

ALGORITHMS FOR NON-PARAMETRIC CLASSIFIERS IN MULTI-RELATIONAL DATA MINING

By

Trilce Marie Encarnación Rivera

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Scientific Computing

UNIVERSITY OF PUERTO RICO

MAYAGÜEZ CAMPUS

December 2006

Approved by

Edgar Acuña Fernandez, Ph.D.
President, Graduate Committee

Date

Paul Castillo, Ph.D.
Member, Graduate Committee

Date

Pedro Vasquez, D.Sc.
Member, Graduate Committee

Date

Héctor Carlo, Ph.D.
Representative of Graduate Studies

Date

Luis F. Cáceres Duque, Ph.D.
Acting Department Chair

Date

ABSTRACT

ALGORITHMS FOR NON-PARAMETRIC CLASSIFIERS IN MULTI-RELATIONAL DATA MINING

By

Trilce Marie Encarnación Rivera

Dissertation Presented to the Graduate School of the University of Puerto Rico
in Partial Fulfillment of the Requirements for the Degree of Master of Science

December 2006

Directed By: Edgar Acuña Fernandez, Ph.D.

Department of Mathematical Sciences

Over the last decades, due to the advances in information technologies, both the industrial and scientific communities have acquired large volumes of data in digital form. Most of these data sets are stored using relational databases consisting of multiple tables and associations. Moreover, the data used in the fields of bio-informatics, computational biology, HTML and XML documents are relational in nature. However, most of the existing approaches to knowledge discovery in databases, assume that the data are stored in a single table. Therefore, new algorithms are needed in order to exploit the relational information provided in these data sets. This thesis proposes two novel solutions to the task of supervised classification in relational domains, based on traditional non-parametric classifiers and built upon relational algebra. The first approach is based on Kernel Density Estimation, and the second technique is based on Gaussian Mixture Models. Both techniques are evaluated using three real world relational data sets, drawn from the fields of organic chemistry, medicine and genetics.

RESUMEN

ALGORITMOS PARA CLASIFICADORES NO PARAMÉTRICOS EN MINERÍA DE DATOS MULTI-RELACIONAL

Por

Trilce Marie Encarnación Rivera

Disertación Presentada a Escuela Graduada de la Universidad de Puerto Rico
como requisito parcial de los Requerimientos para el grado de Maestría en Ciencias

December 2006

Consejero: Edgar Acuña Fernandez, Ph.D.

Departamento de Ciencias Matemáticas

Avances en las tecnologías de información han hecho posible que en las últimas décadas se hayan generado grandes volúmenes de datos de manera digital. La mayoría de estos conjuntos de datos están almacenados en bases de datos relacionales, utilizando múltiples tablas y asociaciones. Por otro lado, existen otros datos cuya naturaleza es relacional, como son los generados en los campos de bioinformática, biología computacional, documentos de HTML y XML. A pesar de esta gran cantidad de datos relacionales disponibles, la mayoría de los enfoques de minería de datos existentes asumen que los datos se encuentran almacenados en una sola tabla o matriz de datos. Por lo tanto, nuevos algoritmos deben desarrollarse para explotar la información relacional que proveen estos datos. Esta Tesis propone dos nuevas soluciones al problema de clasificación supervisada en bases de datos relacionales, estas soluciones surgen como extensiones a técnicas no-paramétricas tradicionales. El primer enfoque está basado en estimación de densidad a base de Kernel y la segunda solución se cimienta sobre la estimación de densidad basada en modelos de mezclas Gaussianas. Ambos enfoques son evaluados experimentalmente utilizando tres conjuntos de datos conocidos, estos conjuntos son extraídos de los campos de bioinformática, medicina y genética.

Copyright © 2006

by

Trilce M. Encarnación Rivera

Acknowledgments

I would like to express my deepest gratitude to my advisor, Dr. Edgar Acuña, for his support and tutelage. I truly appreciate his dedicated collaboration in this research, and I will forever be indebted to him for introducing me to the stimulating world of research.

Next, I wish to thank Dr. Paul Castillo and Dr. Pedro Vasquez for serving as members of my graduate committee, and for providing me with their valuable advice and expertise during the course of my graduate studies.

