Mla98b] and [RS02]. However, unlike Mladenić's work, we consider actual Web documents referenced in the Yahoo! ontology, and not only the items which briefly describe them in the Yahoo! Web directories. Other special conditions that are considered in this work are: 1) no document for some internal nodes; 2) some internal nodes have only one child.

A second difference is in the feature selection process for each internal category. In WebClassIII it is based on an upgrade of the technique implemented and tested in [MEC02], named $maxTF \times DF^2 \times ICF$. Unlike other feature selection methods proposed in the literature on hierarchical document categorization [MG99], $maxTF \times DF^2 \times ICF$ answers the demand for terms that are shared by most of the documents of the same category and possibly no document of other categories. Moreover, it considers the document (and not a term) as an observation unit.

A third difference is that we do not propose a specific method, but we investigate a framework for hierarchical text categorization that can be applied to any classifier that returns a degree of membership (e.g. distance or probability based) of a document to a category. We applied the framework to three classifiers, two of which present some variants with respect to the original methods reported in the literature.

The fourth difference is in the development of a technique for the automated selection of thresholds for the degree of membership returned by the classifier. The thresholds are used to determine whether a document has to be passed down to one of the child categories during the top-down classification process.

Finally, we define new measures for the evaluation of the system performances in order to capture some aspects related to the "semantic" closeness of the predicted category to the actual one.

We conclude by observing that the main contribution of this work is the systematic investigation of the usage of information provided by the category hierarchy in all aspects of text categorization, such as definition of training sets, feature sets, classifiers, threshold-based document classification and evaluation measures.

Comparison with related work: experimental results

Previous studies on hierarchical text categorization have already contributed to clarifying some aspects that have not been explored in this work. Koller and Sahami [KS97] experimentally showed that there is a substantial improvement in accuracy

[11] http://www.wipo.int/

when feature selection is aggressively employed versus the case where all domain features are used. This improvement has been observed both in the hierarchical case and in the flat case for Bayesian classifiers. McCallum et al. [MRMN98] show that aggressive feature selection is not necessary if shrinkage is used to smooth parameter estimates. Shrinkage helps especially when training data are sparse, which is the case when small sets of documents are assigned to leaf categories. Mladenić [Mla98b] compared six feature selection techniques for automatic document categorization, based on text hierarchies and her conclusions were in favor of Odds ratio when combined with a naive Bayes classifier. D'Alessio et al. [DMSK00] investigated the possibility of restructuring a pre-existing hierarchy, and concluded that the usage of a hierarchy, either modified or built from scratch, can significantly improve both the speed and effectiveness of the categorization process. Dumais and Chen [DC00] explored two ways to combine probabilities returned by the classifiers for the first and second level of a two-level hierarchy. The multiplicative approach assigns the document to a leaf if the product of both probabilities exceeds a given threshold, which is unique for all categories. The Boolean approach assigns the document to a leaf if the threshold is exceeded at every level. No difference between the two approaches was observed in terms of F1 measure, hence leading to the recommendation for the Boolean approach which is the most efficient. Ruiz and Srinivasan [RS02] reported good results for the (flat) Rocchio classifier when both training data and features are selected, and categories have a medium/high number (≥ 15) of training examples. Results reported by Weigend et al. [WWP99], who observed that the largest gains in average precision for the hierarchical classifier concern "rare" (i.e., with few training examples) categories, are also consistent with Ruiz and Srinivasan's findings. The main difference between the two findings is that in the work by Ruiz and Srinivasan, rare categories can occur at any node in the hierarchy, while in the work by Weigend et al. they are always leaf categories.

As to the real advantages of the hierarchical vs. flat approach, no conclusive result has been reported for predictive accuracy. Koller and Sahami [KS97] observed that the hierarchical approach appears to provide few benefits when attention is restricted to simple classifiers, such as naïve Bayes. Dumais and Chen [DC00] reported minor improvements for hierarchical models over flat models. Similarly, Ruiz and Srinivasan [RS02] do not show a clear superiority of the HME with respect to Rocchio. On the contrary, McCallum et al. [MRMN98] demonstrate that shrinkage with a class hierarchy significantly reduces the classification error, Ng et al. [NGL97] report accuracy improvements of the hierarchical method with respect to the flat method, and Weigend et al [WWP99] attribute a statistically significant overall improvement of 5% for averaged precision to the hierarchical approach. This confirms our experimental observation that there is an interaction, in terms of accuracy, between the hierarchical framework and the type of classifier.

All related works examined here show the clear computational advantage of the hierarchical approach. We have confirmed this conclusion both analytically and experimentally. This work, however, presents additional empirical findings not reported elsewhere. They are summarized in the following points:

1. Among flat classifiers, SVM performs the best across the three datasets.
2. Even though SVM or centroid-based flat classifiers are more accurate than the corresponding hierarchical classifiers, they tend to commit "more serious" errors ("severity" is based on a tree-distance measure).
3. As the number of features increases, the classifier trained with a proper feature set asymptotically tends to the performances of the classifier trained with a hierarchical feature set.
4. Errors committed by NB and SMV show the same trend, which is different from the trend of centroids.
5. Increasing the number of the features, the percentage of misclassifications for NB and SVM increases, while the percentage of "rejected" documents (unknown error) decreases. This behavior is reversed for centroids.

All these results, which extend those reported in a previous work [CM03] [CMLE03], are obtained by extensive experimentation on three datasets with category hierarchies of different complexity.

4.1.5 Conclusions

Most of the research on text categorization has focused on classifying text documents into a set of categories with no structural relationships among them. However, in this case it is difficult to browse or search documents in a large number of categories. Hierarchies are often used to make large collections of document categories more manageable, since they permit the application of the well-known principle of divide-and-conquer. The hierarchical structure is employed in many Internet directories (e.g. Yahoo and Google Directory) and in text databases (e.g., MEDLINE and patent databases), as well as in other document management tools (e.g. Netscape Bookmark). Therefore, whether and how to exploit the additional information on the hierarchical structure among categories in text categorization is an important issue that demands systematic investigation.

Our research adds to a growing body of work exploring how hierarchical structures can be used to improve the efficiency and efficacy of text classification. We have presented and evaluated a hierarchical text categorization framework that involves the hierarchy of categories in all phases of text categorization, namely feature extraction, learning, and classification of a new document. Our conclusion is that for large collections of documents organized in complex hierarchies, the hierarchical approach can offer two main advantages: efficiency gain and reduction of severity of classification errors. The former is particularly important when the hierarchy of categories is subject to changes, since in the flat approach changes affect all classifiers, while in the hierarchical approach they are all localized. The latter advantage is quite important if a trained user cannot supervise decisions taken by the document classifier.

Although we observed good results for the flat SVM across all three datasets used in our experimental validation of the framework, in the hierarchical approach the

Work	**Hierarchy**	**Feature Sets**	**Feature selection**	**Learning**	**Training set**	**Classification**
Koller & Sahami [KS97]	Doc.s only at the leaves	A separate feature set for each category	Probabilistic approach	Naive Bayes & KDB. A *1-of-r* classifier for each internal node	One hierarchical training set per category	Greedy search of a single classification path. Single category assignment
McCallum et al. [MRMN98]	Doc.s only at the leaves	A unique feature set built from category vocabularies	Mutual information	Shrinkage + *1-of-r* naive Bayes classifier. Parameters estimated for each category.	Single set	Both greedy and extensive search of classification paths. Single category assignment.
Mladenić [Mla98b]	Doc.s at any node	A separate feature set for each category	Several measures tested	*1-of-r* naive Bayes classifier	One hierarchical training set per category	Extensive search with pruning. Single class assignment
D'Alessio et al. [DMSK00]	Doc.s only at the leaves	A positive and negative feature set per category	A variant of the ACTION algorithm	Feature weight estimation.Both binary and *1-of-r* classifier	One set per category. Pos.: docs of the categoryNeg.: docs of the parent category	Both greedy and extensive search with pruning. Single or multiple category assignment
Dumais & Chen [DC00]	Doc.s only at the leaves	A separate feature set for each category.	Mutual information	Binary SVM classifier	One set per hierarchy level, with docs of all categories at the same level	Extensive search with pruning. Multiple category assignment
Ng et al. [NGL97]	Doc.s only at the leaves	A separate feature set for each category	Correlation coefficient	Binary Perceptron-based classifier	One set per category. Pos.: docs of the categoryNeg.: some selected docs	Extensive search. Multiple category assignment

Table 4.1: Classification of previous works (1 of 2)

Work	**Hierarchy**	**Feature Sets**	**Feature selection**	**Learning**	**Training set**	**Classification**
Ruiz & Srinivasan [RS02]	Doc.s at any node	A separate feature set for each category	Correlation coefficient, Mutual information, Odds ratio	Neural Networks for binary classification	One set per category. Pos.: docs of the categoryNeg.: some selected docs	Extensive search. Multiple category assignment
Weigend et al. [WWP99]	Doc.s only at the leaves	Both separate and unique feature set	LSI and χ^2	Neural Networks for binary classification	One hierarchical training set per category	Extensive search. Multiple category assignment
This work WebClassIII	Doc.s at any node. Internal nodes without docs and single-child are allowed	A separate feature set (hierarchical or proper) for each internal category	$maxTF \times DF^2 \times IC$	*1-of-r* naive Bayes, centroid-based and SVM-based classifiers. Automatic threshold definition	One hierarchical training set per category	Greedy search of a single classification path. Single category assignment

Table 4.2: Classification of previous works (2 of 2)

naive Bayes classifiers, built with proper feature sets, seem to be a valid alternative to SVM, especially in those application contexts where a "commission error" is considered more serious than an "omission error".

In this work we have not investigated the possibility of restructuring the original category hierarchy. Vinokourov and Girolami [VG02] proposed a probabilistic mixture model for the hierarchic partition organization of a collection of documents. Sona et al. [SVAP04] address the problem of document clustering where documents are assigned both to the leaves and to internal nodes. An alternative to building the hierarchy from scratch is restructuring a given hierarchy on the basis of some training examples. It can be realized by means of a greedy procedure that adds or removes categories until no further improvements can be made. Hierarchy restructuring can substantially improve the accuracy of the hierarchical approach, which can eventually give better performance than the flat approach.

Another limitation of this work is the consideration of a single-category assignment rather than the more general case of a multi-category assignment. However, the multi-category assignment occurs either when the hierarchy appears to be ill-structured with respect to documents collected over time, or when the same documents can be actually classified along several dimensions. In the former case, the single-category assignment can be kept if the hierarchy is restructured. In the latter case, it would be better to consider a multi-dimensional framework, as that investigated by Theeramunkong and Lertnattee [TL02]. In the future, we intend to extend this work by considering the integration of both the multi-dimensional and the hierarchical frameworks, in order to support WebClass users with OLAP-like roll-up, drill-down and pivoting operations in an information retrieval context.

4.2 Document Image Analysis

The large and increasing amount of paper documents to be processed daily demands for new document management systems with abilities to catalog and organize these documents automatically on the basis of their contents semantics. Personal document processing systems that can provide functional capabilities of classifying, storing, retrieving, and reproducing documents, as well as extracting, browsing, retrieving and synthesizing information from a variety of documents are in ever-growing demand [FSN99]. However, they operate on electronic documents and not on the more common paper documents. This issue is considered in the area of **Document Image Analysis** (DIA), which investigates the theory and practice of recovering the symbol structure of digital images scanned from paper or produced by computer.

The representation of extracted information into some common data format is a key issue. Some general data formats (e.g. DAFS [wis95]) and many ad-hoc formats have been developed for this purpose, but none of them is extensible and general enough to hold for all different situations. This variety of formats prevents the easy exchange of data between different environments. A solution to this problem can come from the XML technology. XML has been proposed as a

data representation format in general, but it was originally developed to represent (semi-) structured documents, therefore it is a natural choice for the representation of the output of DIA systems. XML is also an Internet language, a characteristic that can be profitably exploited to make information present on paper more quickly web-accessible and retrievable than distributing the bitmaps of document images on a web server. Moreover, it is possible to define some hypertext structures which improve document reading [WS99]. Finally, in the XML document, additional information on the semantics of the text can be stored in order to improve the effectiveness of the retrieving. This is a way to reduce the so-called semantic gap in the document retrieving [ZG02], which corresponds to the mismatch between user's request and the way automated search engines try to satisfy these requests.

Commercial OCR systems are still far from supporting the XML format generation satisfactorily. Most of them can save scanned documents in HTML format, but generally their appearance on the browser is not similar to the original documents. Rendering problems, such as missing graphical components, wrong reading ordering in two-columned papers, missing indentation and broken text lines, are basically due to poor layout information extracted from the scanned document. In addition, no information on the semantics of some content portions is associated to documents saved in HTML format. The extraction of semantics from the document image requires knowledge technologies, which offer various solutions to the knowledge representation problem and automated reasoning, as well as to the knowledge acquisition problem by means of machine learning techniques. The importance of knowledge technologies has led some distinguished researchers to claim that document image analysis and understanding belongs to a branch of artificial intelligence [TYS94], despite most of the contributions fall within the area of pattern recognition [Nag00]. In this chapter we present the multi-page DIA system WISDOM++[12] [MCB03] (whose architecture is knowledge-based and supports all processing steps required for semantic indexing and storing in XML format [AEM01] and we show the application of the multi-relational naive Bayesian classifier Mr-SBC in Document Understanding tasks.

4.2.1 Processing Documents

The transformation process performed by WISDOM++ (Figure 4.11) consists of the preprocessing of the raster image of a scanned paper document, the segmentation of the preprocessed raster image into basic layout components, the classification of basic layout components according to the type of content (e.g., text, graphics, etc.), the identification of a more abstract representation of the document layout (layout analysis), the classification of the document on the ground of its layout and content, the identification of semantically relevant layout components, the application of OCR only to those textual components of interest and the storing in XML format providing additional information on the semantic of the text.

Five of these processing steps are knowledge-based (see Figure 4.12), namely:

1. Classification of basic-blocks

[12] http://www.di.uniba.it/∼malerba/wisdom++/

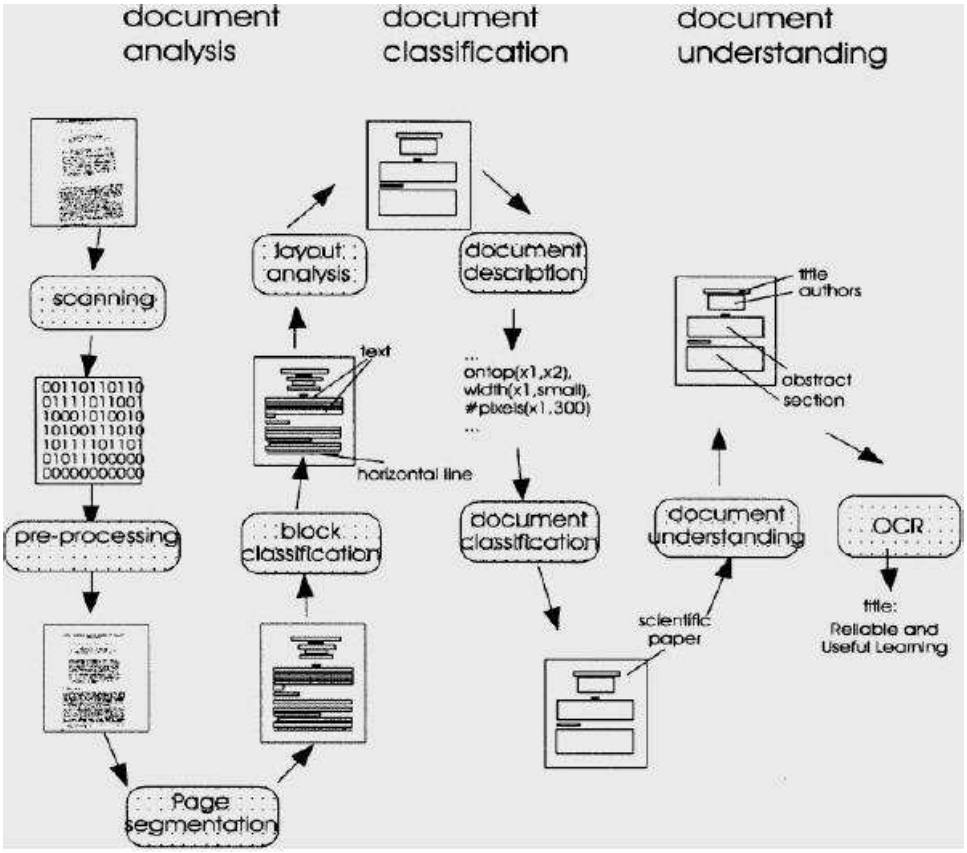

Figure 4.11: WISDOM++ steps

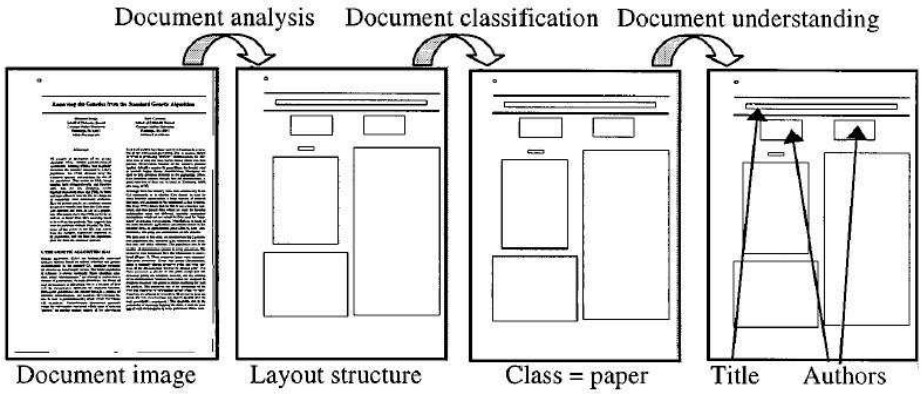

Figure 4.12: Layout Analysis, Document Classification and Document Understanding

2. Layout analysis
3. Automatic global layout analysis correction
4. Semantic indexing - document image classification
5. Semantic indexing - document image understanding

In this section we briefly describe the WISDOM++ principal steps.

Preprocessing

Document preprocessing consists in the evaluation of the skew angle, the rotation of the document, as well as the computation of a spread factor. The skew angle of a document image I is the orientation angle θ of its text baselines. It is positive when the image is rotated anti-clockwise, otherwise it is negative. The evaluation of the skew angle is essential, since for the subsequent step of document segmentation, we use a top-down method, which is quite fast, but generally ineffective when applied to skewed documents. Once the skew angle has been estimated the document image can be corrected by means of an inverse rotation operator.

The estimation $\hat{\theta}$ of the actual skew angle θ is obtained as the composition of two functions: $S(I)$, which returns a *sample region* R of the document image I, and $E(R)$, which returns the *estimation* of the skew angle in the sample region R. The selection of a sample region is peculiar to WISDOM++ and has the advantage of reducing the computational cost of the estimation step, while its main disadvantage is the possibility of errors in the estimation of the *dominant* (i.e., the most frequent) skew in documents with many local skews for text lines.

In order to select the sample region WISDOM++ computes both the horizontal projection profile H of the document image and the average number of pixels per row ($avpx$). Then it extracts a set of *regions* from H: A region R_i is a sequence of adjacent rows in H, whose height is greater than $avpx/4$. In this way, only regions with prominent peaks will be considered, since $E(R_i)$ is more likely to be close to the true skew angle θ. Each region is classified as horizontal line, text, or image as

specified in [AEM99]. Since the focus is on the estimation of the skew angle of text regions, the system selects, if any, the text region R_i with the maximum average density of black pixels per row. Otherwise, the system returns the region, classified as horizontal line or image, satisfying the following conditions: its base is smaller than 310 pixels and it has the maximum average density of black pixels per row.[13]

Once the sample region R has been selected, $E(R)$is computed. Let H_θ be the horizontal projection profile of R after a virtual rotation of an angle θ. The histogram H_θ shows sharply rising peaks with a base equal to the character height when text lines span horizontally, while it presents smooth slopes and lower peaks when the skew angle is large. This observation is mathematically captured by a real-valued function, $A(\theta) = \sum_{j \in R} H_\theta^2(j)$, which has a global maximum at the correct skew angle. Thus, finding the actual skew angle means locating the global maximum value of A(θ). Since this measure is not smooth enough for the application of gradient techniques, the system adopts some peak-finding heuristics. Details of these heuristics are reported in [AEM01].

In the preprocessing phase the *spread factor* of the document image is also computed. It is defined as the ratio of the average distance between the regions R_i (*avdist*) and the average height of the same regions (*avheight*). In quite simple documents with few sparse regions this ratio is greater than 1.0, while in complex documents with closely written text regions the ratio is lower than the unit. The spread factor is used to define some parameters of the segmentation algorithm.

Separation of text from graphics

Wherever the primary goal of the document analysis process is interpretation of text data, graphic data present within the digitized document must be first separated from the text so that subsequent processing stages may operate exclusively on the textual information. The separation of text from graphics is performed into two steps: image segmentation and block classification. The former is the identification of rectangular blocks enclosing content portions while the latter aims at discriminating blocks enclosing text from blocks enclosing graphics (pictures, drawings and horizontal/vertical lines).

WISDOM++ segments the reduced document image into rectangular blocks by means of an efficient variant of the Run Length Smoothing Algorithm (RLSA) [WCW82]. The RLSA applies four operators to the document image:

1. horizontal smoothing with a threshold C_h;
2. vertical smoothing with a threshold C_v;
3. logical AND of the two smoothed images;
4. additional horizontal smoothing with another threshold C_a.

Although it is conceptually simple, this algorithm requires scanning the image four times. WISDOM++ implements a variant that scans the image only twice,

[13]When no full region satisfying these conditions exists, a sub-region of exactly 310 pixel is selected.

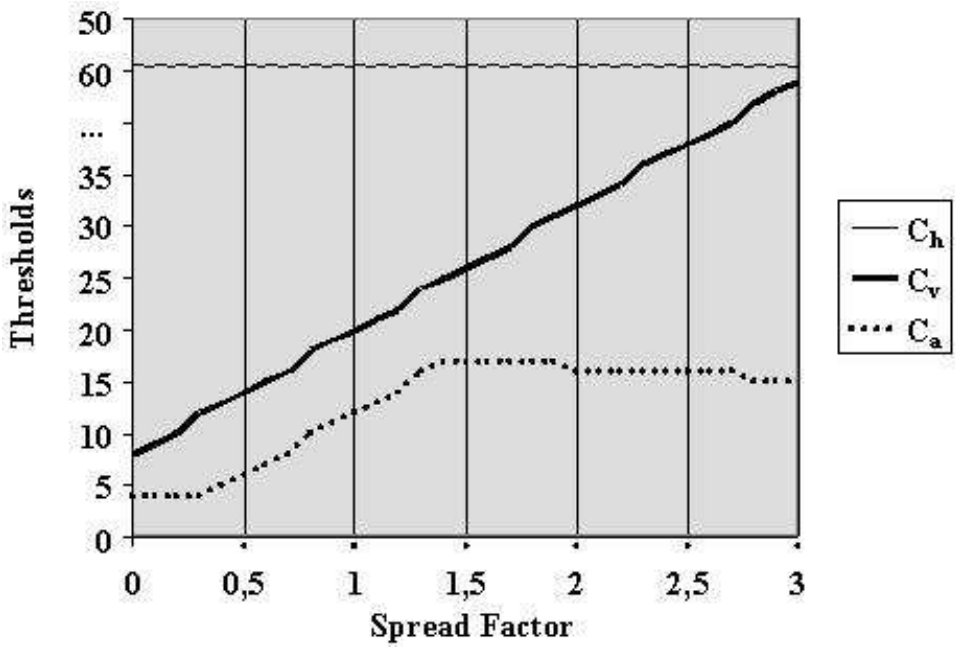

Figure 4.13: Adaptive threshold definition depending on the spread factor.

with no additional cost [SC96]. Furthermore, the smoothing parameters C_v and C_a are adaptively defined depending on the spread factor computed during the skew evaluation process, while C_h is set to one tenth of the number of columns in the reduced bitmap (See Figure 4.13).

The segmentation algorithm returns blocks that may contain either textual or graphical information. In order to facilitate subsequent document processing steps, it is important to classify these blocks according to the type of content: text block, horizontal line, vertical line, picture (i.e., halftone images) and graphics (e.g., line drawings). The classification of blocks is performed by means of a decision tree automatically built from a set of training examples (blocks) of the five classes. The choice of a "treebased" method is due to its inherent flexibility, since decision trees can handle complicated interactions among features and give results that can be easily interpreted.

The numerical features used by the system to describe each block are the following:

- *height*: height of the reduced image block;
- *length*: length of the reduced image block;
- *area*: area of the reduced image block (*height*length*);
- *eccen*: eccentricity of the reduced image block (*length/height*);
- *blackpix*: total number of black pixels in the reduced image block;
- *bw_trans*: total number of black-white transitions in all rows of the reduced image block;

- *pblack*: percentage of black pixels in the reduced image block (*blackpix/area*);
- *mean_tr*: average number of black pixels per black-white transition (*blackpix/bw_trans*);
- $F1$: short run emphasis;
- $F2$: long run emphasis;
- $F3$: extra long run emphasis.[14]

Given a block description consisting of the above eleven features, the tree-based classification system performs a sequence of tests which result in the determination of the type of the block. The sequence of tests can vary from block to block, and the number of different sequences equals the number of leaves in the decision tree. For purposes, the learning algorithm ITI 2.0 [Utg94] is integrated in WISDOM++.

Layout Analysis

The result of the segmentation process is a list of classified blocks, corresponding to printed areas in the page image. Each block is described by a pair of coordinates, namely top left-hand corner and bottom right-hand corner, and the type. The number of blocks is generally less than a hundred; thus, a segmented page is certainly easier to manage than the original bitmap. However, this new page representation is still too detailed for learning rules used in document classification and understanding. The perceptual organization process that aims to detect structures among blocks is called the layout analysis. The result is a hierarchy of abstract representations of the document image, the geometric (or layout) structure. The leaves of the layout tree (lowest level of the abstraction hierarchy) are the blocks, while the root represents the whole document.

In multi-page documents, the root represents a set of pages. A page may group together several layout components, called frames, which are rectangular areas of interest in the document page image. An ideal layout analysis should produce a set of frames, each of which can be associated with a distinct logical component, such as title and author of a scientific paper. In practice, however, a suboptimal layout structure, in which it is still possible to distinguish the logical meaning of distinct frames, should be considered a good output of the layout analyzer.

The various approaches to the extraction of the layout structure can be classified in two distinct dimensions: 1) direction of construction of the layout tree (top-down or bottom-up), and 2) amount of explicit knowledge used during the layout analysis. As to the second dimension, Nagy and his colleagues [NKK+88] distinguish three levels of knowledge in the layout structure of a document:

- Generic knowledge (e.g., type base lines of a word are collinear).
- Class-specific knowledge (e.g., no text line is lateral to a graphical object).
- Publication-specific knowledge (e.g., maximum type size is 22 points).

[14] Computed using the following thresholds: T1=10 and T2=20 [WS89].

They observe that knowledge used in bottom-up layout analysis is necessarily different from that used for top-down processing: it is much less document specific. In addition, we note that knowledge used in top-down approaches is typically derived from the relations between the geometric and the logical structures of specific classes of documents.

In Wisdom++, the applied page decomposition method is hybrid, since it combines a variant of the RLSA to segment the document image and a bottom-up layout analysis method to assemble basic blocks into larger components called frames.

More precisely, the layout analysis is done in two steps:

1. A global analysis of the document image in order to determine possible areas containing paragraphs, sections, columns, figures and tables. This step is based on an iterative process, in which the vertical and horizontal histograms of text blocks are alternatively analyzed in order to detect columns and sections/paragraphs, respectively.

2. A local analysis of the document to group together blocks which possibly fall within the same area. Three perceptual criteria are considered in this step: proximity (e.g. adjacent components belonging to the same column/area are equally spaced), continuity (e.g. overlapping components) and similarity (e.g. components of the same type, with an almost equal height).

Pairs of layout components that satisfy some of these criteria may be grouped together. Each layout component is associated with one of the following types: text, horizontal line, vertical line, picture, graphic and mixed. When the constituent blocks of a logical component are homogeneous, the same type is inherited by the logical component; otherwise, the associated type is set to mixed. The layout structure extracted by WISDOM++ is a hierarchy with six levels: basic blocks, lines, set of lines, frame1, frame2, pages.

Experimental results proved the effectiveness of this knowledge-based approach on images of the first page of papers published in conference proceedings and journals [AEM01]. However, performance degenerates when the system is tested on intermediate pages of multi-page articles, where the structure is much more variable, due to the presence of formulae, images, and drawings that can stretch over more than one column, or are quite close. The majority of errors made by the layout analysis module were in the global analysis step, while the local analysis step performed satisfactorily when the result of the global analysis was correct.

To avoid this problem, WISDOM++ supports the user during the correction of the results of the global analysis. This is done by allowing the user to correct the results of the global analysis and then by learning rules for layout correction from his/her sequence of actions [BCEM03] [MEA+03]

Global analysis aims to determine the general layout structure of a page and operates on a tree-based representation of nested columns and sections. The levels of columns and sections are alternated 4.14, which means that a column contains sections, while a section contains columns. At the end of the global analysis, the user can only see the sections and columns that have been considered atomic, that

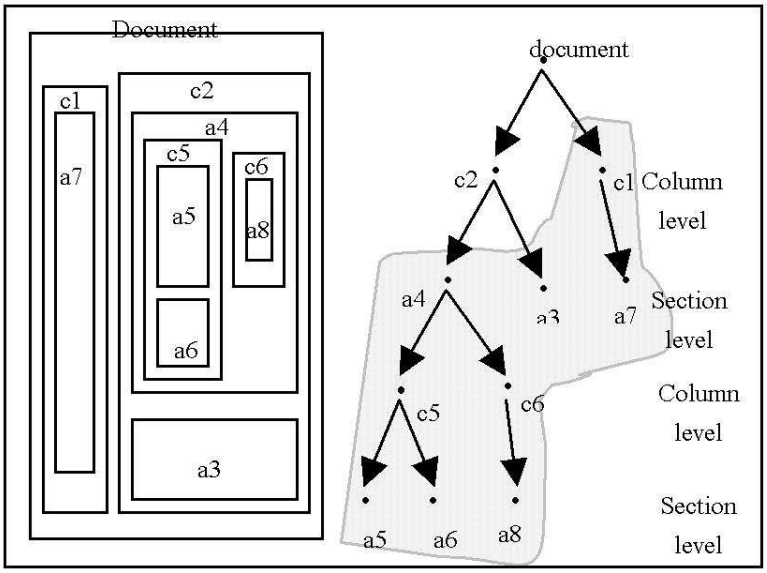

Figure 4.14: Layout tree. Columns and sections are alternated.

is, not subject to further decomposition. The user can correct this result by means of three different operations:

- Horizontal splitting: a column/section is cut horizontally.
- Vertical splitting: a column/section is cut vertically.
- Grouping: two sections/columns are merged together.

After each splitting/grouping operation, WISDOM++ recomputes the result of the local analysis process, so that the user can immediately perceive the final effect of the requested correction and can decide whether to confirm the correction or not.

Rules for the automated correction of the layout analysis can be automatically learned by means of the learning system ATRE [Mal03]. The learning problem solved by ATRE can be formulated as follows:

Given

- a set of concepts $C_1, C_2, .., C_r$ to be learned,
- a set of training observations O described in a language LO,
- a user's preference criterion PC,

Find a (possibly recursive) logical theory T for the concepts $C_1, C_2, \ldots, C_r$ such that T is complete and consistent with respect to O and satisfies the preference criterion PC. In the context of the global analysis correction, the set of concepts to be learned are split(X)=horizontal, split(X)=vertical, group(X,Y)=true, since we are interested to find rules predicting both when to split horizontally/vertically a columns/section and when to group two columns/section. No rule is generated for the case split(X)=no_split and group(X)=false. The preference criterion PC is a

set of conditions used to discard some solutions and favor others. In particular, we prefer short rules that explain a high number of positive examples and a low number of negative examples.

In practice ATRE learns operations that are expressed as a set of "production" rules in the form of an antecedent and a consequent, where the antecedent expresses the precondition to the application of the rule and the consequent expresses the action to be performed in order to modify the layout structure.

Document Classification

After having detected the layout structure, the logical components of the document, such as title, authors, sections of a paper, can be identified. The logical components can be arranged in another hierarchical structure, which is called logical structure. The logical structure is the result of repeatedly dividing the content of a document into increasingly smaller parts, on the basis of the human-perceptible meaning of the content. The leaves of the logical structure are the basic logical components, such as authors and title. The heading of an article encompasses the title and the author and is therefore an example of composite logical component. Composite logical components are internal nodes of the logical structure. The root of the logical structure is the document class (e.g. "scientific paper", "letter" or "censorship card"). WISDOM++ supports two-level logical structures, where the document class is the only composite logical component.

The problem of finding the logical structure of a document can be cast as the problem of associating some layout components with a correspondent logical component. In WISDOM++ this mapping is limited to the association of a page with a document class (document classification) [EMS+90] and the association of second frames with basic logical components (document understanding) [TA90].

Classification of multi-page documents is performed by matching the layout structure of the first page against models of classes of documents. These models capture the invariant properties of the images/layout structures of documents belonging to the same class. They are rules expressed in a first-order logic language, so that the document classification problem can be reformulated as a matching test between a logic formula that describes a model and another logic formula that represents the image/layout properties of the first page. The choice of a first-order logic language answers to the requirement of flexibility and generality. In this language unary function symbols, called attributes, are used to describe properties of a single layout component (e.g. height and length), while binary predicate and function symbols, called relations, are used to express spatial relationships between layout components. A complete list of attributes and relations is reported in Table 4.3. A partial description of the page layout of the document in Figure 4.15 follows:

```
image_lenght(1)=3468, image_width(1)=2418,
part_of(1,2)=true, part_of(1,3)=true, ..., part_of(1,25)=true,
width(2)=15, width(3)=20, ..., width(25)=429,
height(2)=239, height(3)=4, ..., height(25)=24,
```

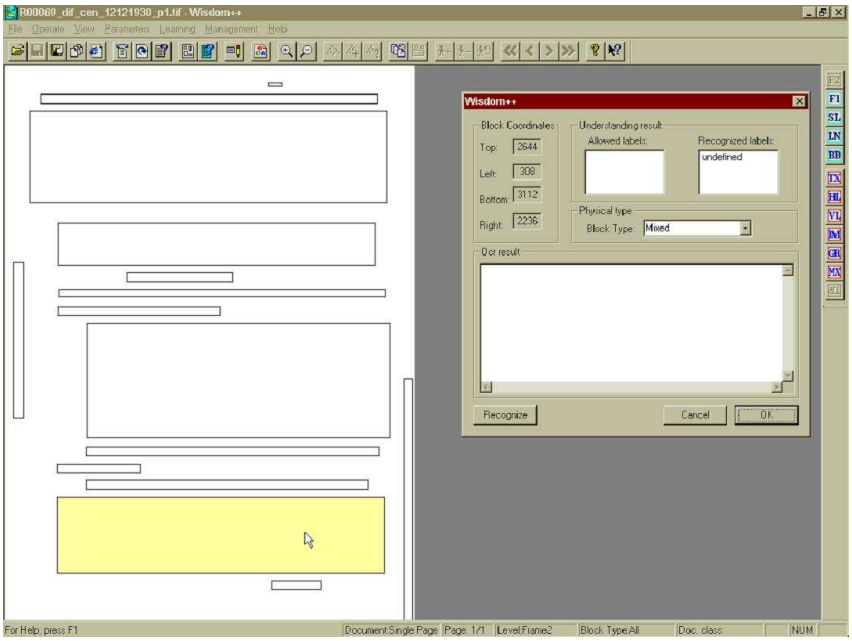

Figure 4.15: Example of layout components at the Frame 2 level

```
type_of(2)=text, type_of(3)=text, ..., type_of(25)=text,
x_pos_centre(2)=20, x_pos_centre(3)=398, ..., x_pos_centre(25)=334,
y_pos_centre(2)=420, y_pos_centre(3)=28, ..., y_pos_centre(25)=558,
on_top(3,9)=true, on_top(9,8)=true, ..., on_top(19,20)=true,
to_right(2,11)=true, to_right(2,15)=true, ..., to_right(25,5)=true,
alignment(3,13)=only_right_col, alignment(9,12)=only_right_col, ...,
   alignment(7,8)=only_upper_row.
```

The constant 1 denotes the whole page, while the constants 2, 3, ..., 25 denote the layout components at the frame2 level. Indeed, in order to reduce the computational complexity of the classification problem, WISDOM++ restricts the description of the document to the properties of the frame2 layout components alone. The description is a logical conjunction of literals of the form:

$$f(t_1, ..., t_n) = Value$$

where f is an n-ary function symbol, that is an attribute or relation, t_i's are constant terms, and Value is one of the possible values of f's domain.

An example of model defined by a single rule is the following:

```
class(X1)=dif_cen_decision
   part_of(X1,X2)=true,
   y_pos_centre(X2) ∈ [754 ..  841],
   alignment(X2,X3)=only_left_col.
```

Attribute/relation name	**Extracted from**	**Definition**
image_length(doc)	Image	Integer domain: (1 .. 5000)
image_width(doc)	Image	Integer domain : (1 .. 4000)
width(block)	Page layout	Integer domain: (1..640)
height(block)	Page layout	Integer domain: (1..890)
x_pos_centre(block)	Page layout	Integer domain: (1..640)
y_pos_centre(block)	Page layout	Integer domain: (1..875)
type_of(block)	Page layout	Nominal domain: text, hor_line, image, ver_line, graphic, mixed
part_of(page,block)	Page layout	Boolean domain: true if page contains block
on_top(block1,block2)	Page layout	Boolean domain: true if block1 is above block2
to_right(block1,block2)	Page layout	Boolean domain: true if block2 is to the right of block1
alignment(block1,block2)	Page layout	Nominal domain: only_left_col, only_right_col, only_middle_col, both_columns, only_upper_row, only_lower_row, only_middle_row, both_rows

Table 4.3: Attributes and relations used to describe both the models and the documents to be classified

where X1 is a variable denoting the whole page, while the remaining variables X2 and X3 denote two layout components at the frame2 level.

The learning system embedded in WISDOM++ for document classification is ATRE [Mal03]. The learning problem solved by ATRE can be formulated as follows:

Given

- a set of concepts $C_1, C_2, \ldots, C_r$ to be learned,
- a set of observations O described in a language L_O,
- a background knowledge BK described in a language L_{BK},
- a language of hypotheses L_H,
- a generalization model Γ over the space of hypotheses,
- a user's preference criterion PC,

Find a (possibly recursive) logical theory T for the concepts $C_1, C_2, \ldots, C_r$, such that T is complete and consistent with respect to O and satisfies the preference criterion PC.

As to the representation languages, the basic component is the *literal* in the two distinct forms:

$f(t_1, \ldots, t_n) = Value$ (*simple literal*) $f(t_1, \ldots, t_n) \in Range$ (*set literal*),

where f and g are function symbols called *descriptors*, t_i's and s_i's are terms, and *Range* is a closed interval of possible values taken by f. Some examples of literals are the following: *color*(X_1)=*red*, *height*(X_1) $\in$ *[1.1 .. 1.2]*, and *on_top(X, Y)=true*.

Document Image Understanding

In document image understanding, layout components are associated with logical components. This association can theoretically affect layout components at any level in the layout hierarchy. However, in WISDOM++ only frame2 components are associated with some component of the logical hierarchy. Moreover, only layout information is used in document image understanding. This approach differs from that proposed by other authors [KDK00] which additionally make use of textual information (e.g. text pattern), font information (e.g. style, size, boldness, etc.) and universal attributes (e.g. number of lines) given by the OCR. This diversity is due to a different conviction on when an OCR should be applied. We believe that only some layout components of interest for the application should be subject to OCR (e.g., title and authors, but not figures and tables of a scientific paper), hence document understanding should precede text reading and cannot be based on textual features. Two basic assumptions are made:

1. Documents belonging to the same class have a set of relevant and invariant layout characteristics (page layout signature).
2. It is possible to identify logical components by using only layout information.

In section 4.2.3 we will explain how we integrate WISDOM++ with Mr-SBC to solve the document Understanding problem.

Generating a document in XML format

Data concerning the result of document processing can be stored in XML format so that the resulting XML document, which includes semantic information extracted in the document analysis and understanding processes, is accessible via web through queries at a high level of abstraction.

The simplest transformation consists in attaching document images to XML pages, after having converted bitmaps into a format supported by most browsers (e.g. GIF or JPEG). Nevertheless, this approach presents at least four disadvantages. First, compressed raster images are still quite large and their transfer can be unacceptably slow. Second, the original document can only be viewed and not edited. Third, in the case of multi-page documents, pages can be presented only in a sequential order, thus missing the advantages a hypertext structure which supports document browsing. Fourth, additional information about the semantics of the content cannot be represented, hence no semantics-based retrieval facility can be supported. Therefore, it is important to transform document images into XML format by integrating textual, graphical, layout and semantic information extracted in the document analysis and understanding processes. Moreover, the XML specification includes a facility for physically isolating and separately storing any part of a document, for example, storing data without contamination of formatting information.