I would like to give thanks to my family for being my strength and inspiration. To my Grandmother, Teté, for providing the foundation to all my accomplishments. To my Mother, for teaching me to ask *why* and giving me the freedom to find the answers. To my uncle, Gastón Rivera, for teaching me the work ethics that have allowed me to be successful in this endeavor. To my stepfather, Harry Carbonell, for his love and support all these years. Lastly, I have to give a special acknowledgment to Sanyia Flavia, for being a constant source of motivation and guidance through my time in Puerto Rico.

Also, I sincerely appreciate the friendship and collaboration that I have found in my fellow CASTLE research group members and the graduate students of the Mathematics Department.

Finally, as testimony of my Faith, I have to thank God for showing me the way to improving myself with each day of my life.

This work was funded by The Mathematics Department of the University of Puerto Rico at Mayagüez and The Office of Naval Research (ONR) under grant number N00014-03-1-0359.

Contents

List of Tables	x
List of Figures	xi
1 Introduction	1
2 Multi-Relational Data Mining	3
2.1 Introduction	3
2.2 Inductive Logic Programming	3
2.3 Propositionalization Approaches	6
2.3.1 General-purpose Feature Construction	7
2.3.2 Special-purpose Feature Construction	8
2.4 Multi-Relational Distance Based Methods	9
2.5 Probabilistic Relational Models	11
2.6 Multi-Relational Tree-Based Approaches	12
3 Multi-Relational Data Mining Framework	14
3.1 Introduction	14
3.2 Relational Data	15
3.2.1 First-order logic Knowledge Bases	16
3.2.2 Graph Based Representation	17
3.2.3 Relational Databases	17
3.3 Concepts of Relational Databases	18

3.4	Implementing Relational Databases	20
3.5	Structured Query Language	22
4	Multi-Relational Supervised Classification Based on Kernel Density Es- timation	24
4.1	Supervised Classification	24
4.1.1	Multi-Relational Classification	26
4.1.2	Training Sample	26
4.1.3	Test Sample	27
4.1.4	Error Rate Estimators	27
4.2	Multi-Relational Kernel Density Estimation	28
4.2.1	Introduction	28
4.2.2	Multi-Relational Kernel Density Estimation	32
4.2.3	Supervised Classification using Multi-Relational Kernel Density Es- timation	34
5	Multi-Relational Supervised Classification based on Gaussian Mixture Models	35
5.1	Introduction	35
5.1.1	Finite Mixture	36
5.2	The EM Algorithm	37
5.2.1	Definition of the EM Algorithm	39
5.2.2	Number of Components in the Mixture	41
5.3	Multi-Relational Classifier Based on Gaussian Mixture Models	42
6	Experimental Results	44
6.1	Introduction	44
6.2	Relational Data Sets	45
6.2.1	Mutagenesis Database	45
6.2.2	Prediction of Protein/Gene Localization and Function	46
6.2.3	Medical Domain - Prediction of Thrombosis	48

6.3	Implementation Details	49
6.4	Multi-Relational Kernel Density Estimation	50
6.5	Multi-Relational Gaussian Mixture Models	52
7	Conclusions and Future Work	55

List of Tables

6.1	Summary of Relational Data Sets	44
6.2	Experimental Results for Multi-Relational Kernel Classifier	51
6.3	Experimental Results for Multi-Relational Gaussian Mixtures Classifier . .	53
6.4	Running Time for the Implemented Methods	54
6.5	Performance Comparison with Best Known Results	54

List of Figures

3.1	Propositional Representation of University Database	15
3.2	Example of Relational Schema for the University Database	16
3.3	Physical Implementation of the Schema for the University Database	21
4.1	Histogram Example	29
4.2	Example of Kernel Density Estimation in One and Two Dimensions	30
4.3	Algorithm for Multi-Relational Supervised Classification based on Kernel Density Estimation	34
5.1	Multi-Relational Gaussian Mixture Classification Algorithm	43
6.1	Relational Schema for Mutagenesis Data Set	45
6.2	Physical Implementation of for Mutagenesis Database	46
6.3	Relational Schema for the KDD Cup 2001 Data Set	47
6.4	KDD Cup 2001 Database	48
6.5	Relational Schema of Thrombosis Data	49
6.6	Thrombosis Database	50
6.7	EM Algorithm	52

Chapter 1

Introduction

Over the last decades, due to the advances in information technologies, both the scientific and industrial communities have acquired large volumes of data in digital form. These databases have become increasingly large, and thus more difficult to process with the available technologies. The field of Knowledge Discovery in Databases (KDD) has arisen from the need to obtain useful information from these databases, and since its beginning it has generated a large body of research. In general, a knowledge discovery process consists of several steps. The central step in the KDD process is Data Mining, which is defined as the search for interesting patterns and important regularities within large databases [26].