A DTD is associated to each document class and the XML document refers to the appropriate DTD. In the following, an example of a DTD generated by WISDOM++ for the class "tpami" is reported.

```
<!– standard DTD file for tpami class –>
<!ELEMENT tpami (logic-structure?, geometric-structure)>
<!ELEMENT logic-structure (undefined |affiliation |page-number |figure |caption
|index-term |running-head |author|title |abstract |biografy|references |paragraph |section-
title |subsection-title)*>
<!ELEMENT undefined (paragraph)*>
<!ATTLIST undefined ID NMTOKEN #IMPLIED>
<!ELEMENT affiliation (paragraph)*>
<!ATTLIST affiliation ID NMTOKEN #IMPLIED>
<!ELEMENT page-number (paragraph)*>
<!ATTLIST page-number ID NMTOKEN #IMPLIED>
<!ELEMENT figure (paragraph)*>
<!ATTLIST figure ID NMTOKEN #IMPLIED>
<!ELEMENT caption (paragraph)*>
<!ATTLIST caption ID NMTOKEN #IMPLIED>
<!ELEMENT index-term (paragraph)*>
<!ATTLIST index-term ID NMTOKEN #IMPLIED>
<!ELEMENT running-head (paragraph)*>
<!ATTLIST running-head ID NMTOKEN #IMPLIED>
<!ELEMENT author (paragraph)*>
```

```
<!ATTLIST author ID NMTOKEN #IMPLIED>
<!ELEMENT title (paragraph)*>
<!ATTLIST title ID NMTOKEN #IMPLIED>
<!ELEMENT abstract (paragraph)*>
<!ATTLIST abstract ID NMTOKEN #IMPLIED>
<!ELEMENT biografy (paragraph)*>
<!ATTLIST biografy ID NMTOKEN #IMPLIED>
<!ELEMENT references (paragraph)*>
<!ATTLIST references ID NMTOKEN #IMPLIED>
<!ELEMENT paragraph (paragraph)*>
<!ATTLIST paragraph ID NMTOKEN #IMPLIED>
<!ELEMENT section-title (paragraph)*>
<!ATTLIST section-title ID NMTOKEN #IMPLIED>
<!ELEMENT subsection-title (paragraph)*>
<!ATTLIST subsection-title ID NMTOKEN #IMPLIED>
<!ELEMENT paragraph (#PCDATA|TAB)*>
<!ELEMENT TAB EMPTY>
<!ELEMENT geometric-structure (image, blocklevels)>
<!ELEMENT image EMPTY>
<!ATTLIST image urlimage CDATA #REQUIRED
   length NMTOKEN #REQUIRED
   width NMTOKEN #REQUIRED
   formatimage NMTOKEN #REQUIRED
   resolution NMTOKEN #REQUIRED>
<!ELEMENT blocklevels (basic-block, line, setofline, frame1, frame2)>
<!ELEMENT basic-block (block+)>
<!ELEMENT line (block+)>
<!ELEMENT setofline (block+)>
<!ELEMENT frame1 (block+)>
<!ELEMENT frame2 (block+)>
<!ATTLIST basic-block numBB NMTOKEN #REQUIRED>
<!ATTLIST line numL NMTOKEN #REQUIRED>
<!ATTLIST setofline numSL NMTOKEN #REQUIRED>
<!ATTLIST frame1 numF1 NMTOKEN #REQUIRED>
<!ATTLIST frame2 numF2 NMTOKEN #REQUIRED>
<!ELEMENT block EMPTY>
<!ATTLIST block indexblock NMTOKEN #REQUIRED
   top NMTOKEN #REQUIRED
   bottom NMTOKEN #REQUIRED
   left NMTOKEN #REQUIRED
   right NMTOKEN #REQUIRED
   physical-type NMTOKEN #REQUIRED
   subblockslist CDATA #IMPLIED
```

label (undefined|affiliation |page-number |figure |caption |index-term |running-head |author |title |abstract |biografy |references |paragraph |section-title |subsection-title) "undefined" >

The keyword ELEMENT introduces an element declaration which represents the information on the semantics of the content (e.g. affiliation, page-number, figure, caption, index-term, running-head, author, title, abstract, biografy, references, paragraph, section-title, subsection-title, undefined[15]). An element may have no content at all, may have a content of only text, of only child element, or of a mixture of elements and text. For example, in the DTD presented the content of the element tpami is a child element, which is structured. An attribute may be associated with a particular element in order to provide refined information on an element. Examples of attributes are the URL, the height, the width, the format and the resolution of a document image. All the attributes are declared separately from the element, but are usually declared together, in the attribute list declaration. It is also noteworthy that the DTD generated by WISDOM++ distinguishes the logical structure (logic-structure) from the layout structure (geometric-structure). The layout structure is used for storing purposes, in particular it is used to build XSL specifications in order to render the document similar in appearance to the original document, since XML language is not concerned with visualization aspects.

The XML document generated can be stored in an XML-based Content Management System (XMLCM), which is the back-end of WISDOM++. XMLCM uses the XML language to represent/manage documents, structured data and metadata (DTD or XML Schema) and to exchange them over Internet. Because Internet-based applications deal with complex, heterogeneous and worldwide information, the XMLCM is based on basic open communication standards for information processing, such as HTTP, XML and SOAP.

4.2.2 Wisdom++ architecture

The general architecture of WISDOM++, shown in 4.16, integrates several components to perform all the steps reported in the previous section.

The System Manager manages the system by allowing user interaction and by coordinating the activity of all other components. It interfaces the system with the data base module in order to store intermediate information. The System Manager is also able to invoke the OCR on textual layout blocks which are relevant for the specific application (e.g., title or authors).

The *Image Processing Module* is in charge of the image preprocessing facilities and is able to perform a series of image-to-image transformations. The *Layout Analysis Module* is in charge of the separation of text from graphics and the layout analysis. It interfaces the ITI decision tree learner for basic block classification. The *Production System for Layout Correction Module* is able to use production rules extracted by ATRE, it operates with a forward-chaining control structure.

[15]The element undefined refers to all those logical components of no specific interest for the application.

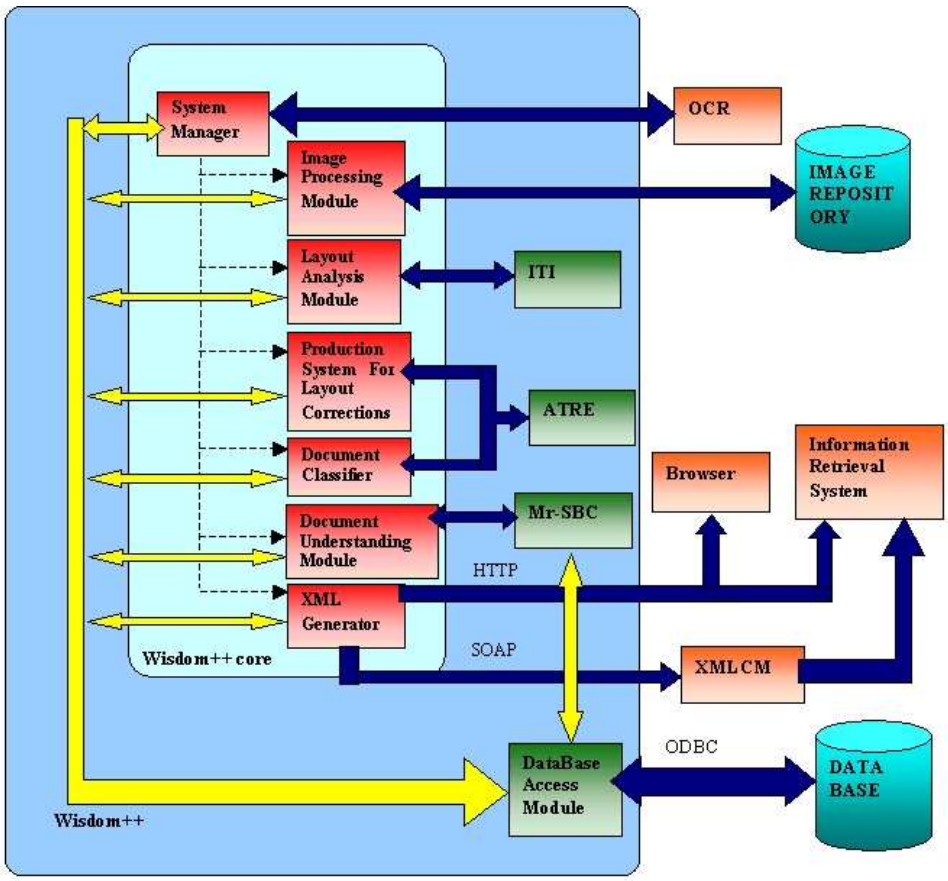

Figure 4.16: WISDOM++ architecture

The production system is implemented with a theorem prover, using resolution to do forward chaining over a full first-order knowledge base. The system maintains a knowledge base (the working memory) of ground literals describing the layout tree. Ground literals are automatically generated by WISDOM++ after the execution of an operation. In each cycle, the system computes the subset of rules whose condition part is satisfied by the current contents of the working memory (match phase). Conflicts are solved by selecting the first rule in the subset. The Production System for Layout Correction Module returns a hierarchy of abstract representations of the document image, the geometric (or layout) structure, which can be modeled by a layout tree. The *Document Classification Module* is in charge of the document classification. It interfaces the system ATRE for the learning phase. The *Document understanding Module* is in charge of the document understanding phase and interfaces the system Mr-SBC that is able to induce a statistical classification model on the basis of the document descriptions. Mr-SBC aims at automatically associating some layout components with components of a logical hierarchy. After document classification and understanding, WISDOM++ actually replaces the low-level image feature space (based on geometrical and textural features) with a higher-level semantic space. Query formulation can then be performed using these higher level semantics, which are much more comprehensible to the user than the low level image features [Bra00]. Finally, the XML Generator Module is used to save the document in XML format. It transforms document images into XML format by integrating textual, graphical, layout and logical information extracted in the document analysis and understanding processes.

4.2.3 Naive Bayes Multi-relational Classification in Document Image Understanding

In the proposed framework, WISDOM++ makes use of Mr-SBC for the Document Image Understanding process in order to recognize and classify significant logical components in the processed documents.

Mr-SBC fits well for the task in hand for three important reasons: First, the preprocessing step is straightforward and simply consists in creating a Database schema by isolating and transforming interesting relational tables that are already stored in the WISDOM++ database. This is performed by means of SQL views. The preprocessing would be much more complicate in the case of systems that work on a set of main-memory Prolog facts. In fact, facts correspond to tuples stored on relational databases, some pre-processing is required in order to transform tuples into facts. Anyway, this has some disadvantages. First, only part of the original hypothesis space implicitly defined by foreign key constraints can be represented after some pre-processing. Second, much of the pre-processing may be unnecessary, since a part of the hypothesis described by Prolog facts space may never be explored, perhaps because of early pruning. Third, in applications where data can frequently change, pre-processing has to be frequently repeated. Finally, database schemas provide the learning system free of charge with useful knowledge of data model that can help to guide the search process. This is an alternative to asking the users to

specify a language bias.

Second, in the Document Image Understanding problem, the spatial location of layout components should be taken into account. This because the model should capture some invariant aspects related to both spatial properties of layout components and spatial relations among components (directional and topological). These aspects are implicitly defined by the relative positioning of spatial objects with respect to some reference system. Modeling these spatial relations is a key challenge in classification problems that arise in spatial domains [SSV+02] and Mr-SBC is able to deal with such relations.

Third, we use a statistical classifier that returns, in addition to the prediction, the confidence of the classification. On the contrary, in some ILP systems the result is a categorical output which convey no information on the potential uncertainty in classification. Small changes in the attribute values of an object being classified may result in sudden and inappropriate changes to the assigned class. Missing or imprecise information may prevent a new object from being classified at all. This is an important aspect in Document Image Understanding, where data often present irregularities and noise due to the scanning procedure, to the format of the document, to ink specks and so on.

Mr-SBC for Document Understanding

Although Mr-SBC can be used to solve the Document Understanding problem, some modifications are necessary. In particular, it is necessary to modify the search strategy in order to allow acyclic paths. As observed by Taskar and his colleagues [TAK02], the acyclicity constraint hinders representation of many important relational dependencies, so decreasing in flexibility. This is particularly true in the task in hand, where a relation between two logical components is modeled by means of a relational table that expresses the existence of the topological relation. For example, suppose we need to model the relation *on_top* between two blocks, in a database point of view, this is realized by means of the table block and a table on_top containing two foreign keys to the table block (see figure 4.17). The referenced blocks are considered one on top the other. In the original formulation of the problem solved by Mr-SBC, it is not possible to meet the same table in a foreign key path (see section 3.3.1) so, it is not possible to take into account the topological relation. To avoid this problem, we modified the definition of foreign key path, allowing cyclic paths:

Definition 4.1 *A **foreign key path** is an ordered sequence of tables* $\vartheta = (T_{i_1}, T_{i_2}, \ldots, T_{i_s})$*, where*

- $\forall j = 1, ..., s, T_{i_j} \in T$
- $\forall j = 1, ..., s-1, T_{i_{j+1}}$ *has a foreign key to the table* T_{i_j} *or* T_{i_j} *has a foreign key to the table* $T_{i_{j+1}}$

where T *is the set of tables of a relational database.*

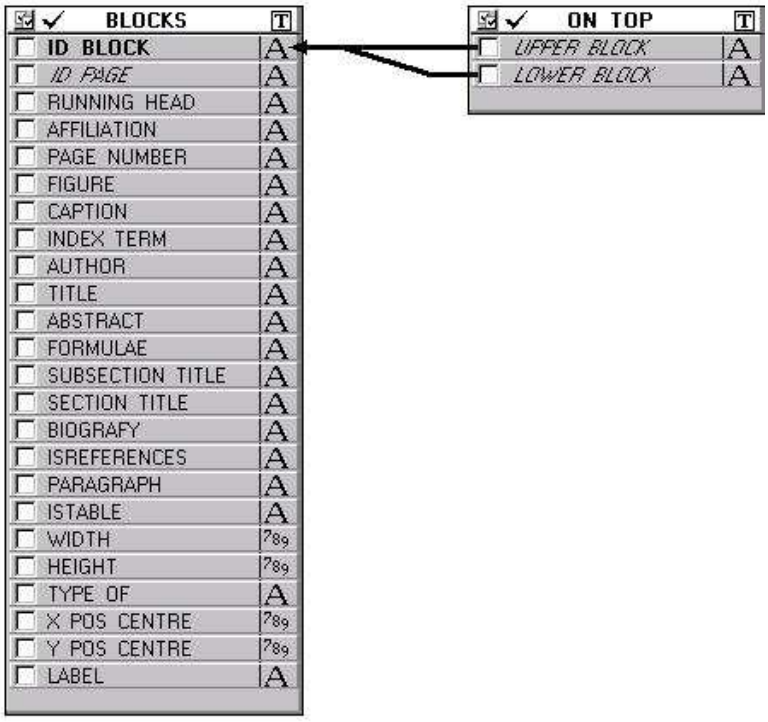

Figure 4.17: Modeling a topological relation

The second problem concerns the concepts to be learned. In Document Understanding, it is possible that the same layout component is associated to two different logical labels. For example, suppose that the layout analysis is not able to separate the page number and the running head of a scientific paper. In this case we have a single layout component that contains two logical components: the page number and the running head. In this case the classifier should associate that component with two labels. For this reason, it is necessary to resort to a multi-classification problem. In particular, we adopted the following solution: we learned a binary classifier for each class, each classifier is able to identify examples belonging to that class and examples that do not belong to it. This solution is usually adopted in text Categorization when the problem is to establish if a document belongs to a particular class or not [Seb02]. It is noteworthy that in the learning step we independently train one classifier at a time because Mr-SBC is not able to learn multiple concepts in parallel.

The use of multiple classification leads to another problem: the unbalanced datasets. In fact, data can be characterized by a predominant number of negative examples with respect to the number of positive examples. Several approaches that face the problem of the unbalanced datasets have been proposed in the literature. Some of them are based on a sampling of examples in order to have a balanced dataset [MG99]. Other approaches are based on a different idea: given the class, a ranking of all the examples in the test set from the most probable member to the least probable member is computed and then, a correctly calibrated estimate

of the true probability that each test example is a member of the class of interest is computed [ZE01]. In other words, a probability threshold that delimitates the membership and the non-membership of a given test example to the class is computed. In our approach, we exploit the consideration that the naive Bayesian classifier for two-class problems tends to rank examples well (even if the classifier does not return a correct probability estimation)[ZE01](section 2.2.3). Our threshold is determined by maximizing the AUC (Area Under the ROC Curve) [PF01] [LF03b]. The ROC curve is defined in the ROC space that denotes the coordinate system used for visualizing classifier performance. In ROC space, TP (True Positive rate) is represented on the Y axis and FP (False Positive Rate) is represented on the X axis. Each classifier is represented by the point in ROC space corresponding to its (FP; TP) pair. For models that produce a continuous output, e.g., posterior probabilities, (such as naive Bayesian) TP and FP vary together as a threshold on the output is varied between its extremes (each threshold defines a classifier); the resulting curve is called the ROC curve.

The expected cost of applying the classifier represented by a point (FP,TP) in ROC space is:

$$cost = P(C_i) \cdot (1 - TP) \cdot c(\neg C_i; C_i) + P(\neg C_i) \cdot FP \cdot c(C_i; \neg C_i) \tag{4.15}$$

where $P(C_i)$ is the a-priori probability that an example belongs to the class C_i, $P(\neg C_i)$ is the a-priori probability that an example does not belong to the class C_i, $c(\neg C_i; C_i)$ is the cost of classifying a positive example as negative (for the class C_i) and $c(C_i; \neg C_i)$ is the cost of classifying a negative example as positive. We denote as $CostRatio$ the value:

$$CostRatio = \frac{c(C_i; \neg C_i)}{c(\neg C_i; C_i)} \tag{4.16}$$

Use and integration

WISDOM++ allows users, in the training phase, to manually label layout components by means of a user interface. The user can assign one or more label to a frame2 component, user's labels are automatically stored in the WISDOM++ Database. In figure 4.18, a system interface snapshot is shown. In particular, it represents the processing phase of the first page of a paper appeared in IEEE Transactions on Pattern Analysis and Machine Intelligence. The document has been acquired by a scanner and has been processed as specified in section 4.2.1.

Once the manual labeling has been completed, the user can run the learner. Mr-SBC is activated by WISDOM++ and operates on data stored in the WISDOM++ Database. When Mr-SBC completes the learning, the classification model is stored in the filesystem and WISDOM++ can use it to automatically recognize layout components of a new testing document.

The Mr-SBC database input schema (see figure 4.19) represents the logical structure of a document image. In particular, we represent both locational features, geometrical features, topological features and aspatial features:

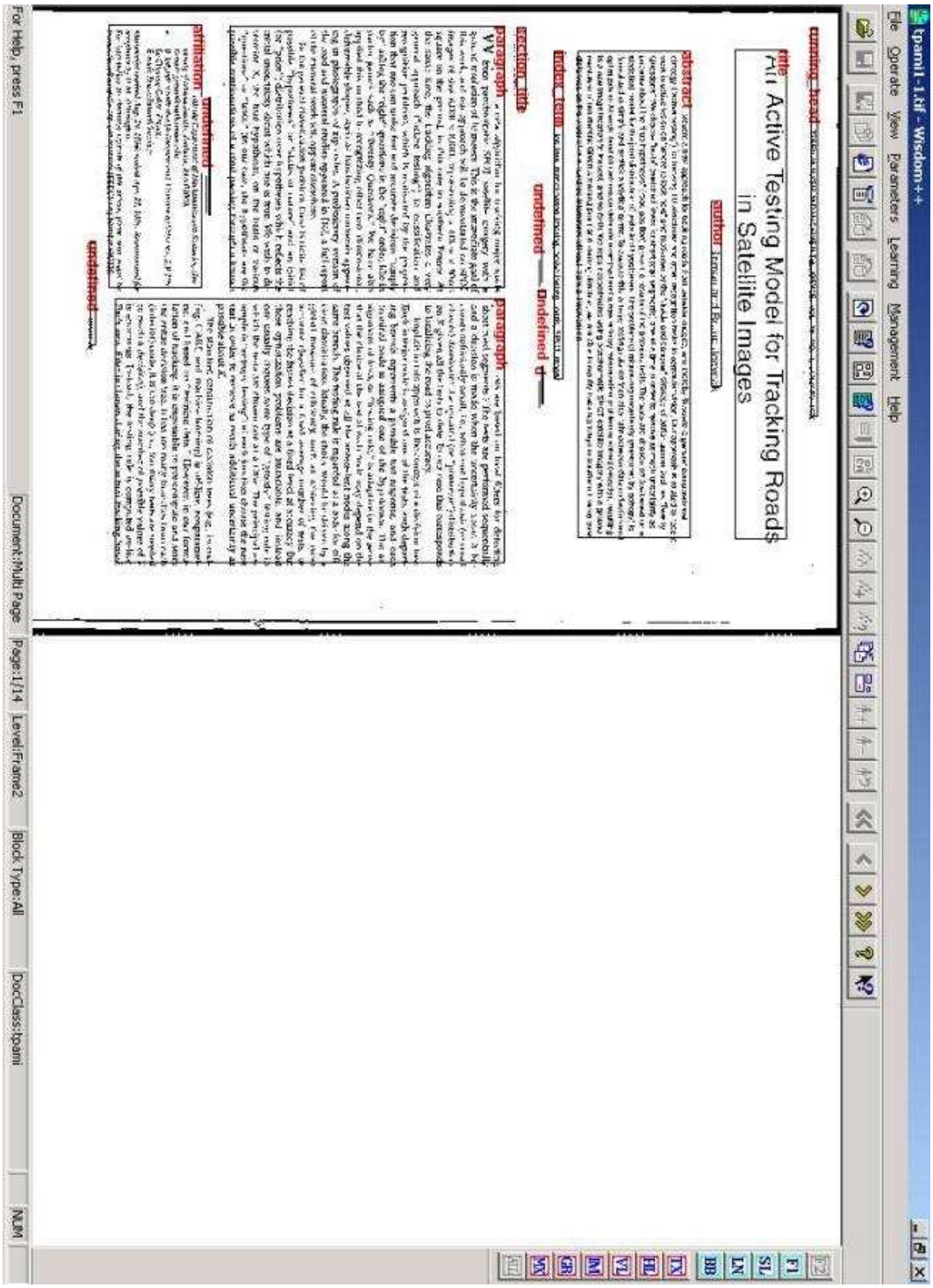

Figure 4.18: Training the system

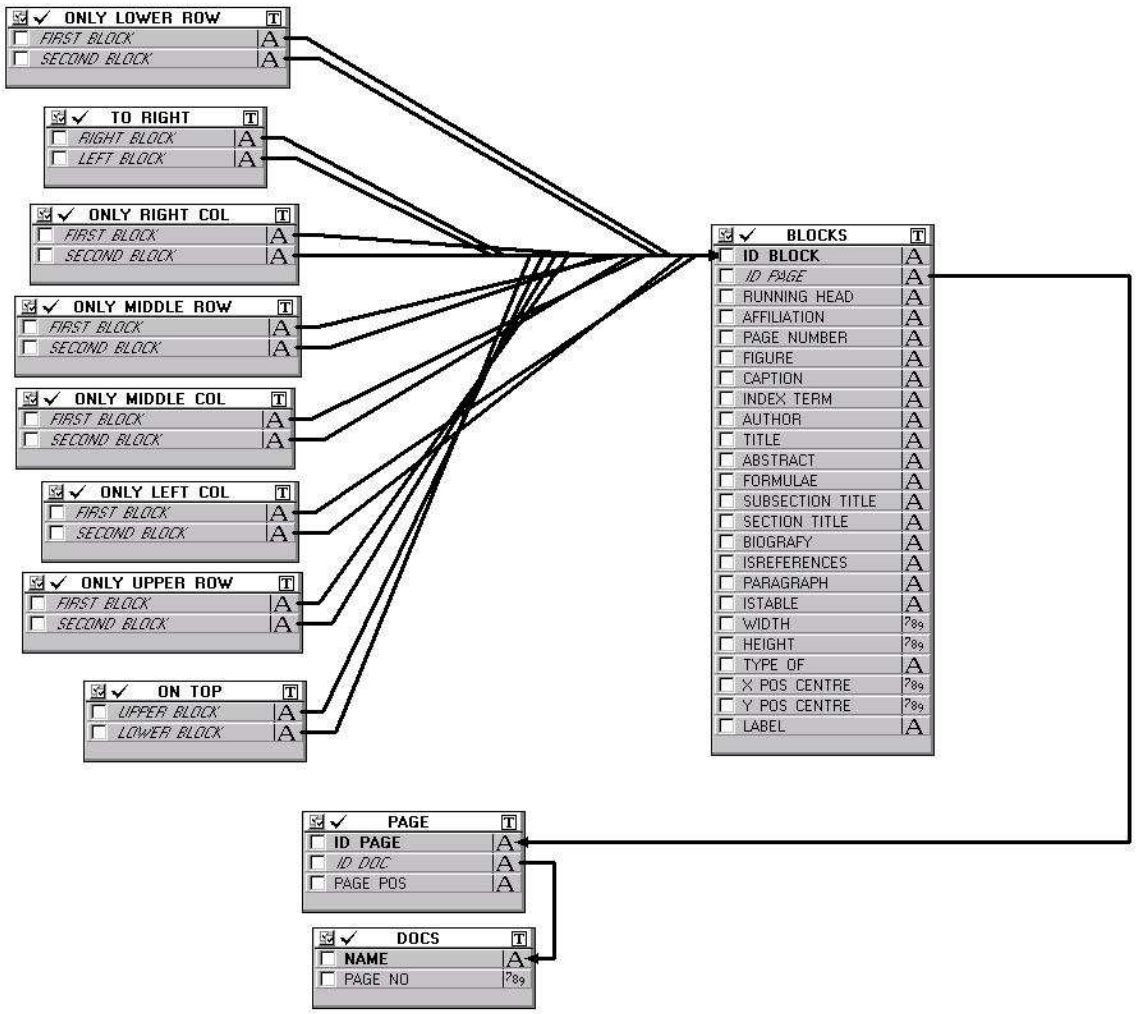

Figure 4.19: Mr-SBC Database input schema

- Locational features identify the position of the component with respect to a coordinate system. They are: *x_pos_center* and *y_pos_center* and represent the coordinates of the centroid along x/y axis of a logical component.

- Geometrical features are *width* and *height* and represent the dimensions of a logical component.

- Topological features represent relations between two components. They are: *on_top* and *to_right*, that define locational relations; in addition we use: *only_right_col*, *only_middle_row*, *only_lower_row*, *only_middle_col*, *only_left_col* and *only_upper_row* that define the alignment of components.

- We also use the aspatial feature *type_of* that specifies the content type of a logical component (e.g. image, text, horizontal line).

4.2.4 Experimental Results

Input Data Description

To investigate the applicability of Mr-SBC in Document Understanding, we considered twenty-one papers, published as either regular or short, in the IEEE Transactions on Pattern Analysis and Machine Intelligence, in the January and February issues of 1996. Each paper is a multi-page document; therefore, we processed 197 document images in all. For 197 document images the user manually labeled layout components.

The total number of labeled components is 2436, that is, in average, 116 components per document, 12.37 per page. About 74% of frame2 layout components have been labeled. The remaining components are considered as "irrelevant" for the task in hand or are considered as "noise". They are automatically considered *undefined*. A description of the dataset is reported in table 4.4.

The performance of the learning task is evaluated by means of a 5-fold cross-validation, that is, the set of twenty-one documents is first divided into five blocks (or folds) 4.4, and then, for every block, Mr-SBC is trained on the remaining blocks and tested on the hold-out block. For each fold, Mr-SBC is trained 16 times, i.e. one for each concept to be learned.

In table 4.5, the set of concepts is reported. The table also reports the average number of positive examples and negative examples for each learning problem. The unbalanced nature of datasets confirms the need of a thresholding procedure (described in section 4.2.3).

The dataset is analyzed by varying the *CostRatio* value in the set of values $\{1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$. Mr-SBC has been executed with the following parameters: MAX_LEN_PATH=4 and MAX_GAIN= 0.1.

Results

For each trial, several measures have been recorded: *accuracy*, *precision* and *recall*, and the number of *omission* and *commission* errors. .

The first measure is the standard *accuracy* defined in machine learning to evaluate the performances of *1-of-r* classifiers.

The precision for a category C_i, denoted as $precision(C_i)$, measures the percentage of correct assignments among all the documents assigned to C_i, while the measure $recall(C_i)$ gives the percentage of correct assignments in C_i among all the documents that should be assigned to C_i. For the whole category space, say $C_1, \ldots, C_L$, the $\mu - AVG$ (micro average) of precision and recall are defined as follows:

$$\mu AVG - precision = \frac{\sum_{i=1}^{L} TP(C_i)}{\sum_{i=1}^{L} (TP(C_i) + FP(C_i)} \tag{4.17}$$

	Name of the multi-page document	*No. of pages*	*No. of labeled components*	*Tot No. of components*
Fold No. 1	TPAMI1	13	476	597
	TPAMI13	3		
	TPAMI14	10		
	TPAMI16	14		
	Total	40		
Fold No. 2	TPAMI8	5	519	684
	TPAMI15	15		
	TPAMI18	10		
	TPAMI24	6		
	Total	36		
Fold No. 3	TPAMI3	15	481	697
	TPAMI7	6		
	TPAMI12	6		
	TPAMI20	14		
	Total	41		
Fold No. 4	TPAMI9	5	541	774
	TPAMI11	6		
	TPAMI119	20		
	TPAMI21	11		
	Total	42		
Fold No. 5	TPAMI4	14	419	549
	TPAMI6	1		
	TPAMI10	3		
	TPAMI17	13		
	TPAMI23	7		
	Total	38		
Total	21 docs	197	2436	3301

Table 4.4: Dataset description: Distribution of pages and examples per document grouped by 5 folds.

	Positive Examples						
	Fold1	Fold2	Fold3	Fold4	Fold5	Avg Pos	Avg Neg
Abstract	4	4	4	4	5	4.2	656
Affiliation	4	5	5	4	4	4.4	655.8
Author	5	4	6	4	6	5	655.2
Biography	9	3	2	4	3	4.2	656
Caption	38	16	49	42	38	36.6	623.6
Figure	52	41	76	98	68	67	593.2
Formulae	50	118	61	62	36	65.4	594.8
Index Term	3	2	2	1	3	2.2	658
Reference	9	7	8	9	7	8	652.2
Table	9	10	15	6	6	9.2	651
Page Number	33	35	35	41	36	36	624.2
Paragraph	181	207	159	207	158	182.4	477.8
Running Head	45	37	41	42	38	40.6	619.6
Section Title	18	17	12	10	5	12.4	647.8
Subsection Title	11	9	1	3	1	5	655.2
Title	5	4	5	4	5	4.6	655.6
Total	*476*	*519*	*481*	*541*	*419*		

Table 4.5: Dataset description: concepts and distribution of examples

$$\mu AVG - recall \;=\; \frac{\sum_{i=1}^{L} TP(C_i)}{\sum_{i=1}^{L} (TP(C_i) + FN(C_i)} \tag{4.18}$$

where $TP(C_i)$ is the number of True Positive examples for a category C_i, $FP(C_i)$ is the number of False Positive examples for a category C_i and $FN(C_i)$ is the number of False Negative examples for a category C_i (see Table 4.6).

category C_i		Expert Judgment	
		YES	NO
Classifier Judgment	YES	$TP(C_i)$	$FP(C_i)$
	NO	$FN(C_i)$	$TN(C_i)$

Table 4.6: Contingency Table for C_i

Omission errors occur when a layout component is excluded from a category when it truly does belong to that category, while Commission errors occur when a layout component is included into a category when it does not belong to that category.

In table 4.7 the accuracy is reported. The results are obtained by averaging the accuracy over the 5 folds and then, by averaging the obtained values varying

CostRatio in the set of values $\{1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$. Results show that the accuracy of the decision taken strongly depends on the concept to be learned. For example Mr-SBC gives better results for the concepts "Running Head" and "Page Number" than "Figures" and "Paragraphs". This can be explained by the intrinsic simplicity of identifying a component rather than another in a document page taking into account its layout properties.

On the other hand, if we compare this results with the results obtained by the trivial classifier that returns the most probable class (i.e. False for each concept), we note that, in terms of accuracy, it is often better to assign the "undefined" class to each logical components. This is not true in the case of frequent concepts (e.g. "paragraph"), where the trivial classifier would return a lower accuracy (in the case of "paragraph" is 0.618250314), but when the concept is characterized by a very low number of examples, the situation is different and, in such case, it is important to assign a higher cost to omission errors rather than commission errors increasing *CostRatio*. This explains the choice of the interval $\{1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$ for the parameter *CostRatio*.

CostRatio	AVG	Standard Dev
Abstract	0.791	0.001
Affiliation	0.746	0.000
Author	0.734	0.000
Biography	0.737	0.000
Caption	0.742	0.007
Figure	0.721	0.002
Formulae	0.886	0.013
Index Term	0.778	0.000
Reference	0.788	0.004
Table	0.931	0.005
Page Number	0.976	0.002
Paragraph	0.729	0.006
Running Head	0.957	0.002
Section Title	0.729	0.000
Subsection Title	0.728	0.000
Title	0.751	0.000

Table 4.7: Average accuracy and Standard Deviation obtained varying *CostRatio* in the set of values $\{1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$

Different considerations can be drawn from Table 4.8 and Figures 4.20 and 4.21, where μAVG precision and recall are reported. We note that increasing the *CostRatio*, the precision decreases and the recall increases. This means that, as we expected, increasing *CostRatio*, we consider more significant omission errors rather than commission errors.

A different statistic in given in table 4.9 and figure 4.22 for omission errors and table 4.10 and figure 4.23 for commission errors. From such results we can draw two

CostRatio	micro Average Precision	Micro Average Recall
1	0.536802324	0.277008624
2	0.513988325	0.293561325
4	0.489784326	0.304663007
6	0.456410213	0.318248714
8	0.447274755	0.322534845
10	0.438241967	0.32457377
12	0.422210621	0.32523431
14	0.415346301	0.326047397
16	0.413492075	0.326898026
18	0.413492075	0.327688796
20	0.407589085	0.328159719

Table 4.8: Micro averaged precision and recall

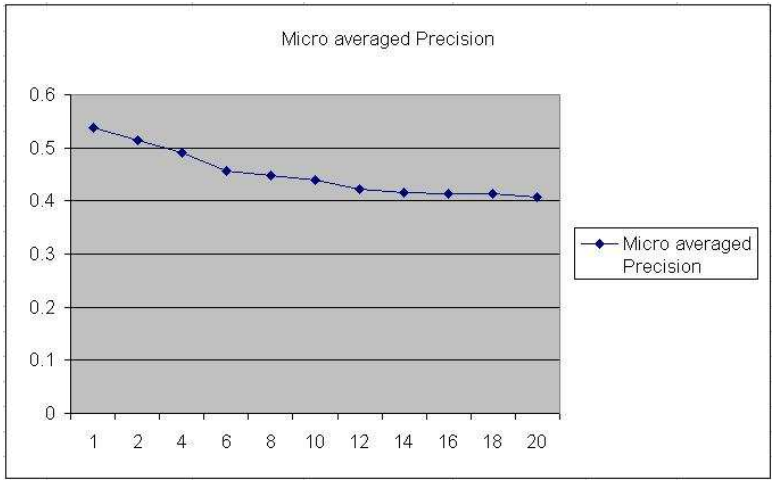

Figure 4.20: Micro averaged precision

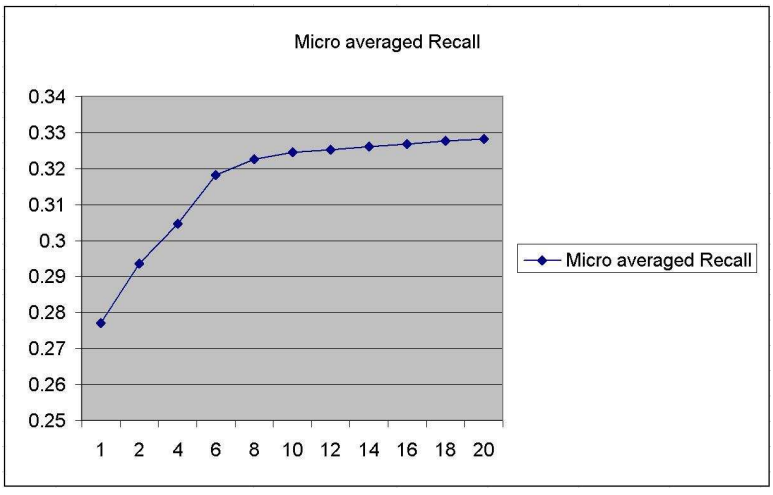

Figure 4.21: Micro averaged recall

main conclusions. First, if we compare omission errors with commission errors, we note that the percentage of commission errors is much lower than the percentage of omission errors. This means that the learned models are generally specific. Second, we note the different behavior of the system depending on the concept to be learned. For example, for some concepts (e.g. "Index term", "Page Number", "Title") we have relatively low omission and commission errors (e.g. for "Index Term", we have a 27.3% of Omission errors and 22.2% of commission errors). For other tasks, such as "Table", we have complete different results (from 89% to 80.0% of commission errors and from 0.5% to 0.6% of commission errors). This different behaviour can be explained by the inherent complexity of some learning tasks in relation to the used descriptors.

CostRatio	1	2	4	6	8	10	12	14	16	18	20
Abstract	0.810	0.810	0.810	0.810	0.810	0.810	0.810	0.810	0.810	0.810	0.810
Affiliation	0.773	0.773	0.773	0.773	0.773	0.773	0.773	0.773	0.773	0.773	0.773
Author	0.400	0.400	0.400	0.400	0.400	0.400	0.400	0.400	0.400	0.400	0.400
Biography	0.571	0.571	0.571	0.571	0.571	0.571	0.571	0.571	0.571	0.571	0.571
Caption	0.754	0.754	0.738	0.738	0.738	0.738	0.738	0.738	0.738	0.727	0.727
Figure	0.642	0.636	0.621	0.621	0.621	0.618	0.618	0.612	0.612	0.612	0.612
Formulae	0.807	0.709	0.670	0.606	0.584	0.572	0.566	0.566	0.563	0.563	0.563
Index Term	0.273	0.273	0.273	0.273	0.273	0.273	0.273	0.273	0.273	0.273	0.273
Reference	0.650	0.650	0.625	0.625	0.600	0.600	0.600	0.600	0.575	0.575	0.575
Table	0.891	0.891	0.891	0.826	0.826	0.826	0.826	0.804	0.804	0.804	0.804
Page Number	0.333	0.278	0.278	0.272	0.261	0.256	0.256	0.256	0.256	0.256	0.256
Paragraph	0.899	0.898	0.895	0.893	0.893	0.893	0.893	0.893	0.893	0.893	0.891
Running Head	0.601	0.596	0.591	0.557	0.557	0.552	0.552	0.552	0.552	0.552	0.552
Section Title	0.484	0.484	0.484	0.484	0.484	0.484	0.484	0.484	0.484	0.484	0.484
Subsection Title	0.720	0.720	0.720	0.720	0.720	0.720	0.720	0.720	0.720	0.720	0.720
Title	0.391	0.391	0.391	0.391	0.391	0.391	0.391	0.391	0.391	0.391	0.391

Table 4.9: AVG #Omission Errors/ AVG #Positive Examples

4.2.5 Conclusions

In this section we proposed a practical application of the multi-relational statistical classifier Mr-SBC to the Document Understanding problem. Document understanding is defined as the formal representation of the abstract relationships indicated by the two-dimensional arrangement of the symbols [Nag00].

We described the system WISDOM++ and its complex processing steps. In particular, we focused our attention on the description of the layout structure of a document page that is used as input to the Document Understanding step.