Data mining seeks to generate similar information to the one that a human expert could produce. In order to achieve this goal, data mining algorithms draw techniques from several fields including neural networks, pattern recognition, visualization, high performance computing, and inductive logic programming, among others. More formally, data mining is considered to be situated at the intersection of the scientific fields of statistics, database systems, pattern recognition and machine learning.

Multi-Relational data mining (MRDM) approaches look for patterns in relational data [25]. In recent years, the most common types of patterns and approaches considered in data mining have been extended to the multi-relational case, including association rules, decision trees, clustering, and distance based methods, among others. Traditionally, data mining algorithms have focused in searching for patterns within a single data table. This

approach is called attribute-value learning or propositionalization, where each example is characterized as a fixed set of attributes for which values are given. However, for reasons of storage optimization and access efficiency, most existing real world databases are not stored as a single table, but as several tables called relations [16]. As a result, the task of learning from multiple tables (relations) has begun to receive significant attention in the literature.

One of the main tasks in data mining is supervised classification, whose goal is to induce a predictive model from a set of training data. This task aims to estimate a function, given data from a number of training examples, that predicts a class label value for any input attribute vector. Thus far, multi-relational data mining techniques have been successfully used in classification tasks, with the majority of the solutions implemented in order to solve this problem based on probabilistic relational models. This thesis proposes two novel solutions to the task of supervised classification in relational domains. The solutions are based on traditional non-parametric classifiers.

The thesis is organized as follows: Chapter 2 overviews the relevant literature in the multi-relational data mining field, Chapter 3 introduces the multi-relational data-mining framework; Chapter 4 provides the definition and scope of supervised classification and introduces the application of kernel density estimation to the task of multi-relational supervised classification; Chapter 5 covers the theory related to multi-relational supervised classification based on Gaussian Mixture models; Chapter 6 presents the experimental results obtained with the described classification techniques; Chapter 7 concludes with a discussion of the main results achieved in this research and an outline for proposed future work.

Multi-Relational Data Mining

2.1 Introduction

Multi-Relational data mining looks for patterns that involve multiple relations in a relational database [24], its main difference with traditional data mining approaches is that it does not need to transform the data into a single table, it learns from the data in its original form preserving its structure and incorporating such structure into the learning process. Typically, most relational data mining algorithms have been upgraded from the single-table case. The techniques available in the literature have been mainly developed in the field of inductive logic programming (ILP), however, the most recent publications show a growing interest in the development of statistical relational models. This chapter presents a brief review of the literature, introducing the main approaches available in relational data mining.

2.2 Inductive Logic Programming

Since its inception [46], ILP has been one of the first approaches taken in relational learning and one of the most expanded. However it has not been widely accepted for the development of relational data mining solutions for knowledge discovery in databases, for the following reasons. The lack of equivalence among the different ILP engines in terms of input specification [36], the language bias and the lack of validation of relational database theory.

Inductive Logic Programming has been deemed as the intersection of Machine Learning and Logic Programming. In machine learning the *learner* generates new concepts or knowledge, based on examples provided by the *teacher* [44]. This form of knowledge discovery is called learning from examples, where the teacher provides examples, and the learner abstracts what is common to these examples to a generalization. In ILP systems, these examples are given to the learning system in the form of logic programs expressed in a logic programming language such as `Prolog`. Moreover, the concepts which the learning system develops from the examples are also expressed in the same language. This is an advantage of ILP systems, because the system can use the information it has previously learned as background knowledge for future learning; this is known as incremental learning.