The application of Mr-SBC in a benchmark dataset of twenty-one multi-page

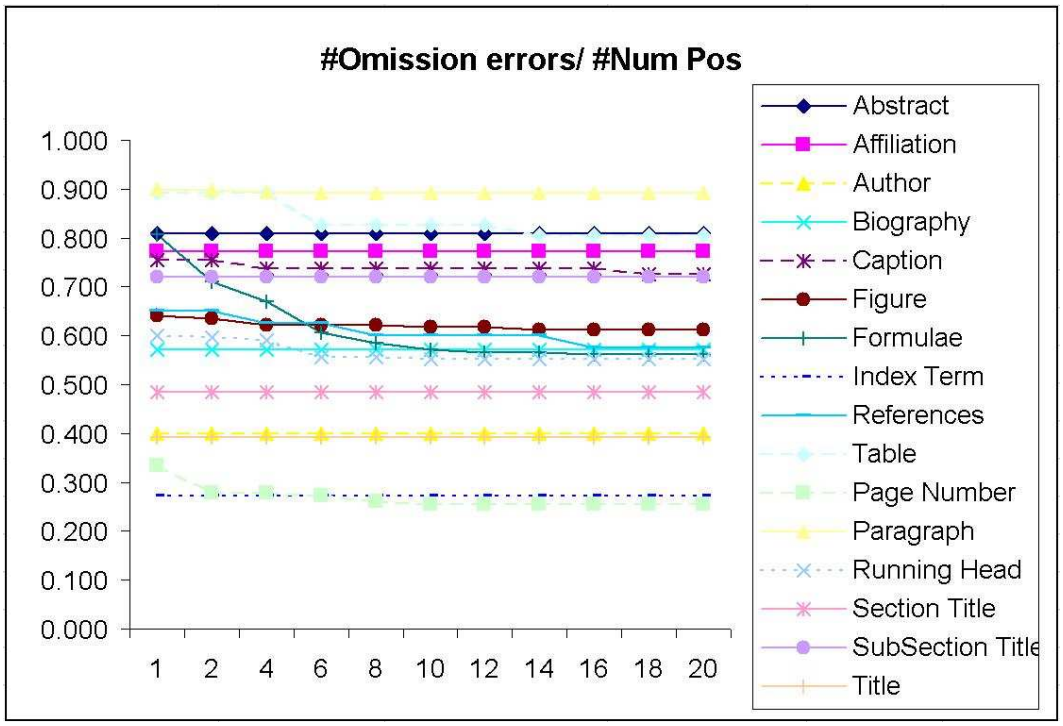

Figure 4.22: AVG #Omission Errors/ AVG #Positive Examples

CostRatio	1	2	4	6	8	10	12	14	16	18	20
Abstract	0.209	0.209	0.209	0.209	0.209	0.209	0.209	0.209	0.211	0.211	0.211
Affiliation	0.251	0.251	0.251	0.251	0.251	0.251	0.251	0.251	0.251	0.251	0.251
Author	0.265	0.265	0.265	0.265	0.265	0.265	0.265	0.265	0.265	0.265	0.265
Biography	0.259	0.259	0.259	0.259	0.259	0.259	0.259	0.259	0.259	0.259	0.259
Caption	0.221	0.221	0.228	0.232	0.232	0.233	0.233	0.233	0.233	0.242	0.242
Figure	0.229	0.230	0.232	0.233	0.233	0.233	0.233	0.239	0.239	0.239	0.239
Formulae	0.014	0.028	0.040	0.055	0.062	0.068	0.074	0.075	0.081	0.081	0.081
Index Term	0.222	0.222	0.222	0.222	0.222	0.222	0.222	0.222	0.222	0.222	0.222
Reference	0.199	0.199	0.201	0.201	0.206	0.206	0.206	0.206	0.210	0.210	0.210
Table	0.051	0.051	0.056	0.059	0.059	0.060	0.060	0.063	0.063	0.066	0.066
Page Number	0.004	0.005	0.007	0.009	0.011	0.012	0.012	0.013	0.013	0.013	0.013
Paragraph	0.019	0.020	0.022	0.028	0.031	0.032	0.039	0.039	0.039	0.039	0.045
Running Head	0.001	0.004	0.008	0.009	0.009	0.011	0.011	0.011	0.011	0.011	0.011
Section Title	0.266	0.266	0.266	0.266	0.266	0.266	0.266	0.266	0.266	0.266	0.266
Subsection Title	0.268	0.268	0.268	0.268	0.268	0.268	0.268	0.268	0.268	0.268	0.268
Title	0.247	0.247	0.247	0.247	0.247	0.247	0.247	0.247	0.247	0.247	0.247

Table 4.10: AVG #Commission Errors/ AVG #Negative Examples

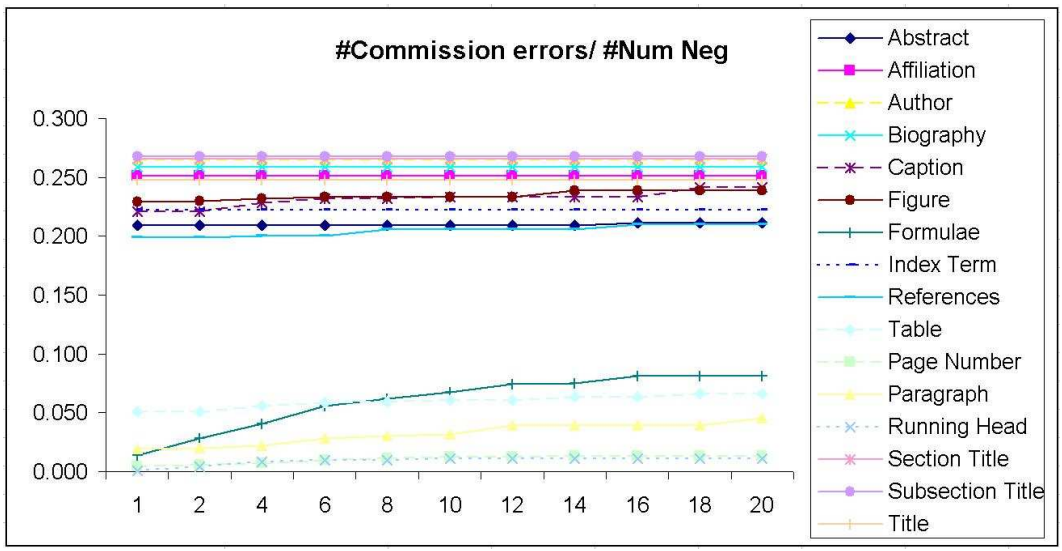

Figure 4.23: AVG #Commission Errors/ AVG #Negative Examples

articles published in IEEE Transactions on Pattern Analysis and Machine Intelligence in the January and February issues of 1996, showed that the percentage of commission errors is very low with respect to the percentage of omission errors. This means that the learned models are generally specific. The second conclusion concerns the different behavior of the system depending on the concept to be learned. The system shows relatively good performances on concepts such as "Index term", "Page Number" and "Title" and does not show good results for concepts such as "Table". This different behaviour can be explained by the inherent complexity of some learning tasks in relation to the used descriptors.

For future developments we intend to enrich the description of the layout structure by adding other information both on the physical description of the block (e.g. density of color pixels) and on the textual content of the block. Indeed, the idea of combining layout and textual information is not novel and in the work by Kovacevic and his colleagues [KDGM04] [KDGM02] this idea has been applied in HTML web-page classification.

We also intend to compare the performances of Mr-SBC with an ILP system. Indeed, early results showed that the ILP system ATRE [Mal03] has good performances if applied to the Document Understanding problem [VBM04]. Although the experimental results reported in [VBM04] has been obtained on the same dataset we used, only the first pages have been taken into account and, in addition, the most complex concept (i.e. "paragraph") has not been included in the set of concepts to be learned.

4.3 Conclusions

In this chapter we proposed the application of algorithms presented in Chapter 2 and Chapter 3 in the filed of Document Engineering. Results are reported and

conclusions are drawn.

Chapter 5

Classification in Multi-Relational Data Mining: other applications

In this chapter we show the application of the system Mr-SBC (described in section 3.3) and the associative classification system (described in section 3.4) in other application domains. In particular, we propose the application of Mr-SBC in predicting the class of biological structured data and the application of the associative classification system in predicting the class of geo-referenced census data.

5.1 Naive Bayes Structural Classification: Predicting the class of complex data

In this section we show the application of Mr-SBC to two well-known benchmark datasets, namely the *Mutagenesis* dataset and the *Biodegradability* dataset.

5.1.1 Experiments on Mutagenesis

These datasets, taken from the MLNET repository [1], concern the problem of identifying the mutagenic compounds [MBHMM89] and have been extensively used to test both inductive logic programming (ILP) systems and (multi-)relational mining systems. We considered, analogously to related experiments in the literature, the "regression friendly" dataset of 188 elements.

A recent study on this database [SKM99] recognizes five levels of background knowledge for mutagenesis which can provide richer descriptions of the examples. In this study we used only the first three levels of background knowledge in order to compare the performance of Mr-SBC with other methods for which experimental results are available in the literature. Table 5.1 shows the first three sets of background knowledge used in our experiments, where $BK_i \subseteq BK_{i+1}$ for $i = 0, \ldots, 2$.

[1] http://www.mlnet.org/cgi-bin/mlnetois.pl/?File=datasets.html&OrderBy=15+DESC

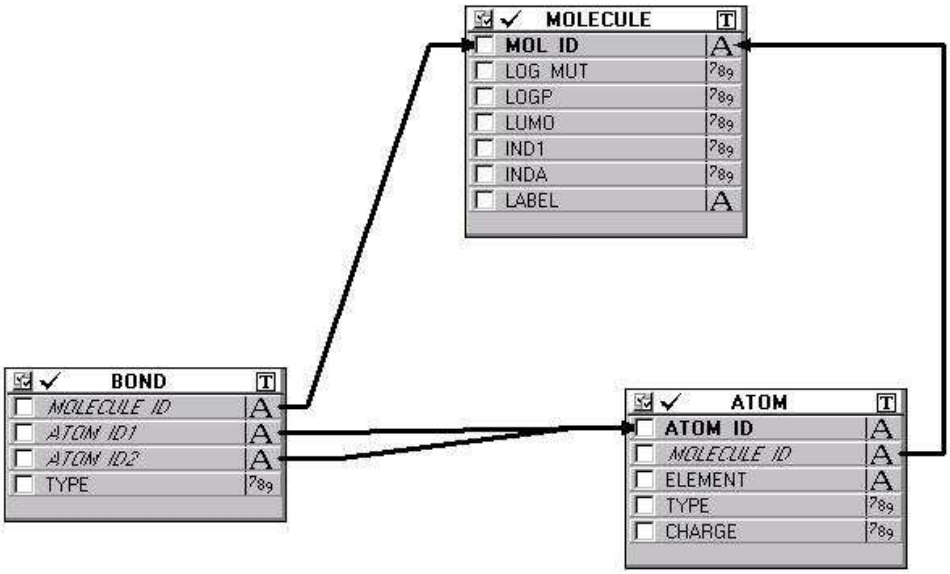

Figure 5.1: The Mutagenesis database schema

The greater the BK, the more complex the learning problem.

Background	Description
BK_0	Consists of those data obtained with the molecular modelling package QUANTA. For each compound it obtains the atoms, bonds, bond types, atom types, and partial charges on atoms.
BK_1	Consists of Definitions in B_0 plus indicators ind1, and inda in molecule table.
BK_2	Variables (attributes) logp, and lumo are added to definitions in BK1.

Table 5.1: Background knowledge for Mutagenesis dataset.

The dataset is analyzed by means of a 10-fold cross-validation, that is, the target table is first divided into ten blocks of near-equal size and distribution of class values, and then, for every block, a subset of tuples related to the tuples in the target table block are extracted. In this way, ten databases are created. Mr-SBC is trained on nine databases and tested on the hold-out database. Mr-SBC has been executed with the following parameters: MAX_LEN_PATH=4 and MAX_GAIN= 0.5. The schema of a single database is shown in figure 5.1.

Experimental results on predictive accuracy are reported in Table 5.2 for increasing complexity of the models. A comparison to other results reported in the literature is also made. Mr-SBC has the best performance (together with 1BC and 1BC2) for the most complex task (BK_2) with an accuracy of almost 90%, while it performance is comparable with other systems for the simplest task. Interestingly,

the predictive accuracy increases with the complexity of the background knowledge, which means that the variables added in BK_1 and BK_2 are meaningful and Mr-SBC takes advantages of that.

System	Accuracy(%)		
	BK_0	BK_1	BK_2
Progol_1	79	86	86
Progol_2	76	81	86
Foil	61	61	83
Tilde	75	79	85
MRDTL	67	**87**	88
1BC2	**82.4**	83	**89.9**
1BC	79.3	85.1	**89.9**
Mr-SBC	77.8	83.7	**89.9**

Table 5.2: Accuracy comparison on the set of 188 regression friendly elements of Mutagenesis. Results for Progol2, Foil, Tilde are taken from [Blo98]. Results for Progol_1 are taken from [SKM99]. The results for 1BC and 1BC2 are taken from [FL04]. Results for MRDTL are taken from [Lei02]. The values are the results of 10-fold cross-validation.

As regards execution time (see Table 5.3). The time required by Mr-SBC increases with the complexity of the background knowledge. Mr-SBC is generally considerably faster than competing systems, such as Progol, Foil and Tilde, that do not operate on data stored in a database. Moreover, except for the task BK_0, Mr-SBC performs better that MRDTL which works on a database. In general, the trade-off between accuracy and complexity is in favour of Mr-SBC.

The average number of extracted rules for each fold is quite high (55.9 for BK_0, 59.9 for BK_1, and 64.8 for BK_2). Some rules are either redundant or cover very few individuals. Therefore, some additional stopping criteria are required to avoid the generation of these rules and to reduce further the cost complexity of the algorithm.

System	Time(Secs)		
	BK_0	BK_1	BK_2
Progol_1	8695	4627	4974
Progol_2	117000	64000	42000
Foil	4950	9138	0.5
Tilde	41	170	142
MRDTL	0.85	170	142
1BC2	–	–	–
1BC	–	–	–
MR-SBC	36	42	48

Table 5.3: Time comparison of the set of 188 regression friendly elements of Mutagenesis. Results for Progol2, Foil, Tilde are taken from [Blo98]. Results for Progol_1 are taken from [SKM99]. Results for MRDTL are taken from [Lei02]. The results of MR-SBC are taken on a PIII WIN2k platform.

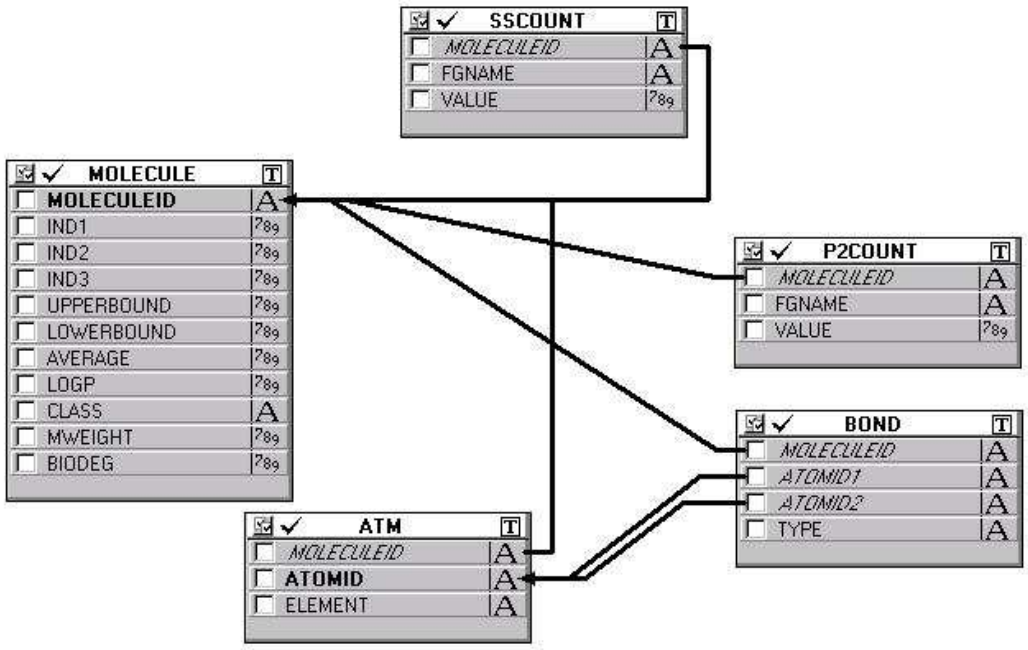

Figure 5.2: The Biodegradability database schema

5.1.2 Experiments on Biodegradability

The Biodegradability dataset has already been used in the literature for both regression and classification tasks [DBK+99]. It consists of 328 structural chemical molecules described in terms of atom and bond. The target variable for machine learning systems is the natural logarithm of the arithmetic mean of the low and high estimate of the HTL (Half-Life Time) for acqueous biodegradation in aerobic conditions, measured in hours. We use a discretized version in order to apply classification systems to the problem. As in [DBK+99], four classes have been defined: chemicals degrade fast, moderately, slowly or are resistant.

The dataset is analyzed by means of a 10-fold cross-validation. For each database Mr-SBC and Tilde are trained on nine databases and tested on the hold-out database. The database schema is shown in Figure 5.2. Mr-SBC has been executed with the following parameters: MAX_LEN_PATH=4 and MAX_GAIN= 0.5. Experimental results on predictive accuracy are reported in Table 5.4. They are in favour of Mr-SBC on the average of accuracy varying the fold.

5.1.3 Conclusions

In this section, the multi-relational data mining system Mr-SBC has been empirically evaluated on biological datasets. In particular, Mr-SBC has been tested on four benchmark tasks. Results on predictive accuracy are in favour of our system for the most complex tasks. Mr-SBC also proved to be efficient and this is mainly due to the tight integration with a relational DBMS.

Fold	*Mr-SBC*	*Tilde Pruned*
0	**0.90909**	0.69697
1	**0.87878**	0.81818
2	0.84848	**0.90909**
3	0.87878	0.87879
4	**0.78788**	0.69697
5	0.84848	**0.90909**
6	0.90625	0.90625
7	**0.87879**	0.81818
8	0.87500	**0.93750**
9	**0.93939**	0.72727
Average	0.87509	0.82983

Table 5.4: Accuracy comparison on the set of 328 chemical molecules of Biodegradability. Results for Mr-SBC and Tilde are reported.

As future work, we plan to extend the comparison of Mr-SBC to other multi-relational data mining systems on a larger set of benchmark datasets. Moreover, we intend to frame the proposed method in a transduction inference setting, where both labelled and unlabelled data are available for training. Although this setting is well-studied for Support Vector Machines [GAV98], it has been recently extended to other learning models [KK02] and, in particular, to the Bayesian Framework [GHO99].

5.2 Naive Bayes associative Classification: A Spatial Data Mining Application

In this section we evaluate the naive Bayesian associative Classification framework proposed and described in section 3.4.

In oder to evaluate the proposed method, the integration of multi-level spatial association rules discovery with naive Bayesian classification has been implemented. In particular, it has been realized in a spatial associative classification system based on a client-server model (see Figure 5.3).

Both the spatial association rule miner SPADA and the multi-relational naive Bayes classifier are on the server side, so that several data mining tasks can be run concurrently by multiple users. For each granularity level, extracted rules concur in building the spatial classification model by exploiting a multi-relational naive Bayesian classifier integrated with the SDB.

On the client side, the framework includes a Graphical User Interface (GUI), which provides users with facilities for controlling all parameters of the mining process.

SPADA, like many other association rule mining algorithms, cannot process numerical data properly, so it is necessary to perform a discretization of numerical features with a relatively large domain. For this purpose, the framework includes in

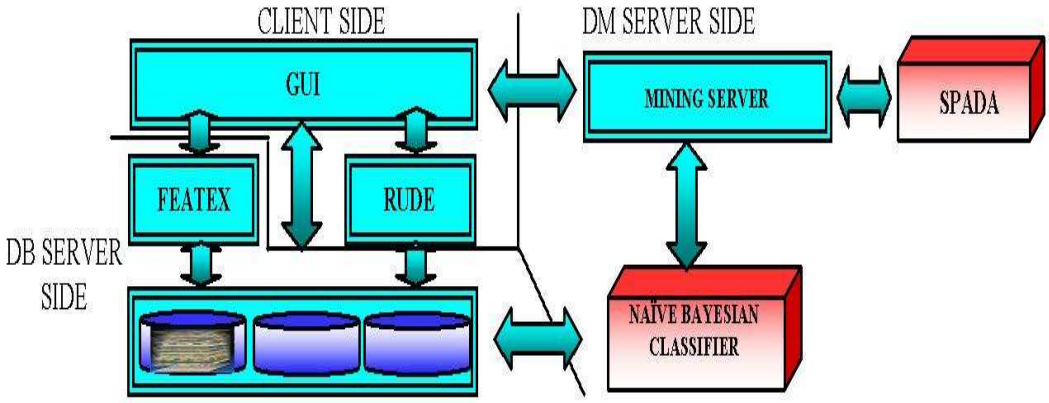

Figure 5.3: Spatial associative classification system

the client side the module RUDE (relative unsupervised discretization algorithm) which discretizes a numerical attribute of a relational database in the context defined by other attributes [LW00].

The SDB (Oracle Spatial) can run on a third computation unit. Many spatial features (relations and attributes) can be extracted from spatial objects stored in the SDB. Feature extraction requires complex data transformation processes to make spatial relations explicit and representable as ground Prolog atoms. Therefore, a middle layer module, named FEATEX (Feature Extractor), is required to make possible a loose coupling between SPADA and the SDB by generating features of spatial objects (points, lines, or regions). The module is implemented as an Oracle package of procedures and functions, each of which computes a different feature [ACL+03]. Transformed data are also stored in SDB tables.

5.2.1 The Application: Mining North West England Census Data

In this section we present a real-world application concerning the mining of both spatial association rules and classification models for geo-referenced census data interpretation. We consider both census and digital map data provided in the context of the European project SPIN! (Spatial Mining for Data of Public Interest) [May00]. They concern Greater Manchester, one of the five counties of North West England (NWE). Greater Manchester is divided into ten metropolitan districts, each of which is decomposed into censual sections or wards, for a total of two hundreds and fourteen wards. Spatial analysis is enabled by the availability of vectorized boundaries of the 1998 census wards as well as by other Ordnance Survey digital maps of NWE, where several interesting layers are found, namely road net, rail net, water net, urban area and green area (see Table 5.5).

Census data are available at ward level. They provide socio-economic statistics (e.g. mortality rate, that is, the percentage of deaths with respect to the number of inhabitants) as well as some measures describing the deprivation level. Indeed,

Layer name		Geometry	Objects
Road net	A-road	Line	3882
	B-road	Line	4368
	Motorway	Line	494
	Primary road	Line	3945
Rail net	Railway	Line	4231
Urban area	Large urban area	Line	384
	Small urban area	Line	2235
Green area	Wood	Line	859
	Park	Line	11
Water net	Water	Line	438
	River	Line	12103
	Canal	Line	968
Greater Manchester Ward	Ward	Region	214

Table 5.5: Geographic layers

the material deprivation of an area may be estimated according to information provided by Census combined into single index scores [AA00]. Over the years different indices have been developed for different applications: the Jarman Underprivileged Area Score was designed to measure the need for primary care, the indices developed by Townsend and Carstairs have been used in health-related analyses, while the Department of the Environment's Index (DoE) has been used in targeting urban regeneration funds. Thereby, we have considered the values of Jarman index, Townsend index, Carstairs index and DoE index. The higher the index value the more deprived a ward is. Both index values as well as mortality rate are all numeric and have been discretized by means of RUDE. More precisely, Jarman index, Townsend index, DoE index and Mortality rate have been automatically discretized in (*low, high*), while Carstairs index has been discretized in (*low, medium, high*).

For this application, we have considered Greater Manchester wards as reference (target) objects. In particular, three different experimental settings have been analysed by varying the target property among mortality rate, Jarman index and DoE index. We have chosen Jarman and DoE indices because they are defined on the basis of different social factors. For each setting, we have focused our attention on investigating dependencies between the target property and socio-economic factors represented in census data as well as geographical factors represented in linked topographic maps. These dependencies are detected in form of spatial association rules having only the target property in the head. Rules in this form may be employed for spatial subgroup mining, that is, discovery of interesting groups of spatial objects with respect to a certain property of interest [KM02] as well as for classification purpose.

For this analysis, we have formulated queries involving the FEATEX *relate* function to compute topological relationships between reference objects and task relevant objects. For instance, a relationship extracted by FEATEX is *crosses(ward_135, urbareaL_151)*, where *ward_#* denotes a specific Greater Manchester ward, while

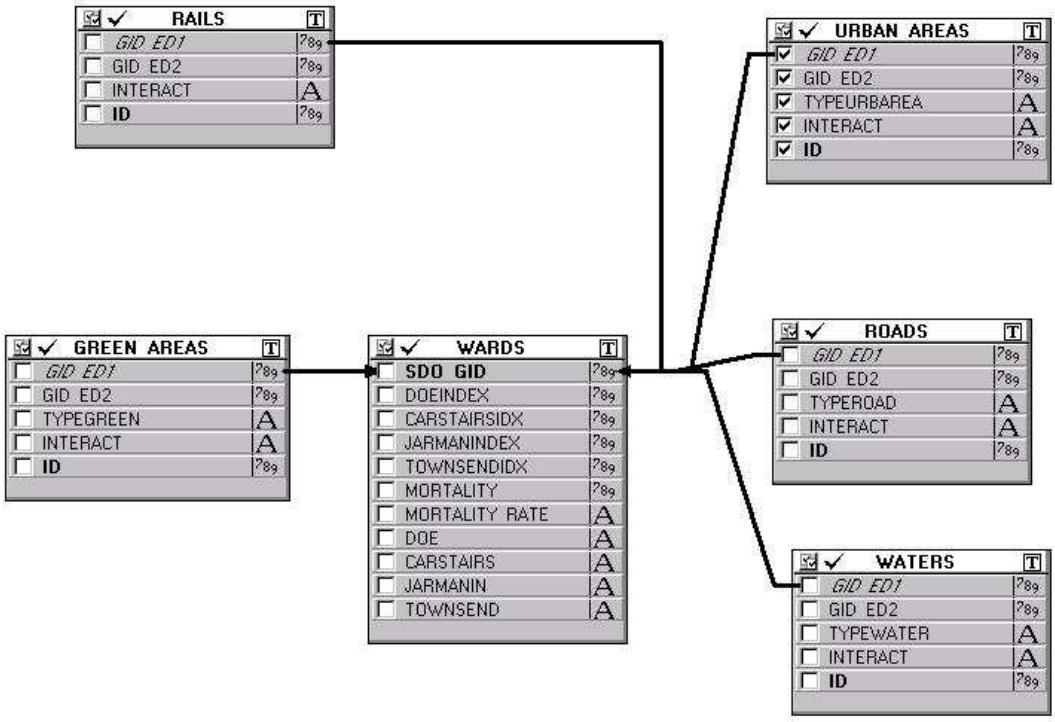

Figure 5.4: DB schema in North West England Census Data

urbanareaL# refers to a large urban area crossing the interested ward. The topological relationship *crosses* is computed according to the 9-intersection model [EF91]. The number of computed relationships is 784,107. Extracted relationships are automatically stored in the Database as relational tables. The database schema is reported in Figure 5.4

To support a spatial qualitative reasoning, a domain specific knowledge (BK) has been expressed in form of a set of rules. Some of these rules are:

crossed_by_urbanarea(X,Y) :- connects(X,Y), is_a(Y, urban_area). ...
crossed_by_urbanarea(X,Y) :- inside(X,Y), is_a(Y, urban_area).

Here the use of the predicate *is_a* hides the fact that a hierarchy has been defined for spatial objects which belong to the urban area layer. In detail, five different hierarchies have been defined to describe the following layers:

- road net
- rail net
- water net
- urban area
- green area

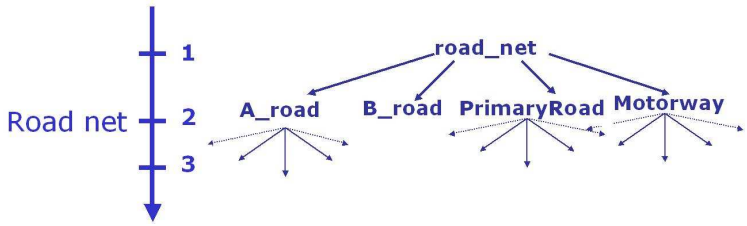

FIGURE 5.5: Road Network hierarchy

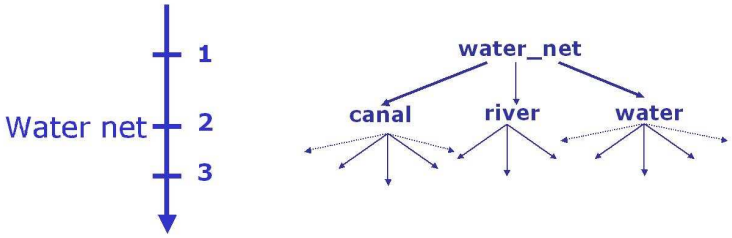

FIGURE 5.6: Water Network hierarchy

The hierarchies are shown in Figures 5.5, 5.6, 5.7, 5.8 and 5.9 , they have depth three and are straightforwardly mapped into three granularity levels. They are also part of the BK.

Finally, we have specified a language bias (LB) both to constrain the search space and to filter out uninteresting spatial association rules. In particular, we have ruled out all spatial relations (e.g. crosses, inside, and so on) directly extracted by FEATEX and asked for rules containing topological predicates defined by means of BK. Moreover, by combining the rule filters *head_constraint([mortality_rate(_),1,1)* and *rule_head_length(1,1)* we have asked for rules containing only *mortality rate* in the head. Similar considerations apply to the classification tasks concerning the Jarman and the DoE indices. In addition, we have specified the maximum number K of refinement steps (i.e. number of literals in the body of rules).

For each setting, a ten-fold cross validation has been performed and results are evaluated. For instance, by analyzing spatial association rules extracted with parameters $minsup = 0.1$, $minconf = 0.6$ we discover the following rule:

$$mortality\_rate(A, high) \leftarrow is\_a(A, ward), crossed\_by\_urbanarea(A, B),$$

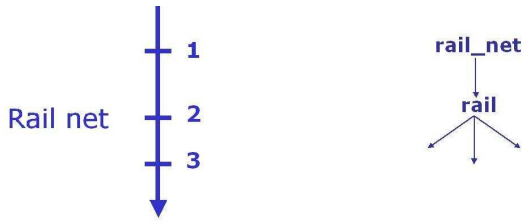

FIGURE 5.7: Rail Network hierarchy

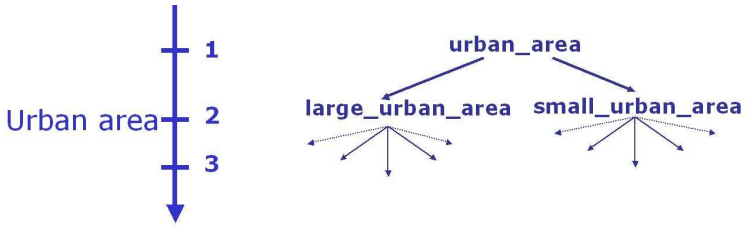

Figure 5.8: Urban Area hierarchy

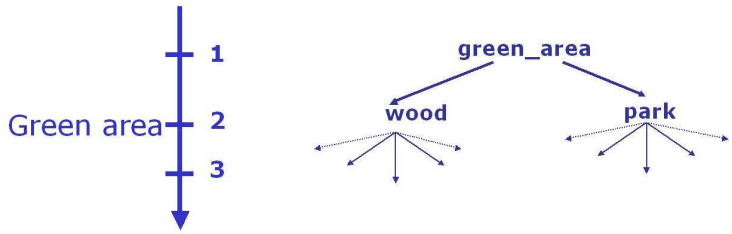

Figure 5.9: Green Area hierarchy

is_ a(B, urban_ area), townsendidx_ rate(A, high) (40.72%, 72.47%)

which states that a high mortality rate is observed in a ward A that includes an urban area B and has a high value of Townsend index. The support (40.72%) and the high confidence (72.47%) confirm a meaningful association between a geographical factor, such as living in deprived urban areas, and a social factor, such as the mortality rate. It is noteworthy that SPADA generates the following rule:

mortality_ rate(A, high) ← *is_ a(A,ward), crossed_ by_ urbanarea(A,B), is_ a(B, urban_ area)* (56.7%, 60.77%)

which has a greater support and a lower confidence. These two association rules show together an unexpected association between Townsend index and urban areas. Apparently, this means that this deprivation index is unsuitable for rural areas.

At a granularity level 2, SPADA specializes the task relevant object B by generating the following rule which preserves both support and confidence:

mortality_ rate(A, high) ← *is_ a(A, ward), crossed_ by_ urbanarea(A, B), is_ a(B, urban_ areaL), townsendidx_ rate(A,high)* (40.72%, 72.47%)

this rule clarifies that the urban area B is large.

The average predictive accuracy of mined multi-level spatial classification model is evaluated by varying *minsup*, *minconf* and K for each setting,. Results are reported in Tables 5.6, 5.7 and 5.8. In the first setting, results show that, predictive accuracy of the Bayesian classifier is slightly better than the accuracy (0.567) of the trivial classifier that returns the most probable class. We explain this result with the inherent complexity of the task. Different conclusions can be drawn from

both Jarman and DoE results, where the Bayesian classifiers significantly improve the trivial classifiers (acc. 0.542 and 0.625, respectively). Another consideration is that the average predictive accuracies of classification models discovered at higher granularity levels (i.e. level=2) are always better or equal to the corresponding accuracies at lowest levels. This means that the classification model takes advantage of the use of the hierarchies defined on spatial objects. Furthermore, results show that by decreasing the number of extracted rules (higher support and confidence) we have lower accuracy. This means that there are several rules that strongly influence classification results and often such rules are not characterized by high values of support and confidence. Finally, we observe that, generally, the higher the number of refinement steps, the better the model.

MORTALITY Avg. Accuracy		$K=4$	$K=5$	$K=6$	$K=7$
minsup=0.1 minconf=0.6	*Level=1*	0.5932	0.5915	0.5932	0.628
	Level=2	0.5932	0.596	0.5932	0.628
minsup=0.2 minconf=0.65	*Level=1*	0.5932	0.602	0.5932	0.623
	Level=2	0.5932	0.602	0.5932	0.623

TABLE 5.6: Mortality Rate average accuracy

JARMAN Avg. Accuracy		$K=4$	$K=5$	$K=6$	$K=7$
minsup=0.1 minconf=0.6	*Level=1*	0.8176	0.8176	0.8176	0.8176
	Level=2	0.8176	0.8176	0.8176	0.8176
minsup=0.2 minconf=0.8	*Level=1*	0.528	0.528	0.528	0.528
	Level=2	0.528	0.528	0.6272	0.6705

TABLE 5.7: Jarman average accuracy

DoE Avg. Accuracy		$K=4$	$K=5$	$K=6$	$K=7$
minsup=0.1, minconf=0.6	*Level=1*	0.912	0.912	0.912	0.912
	Level=2	0.912	0.912	0.912	0.912
minsup=0.2, minconf=0.8	*Level=1*	0.875	0.875	0.875	0.821
	Level=2	0.875	0.9028	0.883	0.874

TABLE 5.8: DoE average accuracy

5.2.2 Conclusions

In this section we empirically evaluated the Naive Bayesian associative classification framework. The application concerns the mining of both spatial association rules and classification models for geo-referenced census data. We considered both census and digital map data provided in the context of the European project SPIN! (Spatial Mining for Data of Public Interest) [May00]. They concern Greater Manchester, one of the five counties of North West England (NWE). Greater Manchester is divided into ten metropolitan districts, each of which is decomposed into censual sections or

wards, for a total of two hundreds and fourteen wards. Spatial analysis is enabled by the availability of vectorized boundaries of the 1998 census wards as well as by other Ordnance Survey digital maps of NWE, where several interesting layers are found, namely road net, rail net, water net, urban area and green area.

Experiments show that the use of different levels of granularity generally increases the accuracy of the mined classification model. As future work, we intend to frame the work within the context of hierarchical Bayesian classifiers, in order to exploit the multi-level nature of extracted association rules.

Chapter 6

Conclusions

In this final chapter, we summarize the contributions of this thesis and discuss a number of promising areas for future work.

6.1 Summary

In this thesis we face the problem of mining Naive Bayes statistical classifiers in presence of structured data taking into account different aspects related to both theoretical and applicative problems.

Among different types of structure in the data, we investigate two cases. The first case concerns the presence of a taxonomical relation on the categories of the units of analysis (categorization structure) and the second case concerns the presence of relationships between objects composing the units of analysis (unit structure). The former is investigated in the context of propositional learning, while to represent the unit structure, we resort to the multi-relational data mining setting.

In order to investigate the classification in presence of taxonomical relation on the categories of the units of analysis, we proposed a method, that has been implemented in the system WebClassIII, that is able to classify examples in a hierarchy of categories. The hierarchical arrangement is essential when the number of categories is quite high and the use of a non-hierarchical classifier (flat classifier) would lead to a fragmentation of the class, producing many classes with few instances per class. Furthermore, the hierarchical classification arranges examples hierarchically, thus supporting a thematic search by browsing topics of interests.

The advantage of this hierarchical view in the classification process is that the problem is partitioned into smaller subproblems, each of which can be effectively and efficiently managed. Another advantage is given by the observation that at different levels of the hierarchy the same example can be represented in a different way. In particular, it is possible to use different abstractions of the same object varying the level of the hierarchy (e.g. it is possible to emphasize some features rather than others at different levels of the hierarchy). WebClassIII includes a tree distance-based thresholding algorithm for the classification of examples in internal categories of the hierarchy. It can be applied to any classifier, such as naive Bayes,

that returns a degree of membership (e.g. probabilistic or distance based) of an example to a category. In our experiments, we applied our algorithm to naive Bayesian classifier and compared obtained results with a centroid-based classifier and with SVMs. We evaluated the performances of the system on three different text categorization datasets and we found that, although the flat SVM is the most accurate classifier, hierarchical naive Bayesian classifier takes great advantage of the use of the hierarchy and seem to be a valid alternative to SVM.

As regards the classification in presence of relationships between objects composing the units of analysis, we proposed two different solutions. In both cases we extend the naive Bayes classification to the multi-relational data mining setting. The first solution is based on the use of a set of first-order classification rules in the context of naive Bayesian classification and has been implemented in the system Mr-SBC. The second solution is inspired by recent studies on the usage of association rules for classification purposes (Associative Classification). In particular, we have presented a spatial associative classifier that combines spatial association rule discovery with naive Bayesian classification. Domain specific knowledge may be defined as a set of rules that makes possible the qualitative reasoning. In addition, it is possible to define hierarchies on the domain of units of observations. Objects are expressed by a collection of ground atoms and are exploited to mine classification models at different granularity levels.

Applications mainly concern the field of Document Engineering. Document Engineering is the computer science discipline that investigates systems for documents in any form and in all media. It is concerned with principles, tools and processes that improve our ability to create, manage, and maintain documents. As for the hierarchical classification we evaluated WebClassIII in a particular field of Document Engineering, namely in the context of text categorization. As for the multi-relational classification, we evaluated a opportunely modified version of Mr-SBC in the context of Document Understanding that is, the Document Engineering field that is concerned with semantic analysis of (paper) documents to extract human understandable information and codify it into machine-readable form. Results showed that, although the problem is intrinsically complex, the system presents good performances for most of concepts to be learned.

Other applications concern other domains where the concept of "structure" is particularly relevant, namely data mining from biological data and spatial data mining. In particular, in the last field, we evaluated the spatial associative classification framework. In this way we take into account two types of "structure" namely, the intrinsic relational structure of spatial data implicitly defined by the spatial location of objects with respect to others and the taxonomical nature in the domain of units of observations in spatial domain.

6.2 Future Work

We consider two possible directions for future work: methodological and applicative.

For the methodological direction, we plan to extend the approach implemented

in WebClassIII by considering the integration of both the multi-dimensional [TL02] and the hierarchical frameworks, in order to support WebClassIII users with OLAP-like roll-up, drill-down and pivoting operations in an information retrieval context. Furthermore, we intend to investigate the possibility of restructuring the original category hierarchy on the basis of some training examples [VG02] [SVAP04]. It can be realized by means of a greedy procedure that adds or removes categories until no further improvements can be made. Hierarchy restructuring can substantially improve the accuracy of the hierarchical approach.

Concerning the approach implemented in Mr-SBC, we plan to frame the proposed method in a transductive setting, where both labelled and unlabelled data are available for training [GAV98] [KK02] [GHO99]. Differently from an inductive approach where the learner tries to induce a model which has low error rate on the whole distribution of examples for the particular learning task, in the trasductive approach we do not care about the particular decision model, but rather that we classify a given set of examples (i.e. the test set) with as few errors as possible. The trusductive approach is particularly useful when there is a little training data, but a very large test set [Joa99b]. This is particularly relevant in the case of Document Engineering and, in particular, in both document understanding and text categorization where the user has often to manually label thousands of examples.

As regards the applicative direction, we intend to strengthen our results for the field of document image understanding by enriching the logical description of the page layout with information on both the color and the textual content. The information on the color would be useful in order to effectively reduce the presence and the effect of noise. Concerning the use of textual information, indeed this idea is not new and has already been proposed by Kovacevic et al. [KDGM02] for HTML web-page classification. The application to document images, however, presents some additional issues such as the identification of the correct reading order of the text [AMTW02].