The ILP paradigm is based on *inductive learning*. Induction is a machine learning technique that focuses on inductive inference, which involves unsound conjectures based on statistical support from data [45]. The models produced by inductive algorithms are able to express their findings in more complex structures than models which generalize specific instances. Logic programming is the programming paradigm that uses first order logic to represent relations between objects, patterns discovered by this paradigm are expressed as logic programs, which constitute of a set of *clauses*.

The ILP paradigm is best explained by an example [23]:

Let

$$E^+ = \{daughter(Mary, Ann), daughter(Eve, Tom)\}$$

be the positive examples of the relation daughter;

$$E^- = \{daughter(Tom, Ann), daughter(Eve, Ann)\}$$

be the negative examples of the same relation;

And

$$B = \left\{ \begin{array}{l} \{mother(Ann, Mary), mother(Ann, Tom), \\ father(Tom, Eve), father(Tom, Ian), \\ female(Ann), female(Mary), female(Eve), \\ male(Ian), male(Tom), \\ parent(X, Y) \leftarrow mother(X, Y), \\ parent(X, Y) \leftarrow father(X, Y)\} \end{array} \right.$$

be the background knowledge, where the relations daughter, mother, father, female, male, and parent have the common meaning.

Then, the goal of this ILP problem is to learn the concept daughter. For predictive ILP, a solution could be the following synthesized clause:

$$daughter(X, Y) \leftarrow female(X), parent(Y, X)$$

Or a set of definite clauses:

$$daughter(X, Y) \leftarrow female(X), mother(Y, X)$$

$$daughter(X, Y) \leftarrow female(X), father(Y, X)$$

In spite of its widely expressive language, ILP techniques pose difficulties to its adaptation due to their limited use of relational database capabilities. In the few implementations where the ILP system uses relational database engines, the efficiency of the techniques is significantly increased. Therefore, three ways have been presented for such connection [7]:

1. The simplest and straightforward manner is to pre-process the relational data into Prolog syntax.

2. In the second approach, a relational database is given together with the `Progol` knowledge base. Each time a clause is evaluated, `Progol` opens a connection to the database and makes the corresponding query to determine whether the predicate $p(a, b)$ represents a positive or negative example.
3. The last solution takes the next step exploiting the close relationships between first order logic and relational databases, where a logic predicate is defined as the relation between its arguments (which are, in turn, attributes in relational databases) and between logical clauses and relational database queries. Therefore, entire logical clauses (not only a literal at a time) can be translated into SQL statements and submitted to the database to obtain the necessary statistics.

There are several implementations of ILP systems available, the most well known is `Progol` [45]. Another very popular system is `FOIL` [51] which is one of the first ILP systems using top-down induction. Other implementations are `Claudien`, `ICL` and `Tilde` [18]. Within the ILP systems, there also exists a distinct technique for induction called learning from interpretations, or descriptive ILP. This setting has inspired the shifting from a pure logical search space to the one consisting exclusively of database queries in relational algebra ([7], [35]).

2.3 Propositionalization Approaches

Propositionalization approaches to relational data mining seek to transform a relational representation of a learning problem into a propositional (featured-based, attribute-based) representation. Experiments have shown that many data mining applications can be successfully treated with this technique, without significant loss of their predictive performance [42]. The premise in propositionalization techniques is that if enough thought is dedicated to the construction of the features that represent the relational domain, any relational data mining task can be solved by simple propositional rule learning systems.

Careful selection of the features that will compose the predictive space is required. These features consist of a conjunction of literals which share variables that refer to parts of the individual to be classified [39]. Therefore, for *propositional representations* it is assumed that the examples or background knowledge is composed of feature-vectors of fixed size. This means that all examples can be described using the same set of features, hence the parallelism to the attribute-value representation. Therefore, *propositionalization* is defined as the representation change from a relational representation to a propositional one.

The main problem with these methods is the fact that they construct the feature space in advance of the learning process. Therefore, they cannot detect the need for a representation change, specially requirement of structural changes in order to avoid serious problems such as overfitting. Some solutions have been proposed to minimize these effects, including the creation of a class-driven feature space [38]. In the subsequent paragraphs we introduce some of the existing approaches to propositionalization found in the literature.