6.3 Conclusion

This thesis has described our attempt to marring two fundamental concepts: on one side there is the concept of "structured data" and on the other side there is the concept of statistical approaches for learning classification models.

Our hope is twofold. First we hope that the proposed methods will be a valid alternative to existing methods in terms of the trade-off between accuracy and complexity and second that the proposed approaches will be useful to particular applicative research areas, especially in the field of Document Engineering.

Bibliography

[AA00] G. Andrienko and N. Andrienko. Exploration of heterogeneous spatial data using interactive geo-visualization tools: study of deprivation indices in north-west england. Technical report, North-West England Report, IST European project SPIN!(Spatial Mining for Data of Public Interest), 2000.

[AAK95] H. Almuallim, Y. Akiba, and S. Kaneda. On handling tree-structured attributes. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 12–20, 1995.

[ACL+03] Annalisa Appice, Michelangelo Ceci, Antonietta Lanza, Francesca A. Lisi, and Donato Malerba. Discovery of spatial association rules in georeferenced census data: A relational mining approach. *Intelligent Data Analysis*, 7(6), 2003.

[ACM03] A. Appice, M. Ceci, and D. Malerba. Mining model trees: A multi-relational approach. In T. Horvath and A. Yamamoto, editors, *Inductive Logic Programming, 13th International Conference, ILP 2003*, volume 2835 of *LNAI*, pages 4–21. Springer-Verlag, 2003.

[ACRF04] Annalisa Appice, Michelangelo Ceci, Simon Rawles, and Peter A. Flach. Redundant feature elimination for multi-class problems. In Russ Greiner and Dale Schuurmans, editors, *Proceedings of the 21st International Conference on Machine Learning (ICML'2004)*, pages 33–40, Banff, Alberta, Canada, July 2004. ACM.

[ADW94] Chidanand Apté, Fred Damerau, and Sholom M. Weiss. Automated learning of decision rules for text categorization. *Information Systems*, 12(3):233–251, 1994.

[AEM99] O. Altamura, F. Esposito, and D. Malerba. Wisdom++: An interactive and adaptive document analysis system. In *in Proceedings of the 5th ICDAR*, pages 366 – 369, 1999.

[AEM01] Oronzo Altamura, Floriana Esposito, and Donato Malerba. Transforming paper documents into XML format with WISDOM++. *IJDAR*, 4(1):2–17, 2001.

[AMTW02] M. Aiello, C. Monz, L. Todoran, and M. Worring. Document understanding for a broad class of documents. *International Journal on Document Analysis and Recognition-IJDAR*, 5(1):1–16, 2002.

[AS94] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[Bay63] Thomas Bayes. *An Essay Toward Solving a Problem in the Doctrine of Chances*, volume 53. 1763. Reprinted in *Facsimiles of two papers by Bayes*, Hafner Publishing Company, New York, 1963.

[BBD+02] H. Blockeel, M. Bruynooghe, S. Dzeroski, J. Ramon, and J. Struyf. Hierarchical multi-classification. In Sašo Džeroski, Luc De Raedt, and Stefan Wrobel, editors, *MRDM02*, pages 21–35. University of Alberta, Edmonton, Canada, July 2002.

[BCEM03] Margherita Berardi, Michelangelo Ceci, Floriana Esposito, and Donato Malerba. Learning logic programs for layout analysis correction. In *Proc. of the 20th International Conference on Machine Learning (ICML 2003)*, pages 27–34, Menlo Park, California, (USA), 2003. AAAI Press.

[BCM00] Paula Brito, Joaquim Costa, and Donato Malerba, editors. *ECML-2000 Workshop on Dealing with structured data in Machine Learning and Statistics*, 2000.

[BD98] H. Blockeel and L. De Raedt. Top-down induction of first order logical decision trees. *Artificial Intelligence*, 101(1-2):285–297, 1998.

[BD00] H.H. Bock and E. Diday, editors. *Analysis of Symbolic Data. Exploratory Methods for Extracting Statistical Information from Complex Data*, volume 15 of *Studies in Classification, Data Analysis, and Knowledge Organisation*. Springer-Verlag, Berlin, 2000.

[Ben00] P. Bennett. Assessing the calibration of naive bayes' posterior estimates cmu-cs-00-155. Technical report, Carnegie-Mellon University, School of Computer Science, 2000.

[Ber93] J.O. Berger. *Statistical Decision Theory and Bayesian analysis*. Springer-Verlag, 1993.

[BFOS84] L. Breiman, J. Friedman, R. Olshen, and J. Stone. *Classification and regression tree*. Wadsworth & Brooks, 1984.

[BG03a] E. Baralis and P. Garza. Majority classification by means of association rules. In *In Proceedings of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 2838, pages 35–46. Springer-Verlag, 2003.

[BG03b] Elena Baralis and Paolo Garza. Majority classification by means of association rules. In N. Lavrač, D. Gamberger, H. Blockeel, and L. Todorovski, editors, *Principles and Practice of Knowledge Discovery in Databases, 7th European Conference, PKDD 2003*, volume 2838 of *LNAI*, pages 35–46. Springer-Verlag, 2003.

[BL97a] Arvin L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1–2):245–271, 1997.

[BL97b] Avrim L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artif. Intell.*, 97(1-2):245–271, 1997.

[Blo98] H. Blockeel. *Top-down induction of first order logical decision trees*. PhD thesis, Department of Computer Science, Katholieke UniversReit, Leuven, Belgium, 1998.

[Bra00] Ben Bradshaw. Semantic based image retrieval: a probabilistic approach. In *MULTIMEDIA '00: Proceedings of the eighth ACM international conference on Multimedia*, pages 167–176. ACM Press, 2000.

[Bun90] Wray Lindsay Buntine. *A Theory of Learning Classification Rules*. PhD thesis, University of Technology, Sydney, 1990.

[CAM03] Michelangelo Ceci, Annalisa Appice, and Donato Malerba. Mr-sbc: a multi-relational naive bayes classifier. In N. Lavrač, D. Gamberger, H. Blockeel, and L. Todorovski, editors, *Principles and Practice of Knowledge Discovery in Databases, 7th European Conference, PKDD 2003*, volume 2838 of *LNAI*, pages 95–106. Springer-Verlag, 2003.

[CAM04] Michelangelo Ceci, Annalisa Appice, and Donato Malerba. Spatial associative classification at different levels of granularity: A probabilistic approach. In J.F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, editors, *Principles and Practice of Knowledge Discovery in Databases, 8th European Conference, PKDD 2004*, volume 3202 of *LNAI*, pages 99–111. Springer-Verlag, 2004.

[CDF+00] Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom Mitchell, Kamal Nigam, and Seán Slattery. Learning to construct knowledge bases from the world wide web. *Artif. Intell.*, 118(1-2):69–113, 2000.

[Ces90] B. Cestnik. Estimating probabilities: A crucial task in machine learning. In *Proc. of the 9th European Conf. on Artificial Intelligence, ECAI'90*, pages 147–149, 1990.

[CGT89] S. Ceri, G. Gottlob, and L. Tanca. What you always wanted to know about datalog (and never dared to ask). *IEEE Transactions on Knowledge and Data Engineering*, 1(1):146–166, 1989.

[CK01] Amanda Clare and Ross D. King. Knowledge discovery in multi-label phenotype data. In *Proceedings of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD 2001*, volume 2168, pages 42+. Springer-Verlag, 2001.

[CKB87] B. Cestnik, I. Kononenko, and I. Bratko. Assistant 86: A knowledge-elicitation tool for sophisticated users. In I. Bratko and N. Lavrač, editors, *Progress in Machine Learning – Proceedings of the Second European Working Session on Learning (EWSL87)*, page 3145, Wilmslow, UK, 1987. Sigma Press.

[CM93a] P. Clark and S. Matwin. Using qualitative models to guide induction learning. In *Int. Conf. Machine Learning (ICML'93)*, pages 49–56, Amherst, MA, 1993.

[CM93b] Peter Clark and Stan Matwin. Using qualitative models to guide inductive learning. In *International Conference on Machine Learning*, pages 49–56, 1993.

[CM03] Michelangelo Ceci and Donato Malerba. Hierarchical classification of HTML documents with WebClassII. In Fabrizio Sebastiani, editor, *Proceedings of ECIR-03, 25th European Conference on Information Retrieval*, pages 57–72, Pisa, IT, 2003. Springer Verlag.

[CMLE03] M. Ceci, D. Malerba, M. Lapi, and F. Esposito. Automated classification of web documents into a hierarchy of categories. In Ö.M.A. Klopotek, S.T. Wierzchon, and K. Trojanowski, editors, *Intelligent Information Processing and Web Mining*, Advances in Soft Computing, pages 59–68. Springer, 2003.

[CN89] Peter Clark and Tim Niblett. The cn2 induction algorithm. *Machine Learning*, 3:261–283, 1989.

[CS96] Peter Cheeseman and John Stutz. Bayesian classification (autoclass): theory and results. pages 153–180, 1996.

[CTYG00] Wesley T. Chuang, Asok Tiyyagura, Jihoon Yang, and Giovanni Giuffrida. A fast algorithm for hierarchical text classification. In *DaWaK 2000: Proceedings of the Second International Conference on Data Warehousing and Knowledge Discovery*, pages 409–418. Springer-Verlag, 2000.

[DBK+99] S. Dzeroski, H. Blockeel, S. Kramer, B. Kompare, B. Pfahringer, and W. Van Laer. Experiments in predicting biodegradability. In S. Dzeroski and P. Flach, editors, *Proceedings of the Ninth International Workshop on Inductive Logic Programming LNAI*, pages 80–91. Springer, 1999.

[DC00] Susan Dumais and Hao Chen. Hierarchical classification of web content. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256–263. ACM Press, 2000.

[DH73] R. Duda and P.E. Hart. *Pattern classification and scene analysis*. John Wiley & Sons, New York, 1973.

[DKS95] James Dougherty, Ron Kohavi, and Mehran Sahami. Supervised and unsupervised discretization of continuous features. In *International Conference on Machine Learning*, pages 194–202, 1995.

[DL01] S. Džeroski and N. Lavrač. *Relational Data Mining*. Springer-Verlag, 2001.

[DMSK00] S. D'Alessio, K. Murray, R. Schiaffino, and A. Kershenbau. The effect of using hierarchical classifiers in text categorization. In *Proc. of the 6th Int. Conf. on "Recherche d'Information Assistée par Ordinateur" (RIAO)*, pages 302–313, 2000.

[DP96] Pedro Domingos and Michael J. Pazzani. Beyond independence: Conditions for the optimality of the simple bayesian classifier. In *International Conference on Machine Learning*, pages 105–112, 1996.

[DP97] P. Domingos and M. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

[DPHS98] S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *In Proceedings of ACM-CIKM98*, pages 148–155, 1998.

[DR97] Luc Dehaspe and Luc De Raedt. Mining association rules in multiple relations. In *ILP '97: Proceedings of the 7th International Workshop on Inductive Logic Programming*, pages 125–132. Springer-Verlag, 1997.

[DR98] L. De Raedt. Attribute-value learning versus inductive logic programming: the missing links. In *Proceedings of the 8 th International Conference on Inductive Logic Programming*, pages 128–137. Springer-Verlag, 1998.

[DS03] Franca Debole and Fabrizio Sebastiani. Supervised term weighting for automated text categorization. In *Proceedings of SAC-03, 18th ACM Symposium on Applied Computing*, pages 784–788, Melbourne, US, 2003. ACM Press, New York, US.

[DZWL99a] Guozhu Dong, Xiuzhen Zhang, Limsoon Wong, and Jinyan Li. Caep: Classification by aggregating emerging patterns. In *In Proceedings of the Second International Conference on Discovery Science*, volume 1721, pages 30–42. Springer-Verlag, 1999.

[DZWL99b] Guozhu Dong, Xiuzhen Zhang, Limsoon Wong, and Jinyan Li. CAEP: Classification by aggregating emerging patterns. In *Discovery Science*, pages 30–42, 1999.

[EF91] M.J. Egenhofer and R. Franzosa. Point-set topological spatial relations. *International Journal of Geographical Information Systems*, 5(4):61–174, 1991.

[EKJ97] M. Ester, H.P. Kriegel, and J.Sander. Spatial data mining: A database approach. In *Proceedings International Symposium on Large Databases*, pages 47–66, 1997.

[ELM03] S. Eyheramendy, D. Lewis, and D. Madigan. the naive bayes model for text categorization. 2003.

[EMS+90] F. Esposito, D. Malerba, G. Semeraro, E. Annese, and G. Scafuro. An experimental page layout recognition system for office document automatic classitication: An integrated approach for inductive generalization. In *In Proc. of the 10th Int. Conf on Pattern Recognition*, pages 557–562, 1990.

[EMTB00] F. Esposito, D. Malerba, V. Tamma, and H.H. BOCK. *Classical resemblance measures*, volume 15 of *Studies in Classification, Data Analysis, and Knowledge Organization*, chapter Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, pages 139–152. Springer-Verlag, 2000.

[FF03] J. Furnkranz and P.A. Flach. An analysis of rule evaluation metrics. In *Proc. 20th International Conference on Machine Learning (ICML'03)*, pages 202–209. AAAI Press, January 2003.

[FGG97] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Mach. Learn.*, 29(2-3):131–163, 1997.

[FGKP99a] Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In Morgan Kaufman, editor, *In Proceedings of the 6 th International Joint Conference on Artificial Intelligence*, pages 1300–1309, 1999.

[FGKP99b] Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1300–1309. Morgan Kaufmann Publishers Inc., 1999.

[FI94] U.M. Fayyad and K.B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *In Proc. Of the 13th International Joint Conference on Artificial Intelligence*, pages 1022–1027, 1994.

[FL00] P.A. Flach and N. Lachiche. Confirmation-guided discovery of first-order rules with tertius. *Machine Learning*, 42(1/2):61–95, 2000.

[FL04] P.A. Flach and N. Lachiche. Naive bayesian classification of structured data. *Machine Learning*, 57(3):233–269, 2004.

[FPSM92] W. J. Frawley, G. Piatetsky-Shapiro, and C. J. Matheus. Knowledge discovery in databases: An overview. *Ai Magazine*, 13:57–70, 1992.

[FPSS96] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery: an overview. *Advances in knowledge discovery and data mining*, pages 1–34, 1996.

[FSN99] Xien Fan, Fang Sheng, and Peter A. Ng. Docpros: A knowledge-based personal document management system. In *DEXA Workshop on Document Analysis & Understanding for Document Databases*, pages 527–531, 1999.

[FWI+98] Eibe Frank, Yong Wang, Stuart Inglis, Geoffrey Holmes, and Ian H. Witten. Using model trees for classification. *Machine Learning*, 32(1):63–76, 1998.

[GAV98] Alexander Gammerman, Katy S. Azoury, and Vladimir Vapnik. Learning by transduction. In *UAI 1998*, pages 148–155, 1998.

[Get01a] L. Getoor. *Learning Statistical Models from Relational Data.* PhD thesis, Stanford University, 2001.

[Get01b] L. Getoor. Multi-relational data mining using probabilistic relational models: research summary. In A. J. Knobbe and D. M. G. van der Wallen, editors, *In Proceedings of the First Workshop in Multi-relational Data Mining*, pages 1300–1309, 2001.

[GHO99] T. Graepel, R. Herbrich, and K. Obermayer. Bayesian Transduction. In *Advances in Neural Information System Processing*, 1999. accepted for publication.

[Got80] Hans-Werner Gottinger. *Elements of Statistical Analysis.* Walter De Gruyter Inc., hardcover edition, 1980.

[GSS00] Luigi Galavotti, Fabrizio Sebastiani, and Maria Simi. Experiments on the use of feature selection and negative evidence in automated text categorization. In *ECDL '00: Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries*, pages 59–68. Springer-Verlag, 2000.

[HAP89] Robert C. Holte, Liane Acker, and Bruce W. Porter. Concept learning and the problem of small disjuncts. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 813–818, 1989.

[HB02] Richard J. Hathaway and James C. Bezdek. Clustering incomplete relational data using the non-euclidean relational fuzzy c-means algorithm. *Pattern Recogn. Lett.*, 23(1-3):151–160, 2002.

[HCC92] Jiawei Han, Yandong Cai, and Nick Cercone. Knowledge discovery in databases: An attribute-oriented approach. In Li-Yan Yuan, editor, *Proceedings of the 18th International Conference on Very Large Databases*, pages 547–559, San Francisco, U.S.A., 1992. Morgan Kaufmann Publishers.

[Hel87] N. Helft. Inductive generalization: a logical framework. In I. Bratko and N. Lavrac, editors, *Progress in Machine Learning,*, pages 149–157. Sigma Press, 1987.

[HK00] Eui-Hong Han and George Karypis. Centroid-based document classification: Analysis and experimental results. In *PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 424–431. Springer-Verlag, 2000.

[HL00] D.W. Hosmer and S. Lemeshow. *Pattern classification and scene analysis, 2nd edition.* John Wiley & Sons, New York, 2000.

[Hol93] R.C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, (11):63–90, 1993.

[HT98] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In *NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10*, pages 507–513. MIT Press, 1998.

[HTF01] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2001.

[Jef61] Harold Jeffreys. *Theory of Probability*. Oxford University Press, London, third edition edition, 1961.

[JKP94] G. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 121–129, New Brunswick, 1994. Morgan Kaufmann.

[JL95] George H. John and Pat Langley. Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, 1995.

[Joa] T. Joachims. Svmlight, an implementation of support vector machines (svms) in c.

[Joa97] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, pages 143–151. Morgan Kaufmann Publishers Inc., 1997.

[Joa98] Thorsten Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *ECML '98: Proceedings of the 10th European Conference on Machine Learning*, pages 137–142. Springer-Verlag, 1998.

[Joa99a] Thorsten Joachims. *Making large-scale support vector machine learning practical*, pages 169–184. MIT Press, 1999.

[Joa99b] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pages 200–209. Morgan Kaufmann Publishers Inc., 1999.

[Kar95] A. Karalic. *First Order regression*. PhD thesis, Faculty of Electrical Engineering and Computer Science, Ljubljana, Slovenia, 1995.

[KB97] A. Karalic and I. Bratko. First order regression. *Machine Learning*, 26:147–176, 1997.

[KBSV99] J. Knobbe, H. Blockeel, A. Siebes, and D. M. G. Van der Wallen. Multi-relational data mining. In *Proceedings of the Benelearn '99*, 1999.

[KD00] K. Kersting and L. De Raedt. Bayesian logic programs. In J. Cussens and A. Frisch, editors, *Work-in-Progress Reports of the Tenth International Conference on Inductive Logic Programming (ILP -2000)*, 2000. `http://SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-35/`.

[KDGM02] Milos Kovacevic, Michelangelo Diligenti, Marco Gori, and Veljko Milutinovic. Recognition of common areas in a web page using visual information: a possible application in a page classification. In *ICDM '02: Proceedings of*

the 2002 IEEE International Conference on Data Mining (ICDM'02), page 250. IEEE Computer Society, 2002.

[KDGM04] Milos Kovacevic, Michelangelo Diligenti, Marco Gori, and Veljko Milutinovic. Visual adjacency multigraphs - a novel approach for a web page classification. In Marco Gori, Michelangelo Ceci, and Mirko Nanni, editors, *Proceedings of the ECML/PKDD'04 Workshop on Statistical Approaches for Web Mining*, pages 38–49, 2004.

[KDK00] S. Klink, A. Dengel, and T. Kieninger. Document structure analysis based on layout and textual features. In *In Proc. of Fourth IAPR International Workshop on Document Analysis Systems, DAS2000*, pages 99–111, 2000.

[KH95] Krzysztof Koperski and Jiawei Han. Discovery of spatial association rules in geographic information databases. In *SSD '95: Proceedings of the 4th International Symposium on Advances in Spatial Databases*, pages 47–66. Springer-Verlag, 1995.

[KHS01] J. Knobbe, M. Haas, and A. Siebes. Propositionalisation and aggregates. In L. De Raedt and A. Siebes, editors, *European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD 2001*, volume 2168 of *LNAI*, pages 277–288. Springer-Verlag, 2001.

[KK02] M. Kukar and I. Kononenko. Reliable classifications with machine learning. In *Proceedings of 13 European Conference on Machine Learning, ECML 2002*, 2002.

[KLF01] S. Kramer, N. Lavrač, and P. Flach. *Relational Data Mining*, chapter Propositionalization Approaches to Relational Data Mining, pages 262–291. LNAI. Springer-Verlag, Berlin Heidelberg Germany, 2001.