2.3.1 General-purpose Feature Construction

These approaches are designed to work with any type of relational domain. The most representative example of this technique is the LINUS system [43], which was the first system to transform a relational representation into a propositional representation. After propositionalization, the system offers the users the choice of a number of propositional algorithms including decision trees and rule induction.

Stochastic propositionalization [40] is another general-purpose technique that finds a set of features which together possess good discriminatory power. Stochastic search is used to find clauses of arbitrary length, where each clause corresponds to a binary feature in the resulting model. The removal of clauses in each generation of the search is done probabilistically, with probability proportional to the fitness of individual clauses. This method

is able to search deeper than other propositional approaches, but it cannot guarantee that the propositionalization obtained is optimal, or even complete. Implementations of this technique include the work by Srinivasan and King [58], the authors proposed a method of stochastic propositionalization that works relatively well for all types of background knowledge; the method builds the feature space based on the hypotheses returned by the `Prolog` language. Another implementation is the `WARMR` algorithm [20], whose objective is to find association rules over multiple relations; `WARMR` achieves its goal by means of detecting frequent successful `Datalog` queries, which are based in a `Prolog` like language.

2.3.2 Special-purpose Feature Construction

Special-purpose feature construction approaches are developed with either domain dependency or assume a strong declarative bias, or are otherwise applicable to a limited problem class. Despite the fact that they are designed for very specialized tasks, any study of propositionalization techniques should cover these methods because they have been very successful in solving the problems that motivate them.

The first attempt at a special form of propositionalization was done in the work by Turney [61]; here `Progol` programs were designed to solve East-West Challenge, classifying trains as eastbound or westbound. The algorithm proposed introduced new combinations of up to three `Progol` literals for each feature, to minimize complexity, the construction of the feature space was made by a decision tree induction algorithm based on the cost computed for each feature.

Another successful approach is the `SUBDUE` algorithm [12]. This technique was designed for structure discovery in graphs, and is an Minimum Description Length (MDL) based algorithm which finds substructures that compress the original data and represents the structural concepts existent in the data. This system is capable not only of finding linearly connected fragments, but is also capable of identifying subgraphs. Recent comparisons [32] of this approach have proven to be more efficient in learning structurally

complex patterns, but the algorithm remains limited in its use of the background knowledge provided by the relational domain.

2.4 Multi-Relational Distance Based Methods

Distance based methods have been very popular for data analysis tasks, these methods assume that it is possible to compute for each pair of objects in a domain their mutual distance, also called similarity measure. The method seeks to define the distance between individuals within a particular domain. Once this distance is defined, most data mining tasks can be executed in a relatively simple fashion. In order to use these techniques in relational domains, a common approach is to extract a vector of features from the database objects and then use the Euclidean distance or some other norm between those feature vectors as similarity measure. But this often results in very high dimensional feature vectors, which are not efficiently handled by the learning algorithms. Other solutions have been proposed to compute similarity measures in multi-relational data mining, in this section we aim to introduce some successful implementations of these approaches.

The first-order distance measure was first introduced by [8], it was developed for the instance based learning system RIBL2. For this technique, each instance or observation is described as a set of facts, which are the queries or clauses related to the individual it represents. In order to select a suitable similarity function, the distance measure here uses the idea of computing distances by recursively comparing the components of the first-order instances until one can finally fall back on propositional comparisons of elementary features. This distance was developed for the multi-relational implementation of the k-nearest neighbor algorithm (kNN), which is a method for classifying objects based on closest training examples in the feature space. The training examples are mapped into multidimensional feature space and the space is partitioned into regions by the class labels of the training samples. Then each point in the space is assigned a class label, this value is selected as the most frequent class label among the k nearest training samples.

The neighbors are identified using a similarity measure, in this case first-order distance; in propositional domains Euclidean distance is commonly used.

Clustering is a distance based technique that has been extensively studied, and has been also implemented in relational domains. An efficient multi-relational clustering technique is provided in the RDBC system [33]. In this work they present a bottom-up agglomerative clustering algorithm for first-order representations or relational data, that relies on distance information only. The algorithm features a novel parameter-free pruning measure for turning the hierarchical cluster tree into a single-level cluster. This algorithms predictions are based on the above mentioned first-order distance measure.

Another popular clustering technique is k-means, which is a variant of the expectation-maximization algorithm in which the goal is to determine the k means of data generated from gaussian distributions. The k-means clustering technique groups objects into k partitions based on their attributes, assuming that the object attributes form a vector space. This techniques has been extended to relational domains with the implementation of FORC [34], which also uses first-order distance. However, in this implementation it is required to make assumptions about the available representation, otherwise the computational complexity suffers greatly.

For graphical representations of relational data, the similarity measures defined are computationally extremely complex (i.e. NP-complete), this makes them unsuitable for data mining in large databases. When deriving a distance measure for attributed graphs, two major aspects must be taken into account: The first one is the structural similarity of graphs and the second one is the similarity of the attributes. Additionally, this two aspects must be weighted, because it is highly application dependent to what extent the structural similarity determines the object similarity and to what extent the attribute similarity has to be considered. In the work by Kriegel [41], a new similarity measure for attributed graphs is presented, called matching distance. Here, they propose a function

that instead of matching the vertices of two graphs, the cost function matches the edges of the two graphs and then derives a minimal weight maximal matching between the edge sets. This way, not only the attribute distribution, but also the structural relationships of the vertices are taken into account

Multi-Relational distance based methods offer a lot of potential, but pose serious concerns about scalability. Most of these methods must store all of the instances and calculated distances until classification time. Moreover, computing distances between structurally rich objects is more expensive than computing the similarity measure in propositional domains. Therefore, these methods are more suitable for small domains.

2.5 Probabilistic Relational Models

Probabilistic Relational Models (PRMs) are an extension of Bayesian Networks to relational domains. In this approach, the nodes of the network are the attributes of a relational database and the edges represent probabilistic dependencies between attributes. The models are expressed as a probability distribution over the attributes and the relations between them [27].

The resulting models have two main components: A relational component that describes the relational schema of the domain, this would be the network structure in Bayesian Networks; The other component is a probabilistic component that describes the probabilistic dependencies that hold in the domain. The main work in this subject has been done by Getoor [28], her model exploits the conditional independence existent in most domains.

The task in this algorithms consists in learning the two components of the PRMs. The parameter estimation is an easier task than learning the structure. The key function used to estimate the parameters in the PRM is the likelihood function. The higher the

value of this function, the better the model predicts the data. For the structure learning task three components need to be defined: the hypothesis space (the structures that must be considered by the algorithm); a scoring function to evaluate each hypothesis relative to the data; and the search algorithm. The search algorithm is usually heuristic, for example the greedy hill-climbing search.

2.6 Multi-Relational Tree-Based Approaches

Decision tree learning is a common method used in data mining. Here, a decision tree describes a tree structure wherein leaves represent classifications and branches represent conjunctions of features that lead to those classifications. The tree itself is a predictive model that maps observations about an object, to conclusions about the target value. In classification tasks, the target value represents the class label. In this model, the leaf represents the predicted value for the target variable, given the values of the features represented in the path from the root. Decision trees are also a descriptive means for calculating conditional probabilities. In relational domains, there are two very distinct techniques used for tree induction, these are tied to the form in which the relational domain is represented.

In first-order logic databases, **S-Cart** [50] is an algorithm that learns a theory for the prediction of either discrete classes or numerical values, from examples and relational background knowledge [31]. The result is a classification tree that expresses its findings as a conjunction of literals in each node, these literals consist of an atomic formula or its negation (also represented as database queries). The class value or the numerical response is stored in each leaf.

Multi-relational decision tree learning algorithms construct decision trees whose nodes are multi-relational patterns i.e., selection graphs. Knobbe [37] proposed a multi-relational decision tree induction algorithm, based on the logical decision tree induction

algorithm called TILDE proposed by Blockeel [6]. Later, this algorithm was implemented in the work by Atramentov et. al. [3], and called MRDTL. This technique deals with records in relational databases, the nodes contain selection graphs. Selection graphs are directed graphs whose nodes contain a set of tuples that fulfill a given condition. These selection graphs are added to the tree through a process of successive refinement until some termination criterion is met, in the case of the supervised classification task, the stopping criterion is the correct classification of instances in the training set.

Experiments performed with tree based approaches to multi-relational data mining show that they compete with other existing methods in terms of predictive accuracy. However, since relational formalisms open a much larger search space for these methods, the algorithms have a high computational cost.