[KM02] W. Klosgen and M. May. Spatial subgroup mining integrated in an object-relationalspatial database. In T. Elomaa, H. Mannila, and H. Toivonen, editors, *European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD 2002*, volume 2431 of *LNAI*, pages 275–286. Springer-Verlag, 2002.

[Koh96] Ron Kohavi. Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 202–207, 1996.

[Kon90] Igor Kononenko. Comparison of inductive and naive bayesian learning approaches to automatic knowledge acquisition. In B. Wielinga, J. Boose, B. Gaines, G. Schreiber, and M. van Someren, editors, *Current trends in knowledge acquisition*. IOS Press., 1990.

[Kon91] Igor Kononenko. Semi-naive bayesian classifier. In *Proceedings of the European working session on learning on Machine learning*, pages 206–219. Springer-Verlag New York, Inc., 1991.

[Kon00] I. Kononenko. Inductive and bayesian learning in medical diagnosis. *Applied Artificial Intelligence*, 7:317337, 2000.

[Kop99a] K. Koperski. *Progressive Refinement Approach to Spatial Data Mining*. PhD thesis, Computing Science, Simon Fraser University, British Columbia, Canada, 1999.

[Kop99b] K. Koperski. *Progressive Refinement Approach to Spatial Data Mining*. PhD thesis, Computing Science, Simon Fraser University, British Columbia, Canada, 1999.

[Kra96] S. Kramer. Structural regression trees. In *Proceedings of the National Conference on Artificial Intelligence*, 1996.

[Kra99] S. Kramer. *Relational Learning vs. Propositionalization: Investigations in Inductive Logic Programming and Propositional Machine Learning*. PhD thesis, Vienna University of Technology, Vienna, Austria, 1999.

[KRYL02] S.B. Kim, H.C. Rim, D. Yook, and H. Lim. Effective methods for improving naive bayes text classifier. In *In 7th International Conference on Artificial Intelligence*, volume 2417 of *LNAI*, pages 95–106. 2002.

[KRZ+03] M. Krogel, S. Rawles, F. Zelezny, P.A. Flach, N. Lavrač, and S. Wrobel. Comparative evaluation of approaches to propositionalization. In *In Proc. of the 13th International Conference on Inductive Logic Programming*, pages 197–214. Springer-Verlag, 2003.

[KS97] Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 170–178. Morgan Kaufmann Publishers Inc., 1997.

[KW01a] S. Kramer and G. Widmer. *Relational Data Mining*, chapter Inducing Classification and Regression Trees in First Order Logic, pages 140–156. LNAI. Springer-Verlag, Berlin Heidelberg Germany, 2001.

[KW01b] M. Krogel and S. Wrobel. Transformation-based learning using multirelational aggregation. In Céline Rouveirol and Michèle Sebag, editors, *Proceedings of the 11 th International Conference on Inductive Logic Programming LNAI*, volume 2157. Springer-Verlag, 2001.

[Lan93] Pat Langley. Induction of recursive bayesian classifiers. In *Proceedings of the European Conference on Machine Learning*, pages 153–164. Springer-Verlag, 1993.

[Lei02] H. A. Leiva. Mrdtl: A multi-relational decision tree learning algorithm. Master's thesis, University of Iowa, USA, 2002.

[Ler00] I.C. Lerman. Comparing taxonomic data. In *Proceeding of ECML-2000 Workshop Dealing with Structured Data in Machine Learning and Statistics*, pages 18–29, 2000.

[Lew98] David D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *ECML '98: Proceedings of the 10th European Conference on Machine Learning*, pages 4–15. Springer-Verlag, 1998.

[LF03a] N. Lachiche and P. Flach. 1bc2: a true first-order bayesian classifier. In Claude Sammut and Stan Matwin, editors, *Proceedings of the Thirteenth International Workshop on Inductive Logic Programming (ILP'02) LNAI*, volume 2583, pages 133–148. Springer-Verlag, Sydney, Australia, 2003.

[LF03b] N. Lachiche and P.A. Flach. Improving accuracy and cost of two-class and multi-class probabilistic classifiers using roc curves. In *Proc. 20th International Conference on Machine Learning (ICML'03)*, pages 416–423. AAAI Press, January 2003.

[LHM98] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining KDD'98*, pages 80–86, New York, 1998.

[LIT92] Pat Langley, Wayne Iba, and Kevin Thompson. An analysis of bayesian classifiers. In *National Conference on Artificial Intelligence*, pages 223–228, 1992.

[LM02] F. Lisi and D. Malerba. Efficient discovery of multiple-level patterns. In *Atti del Decimo Convegno Nazionale su Sistemi Evoluti per Basi di Dati SEBD'02*, pages 237–250, 2002.

[LM04] Francesca A. Lisi and Donato Malerba. Inducing multi-level association rules from multiple relations. *Mach. Learn.*, 55(2):175–210, 2004.

[LS94] Pat Langley and Stephanie Sage. Induction of selective bayesian classifiers. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 399–406. Morgan Kaufmann, 1994.

[LT04] Verayuth Lertnattee and Thanaruk Theeramunkong. Effect of term distributions on centroid-based text categorization. *Inf. Sci. Inf. Comput. Sci.*, 158(1):89–115, 2004.

[LW00] M.C. Ludl and G. Widmer. Relative unsupervised discretization for association rule mining. In *European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD 2000*, volume 1910 of *LNCS*, pages 148–158. Springer-Verlag, 2000.

[LYRL04] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361–397, 2004.

[MAC03] D. Malerba, A. Appice, and M. Ceci. *Database Support for Data Mining Applications*, chapter A Data Mining Query Language for Knowledge Discovery in a Geographical Information System, pages 95–116. LNCS 2682. Springer-Verlag, Berlin Heidelberg Germany, 2003.

[Mal03] D. Malerba. Learning recursive theories in the normal ilp setting. *Fundamenta Informaticae*, 57(1):39–77, 2003.

[May00] M. May. Spatial knowledge discovery: The spin! system. In K. Fullerton, editor, *Proceedings of the EC-GIS Workshop*, 2000.

[MBHMM89] S. H. Muggleton, M. Bain, J. Hayes-Michie, and D. Michie. An experimental comparison of human and machine learning formalisms. In *In Proc. Sixth International Workshop on Machine Learning*, pages 113–118, San Mateo, CA, 1989. Morgan Kaufmann.

[MCB03] D. Malerba, M. Ceci, and M. Berardi. Xml and knowledge technologies for semantic-based indexing of paper documents. In V. Marik, W. Retschintzegger, and O. Stepankova, editors, *Database and Expert Systems Applications, (DEXA 2003)*, volume 2736 of *LNCS*, pages 256–265. Springer-Verlag, 2003.

[MCLA04] Donato Malerba, Michelangelo Ceci, Michele Lapi, and Giulio Altini. A new thresholding algorithm for hierarchical text classification. In Marco Gori, Michelangelo Ceci, and Mirko Nanni, editors, *Proceedings of the ECML/PKDD'04 Workshop on Statistical Approaches for Web Mining*, pages 62–74, 2004.

[MEA+03] Donato Malerba, Floriana Esposito, Oronzo Altamura, Michelangelo Ceci, and Margherita Berardi. Correcting the document layout: A machine learning approach. In *ICDAR*, page 97, 2003.

[MEC02] Donato Malerba, Floriana Esposito, and Michelangelo Ceci. Mining html pages to support document sharing in a cooperative system. In *EDBT '02: Proceedings of the Worshops XMLDM, MDDE, and YRWS on XML-Based Data Management and Multimedia Engineering-Revised Papers*, pages 420–434. Springer-Verlag, 2002.

[MECA04] Donato Malerba, Floriana Esposito, Michelangelo Ceci, and Annalisa Appice. Top-down induction of model trees with regression and splitting nodes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):612–625, 2004.

[MEL+03] D. Malerba, F. Esposito, A. Lanza, F. A. Lisi, and A. Appice. Empowering a gis with inductive learning capabilities: The case of ingens. *Journal of Computers, Environment and Urban Systems, Elsevier Science*, 27:265–281, 2003.

[MG99] Dunja Mladenić and Marko Grobelnik. Feature selection for unbalanced class distribution and naive bayes. In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pages 258–267. Morgan Kaufmann Publishers Inc., 1999.

[Mil90] G.A. Miller. Five papers on wordnet. *International Journal of Lexicology*, 3(4), 1990.

[Mit97] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[Mit98] T. Mitchell. Conditions for the equivalence of hierarchical and flat bayesian classifiers. Technical report, Carnegie-Mellon University, Center for Automated Learning and Discovery, 1998.

[Mla98a] Dunja Mladenić. Feature subset selection in text-learning. In *ECML '98: Proceedings of the 10th European Conference on Machine Learning*, pages 95–100. Springer-Verlag, 1998.

[Mla98b] Dunja Mladenić. *Machine learning on non-homogeneus, distribuited text data*. PhD thesis, University of Ljubjana, Ljubjana, Slovenia, 1998.

[MLAS02] D. Malerba, F.A. Lisi, A. Appice, and F. Sblendorio. Mining spatial association rules in census data: A relational approach. In P. Brito and D. Malerba, editors, *Notes of the ECML/PKDD 2002 Workshop on Mining Official Data*, pages 80–93, 2002.

[MN98] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *In AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[MP91] P. Murphy and M. Pazzani. D2-of-3: Constructive induction of m-of-n concepts for discriminators in decision trees. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 183–187, San Mateo, CA, 1991. Morgan Kaufmann Publishers Inc.

[MRMN98] Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 359–367. Morgan Kaufmann Publishers Inc., 1998.

[MSTC94] Donald Michie, D. J. Spiegelhalter, C. C. Taylor, and John Campbell. *Machine learning, neural and statistical classification.* Ellis Horwood, 1994.

[MT97] Heikki Mannila and Hannu Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Min. Knowl. Discov.*, 1(3):241–258, 1997.

[Nag00] George Nagy. Twenty years of document image analysis in pami. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):38–62, 2000.

[NGL97] Hwee Tou Ng, Wei Boon Goh, and Kok Leong Low. Feature selection, perception learning, and a usability case study for text categorization. *SIGIR Forum*, 31(SI):67–73, 1997.

[NH97] Liem Ngo and Peter Haddawy. Answering queries from context-sensitive probabilistic knowledge bases. *Theoretical Computer Science*, 171(1–2):147–177, 1997.

[NJFH03] Jennifer Neville, David Jensen, Lisa Friedland, and Michael Hay. Learning relational probability trees. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 625–630. ACM Press, 2003.

[NJG03] Jennifer Neville, David Jensen, and Brian Gallagher. Simple estimators for relational bayesian classifiers. In *Third IEEE International Conference on Data Mining*, page 609, Melbourne, Florida, 2003. IEEE Computer Society.

[NKK+88] George Nagy, Junichi Kanai, Mukkai Krishnamoorthy, Mathews Thomas, and Mahesh Viswanathan. Two complementary techniques for digitized document analysis. In *DOCPROCS '88: Proceedings of the ACM conference on Document processing systems*, pages 169–176. ACM Press, 1988.

[Paz96] M.J. Pazzani. Searching for dependencies in bayesian classifiers. In *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, pages 239–248. Springer-Verlag, 1996.

[Pea88] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann Publishers Inc., 1988.

[PF01] Foster Provost and Tom Fawcett. Robust classification for imprecise environments. *Mach. Learn.*, 42(3):203–231, 2001.

[PK94] U. Pompe and I. Kononenko. Linear space induction in first order logic with relief. In R. Viertl. & G. Della Riccia R. Kruse, editor, *CISM Lecture Notes.* Udine, Italy, 1994.

[PK95] U. Pompe and I. Kononenko. Naive bayesian classifier within *ilpr*. In L. De Raedt, editor, *In Proc. of the 5th Int. Workshop on Inductive Logic Programming*, pages 417–436, Dept. of Computer Science, Katholieke Universiteit Leuven, 1995.

[Pla98] J. Platt. *Advances in kernel methods - support vector learning*, chapter Fast training of support vector machines using sequential minimal optimization. MIT Press, 1998.

[Pla99] John C. Platt. Fast training of support vector machines using sequential minimal optimization. pages 185–208, 1999.

[Plo70] G.D. Plotkin. A note on inductive generalization. In *Machine Intelligence*, volume 5, pages 153–163. Edinburgh University Press, 1970.

[PMS97a] Michael J. Pazzani, Subramani Mani, and William Rodman Shankle. Beyond concise and colorful: Learning intelligible rules. In *Knowledge Discovery and Data Mining*, pages 235–238, 1997.

[PMS97b] Michael J. Pazzani, Subramani Mani, and William Rodman Shankle. Beyond concise and colorful: Learning intelligible rules. In *Knowledge Discovery and Data Mining*, pages 235–238, 1997.

[Por97] M. F. Porter. An algorithm for suffix stripping. pages 313–316, 1997.

[Qui86] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, 1986.

[Qui93] J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., 1993.

[Qui96] J. Ross Quinlan. Learning first-order definitions of functions. *J. Artif. Intell. Res. (JAIR)*, 5:139–161, 1996.

[Roc71] J.J. Rocchio. *The SMART Retrieval System: Experiments in Automatic Document Processing*, chapter Relevance Feedback in Information Retrieval - chapt.14, pages 313–323. Prentice Hall, Englewood Cliffs, 1971.

[RS02] Miguel E. Ruiz and Padmini Srinivasan. Hierarchical text categorization using neural networks. *Inf. Retr.*, 5(1):87–118, 2002.

[Sah96] Mehran Sahami. Learning limited dependence Bayesian classifiers. In *Second International Conference on Knowledge Discovery in Databases*, 1996.

[SB88] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[SC96] F. Y. Shih and S.S. Chen. Adaptive document block segmentation and classification. *IEEE Trans. on Systems, Man, and Cybernetics - Part B*, 26(5):797–802, 1996.

[Seb02] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[Sha90] Mary Shaw. Prospects for an engineering discipline of software. *IEEE Softw.*, 7(6):15–24, 1990.

[SJ03] Y. Shen and J. Jiang. Improving the performance of naïve bayes for text classification, cs224n spring. Technical report, Stanford University, 2003.

[SKM99] A. Srinivasan, R. D. King, and S. Muggleton. The role of background knowledge: using a problem from chemistry to examine the performance of an ilp program. In *Technical Report PRG-TR-08-99*. Oxford University Computing Laboratory, 1999.

[SL01] Aixin Sun and Ee-Peng Lim. Hierarchical text classification and evaluation. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 521–528. IEEE Computer Society, 2001.

[SS00] Robert E. Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Mach. Learn.*, 39(2-3):135–168, 2000.

[SSS98] Robert E. Schapire, Yoram Singer, and Amit Singhal. Boosting and rocchio applied to text filtering. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 215–223. ACM Press, 1998.

[SSV+02] S. Shekhar, P. R. Schrater, R.R. Vatsavai, W. Wu, and S. Chawla. Spatial contextual classification and prediction models for mining geospatial data. *IEEE Transactions on on Multimedia*, 4(2):174–188, 2002.

[Sta96] I. Stahl. Predicate invention in inductive logic programming. In L. De Raedt, editor, *Advances in Inductive Logic Programming*, pages 34–47. IOS, 1996.

[SVAP04] D. Sona, S. Veeramachanemi, P. Avesani, and N. Polettini. Clustering with propagation for hierarchical document classification. In Marco Gori, Michelangelo Ceci, and Mirko Nanni, editors, *Proceedings of the ECML/PKDD'04 Workshop on Statistical Approaches for Web Mining*, pages 50–61, 2004.

[SW86] Craig Stanfill and David Waltz. Toward memory-based reasoning. *Commun. ACM*, 29(12):1213–1228, 1986.

[TA90] S. Tsujimoto and H. Asada. Understanding multi-articled documents. In *in Proceedings of the 10th International Conference on Pattern Recognition*, pages 551–556, 1990.

[TAK02] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceeding of UAI-2002, 18th Conference on Uncertainty in Artificial Intelligence*, pages 485–492, Edmonton, Canada, 2002.

[TB03] Domonkos Tikk and György Biró. Experiment with a hierarchical text categorization method on the wipo-alpha patent collection. In *ISUMA '03: Proceedings of the 4th International Symposium on Uncertainty Modelling and Analysis*, page 104. IEEE Computer Society, 2003.

[TL02] T. Theeramunkong and V. Lertnattee. Multi-dimensional text classification. In *In Proc. of 19th International Conference on Computational Linguistics (COLING 2002)*, Taipei, Taiwan, 2002.

[TSK01] Benjamin Taskar, Eran Segal, and Daphne Koller. Probabilistic classification and clustering in relational data. In Bernhard Nebel, editor, *Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence*, pages 870–878, Seattle, US, 2001.

[TYS94] Y. Y. Tang, C. D. Yan, and C. Y. Suen. Document processing for automatic knowledge acquisition. *IEEE Transactions on Knowledge and Data Engineering*, 6(1):3–21, 1994.

[Utg94] P.E. Utgoff. An improved algorithm for incremental induction of decision trees. In *Proc. of the Eleventh Int. Conf. on Machine Learning*. Morgan Kaufmann, 1994.

[Vap95] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[VBM04] Antonio Varlaro, Margherita Berardi, and Donato Malerba. Learning recursive theories with the separate-and-parallel-conquer strategy. In *Proceedings of the ECML/PKDD'04 Workshop on Advances in Inductive Rule Learning*, pages 179–193, 2004.

[VG02] Alexei Vinokourov and Mark Girolami. A probabilistic framework for the hierarchic organisation and classification of document collections. *J. Intell. Inf. Syst.*, 18(2-3):153–172, 2002.

[VQ90] Jacques André Vincent Quint, Marc Nanard. Towards document engineering, rapport de recherce 1244. Technical report, INRIA, Rocquencourt, France, 1990.

[WBW02] G. I. Webb, J. Boughton, and Z. Wang. Averaged one-dependence estimators: Preliminary results. In *Proceedings of the Australasian Data Mining Workshop (ADM 02)*, pages 65–73. University of Technology, Sydney, 2002.

[WBWss] G. I. Webb, J. Boughton, and Z. Wang. Not so naive bayes: Aggregating one-dependence estimators. *Machine Learning*, In Press.

[WCW82] K.Y. Wong, R.G. Casey, and F.M. Wahl. Document analysis system. *IBM Journal of Research Development*, 26(6):647–656, 1982.

[WF99] I.H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, 1999.

[wis95] *RAF Technology, Inc. DAFS Library, Programmer's Guide and Reference*, August 1995.

[Wro01] S. Wrobel. *Relational Data Mining*, chapter Inductive logic programming for knowledge discovery in databases, pages 74–101. LNAI. Springer-Verlag, Berlin Heidelberg Germany, 2001.

[WS89] Dacheng Wang and Sargur N. Srihari. Classification of newspaper image blocks using texture analysis. *Comput. Vision Graph. Image Process.*, 47(3):327–352, 1989.

[WS99] M. Worring and A.W.M. Smeulders. Content based internet access to scanned documents. *International Journal on Document Analysis and Recognition*, 1(4), 1999.

[WW97] Y. Wang and I.H. Witten. Inducing model trees for continuous classes. In M. van Someren and G. Widmer, editors, *Proceedings of the 9th European Conference on Machine Learning (ECML 97)*, pages 128–137, Prague, Czech Republic, 1997.

[WWP99] Andreas S. Weigend, Erik D. Wiener, and Jan O. Pedersen. Exploiting hierarchy in text categorization. *Inf. Retr.*, 1(3):193–216, 1999.

[Yan96] Y. Yang. An evaluation of statistical approaches to medline indexing. In *Proceedings of the AMIA*, pages 358–362, 1996.

[Yan99] Yiming Yang. An evaluation of statistical approaches to text categorization. *Inf. Retr.*, 1(1-2):69–90, 1999.

[YL99] Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49. ACM Press, 1999.

[YP97] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, pages 412–420. Morgan Kaufmann Publishers Inc., 1997.

[ZE01] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 609–616. Morgan Kaufmann Publishers Inc., 2001.

[ZG02] R. Zhao and W. Grosky. Narrowing the semantic gap - improved text-based web document retrieval using visual features. *IEEE Trans. on Multim.*, 4(2):189–200, 2002.

[ZJYH03] J. Zhang, R. Jin, Y. Yang, and A. G. Hauptmann. Modified logistic regression: An approximation to svm and its applications in large-scale text categorization. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[ZW00] Zijian Zheng and Geoffrey I. Webb. Lazy learning of bayesian rules. *Machine Learning*, 41(1):53–84, 2000.

[ZWS04] Zhaohui Zheng, Xiaoyun Wu, and Rohini Srihari. Feature selection for text categorization on imbalanced data. *SIGKDD Explor. Newsl.*, 6(1):80–89, 2004.