Multi-Relational Data Mining Framework

3.1 Introduction

Relational data often have irregular structures and complex dependencies that contradict the assumptions of conventional machine learning techniques. The field of Multi-Relational data mining aims to improve the existing techniques in order to discover patterns within relational databases. Algorithms in the attribute-value paradigm assume that the data instances are recorded in homogeneous structures, but the objects stored in relational databases are usually more varied and complex. The ability to generalize across heterogeneous data instances is the defining trait of multi-relational learning algorithms.

Representation is fundamental in the process of knowledge discovery. In a multi-relational framework, the notion of a database begins with constructs [49]. Constructs can be logical objects such as data models, rules, subtypes or physical objects like tables and records; The primitives used here are objects and their attributes. Independent of what kind of database model is assumed to be used, at the time of designing a database, the universe X of objects that will be described in the database should be distinguished. Usually such a universe is considered as consisting of heterogeneous sets of objects. Hence, a number of types of objects are distinguished. For each type of objects in X , some at-

Popularity	ProfessorID	CourseID	Difficulty	Satisfaction	...
8.5	P10106	Q0101	8.0	7.3	...
8.5	P10106	Q0101	8.0	9.5	...
8.5	P10106	Q0101	8.0	9.7	...
4.3	N10106	M2101	9.5	4.1	...
...

Figure 3.1: Propositional Representation of University Database

tributes have to be predetermined and to each attribute a particular meaning is assigned. Loosely speaking, the conceptual domain of a database is defined by specifying a universe X and its distribution over sets of different types $Xt, t \in T$, and then assigning to each type $t \in T$ a set of attributes At .

3.2 Relational Data

Relational data records characteristics of heterogeneous objects and persistent relationships among those objects. Relational data is often stored in multiple tables, with separate tables recording the attributes of different object types and the relationships among these objects. In contrast to propositional data, which records characteristics of a single set of homogeneous objects and is often stored in a single table.

Consider the task of determining the popularity of professors based on the data collected at a university between the registered students. A propositional representation of the available data is given in Figure 3.1. Now, in relational domains the language of representation is richer and therefore is able to portray the relationship between the different entities in the database. In figure 3.2, the same university database is modelled as a set of separate entities and the links which represent their relationship.

A data model is not just a way of structuring data: it also defines a set of operations

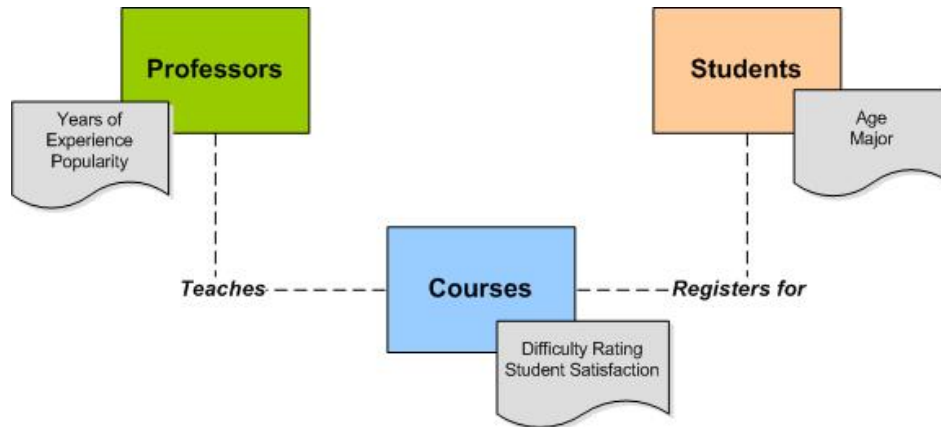


Figure 3.2: Example of Relational Schema for the University Database

that can be performed on the data. The relational model, for example, defines operations such as selection, projection, and joins. Various techniques are used to model the data structure in relational domains. Most database management systems are built around one particular data model, but recent developments are making possible the support for more than one model within the same system. However, this is not the case in the development of data mining algorithms where representation is an intrinsic part of the technique, and the choices that are made have a significant effect on performance. According to Neville [47], there are three main representations of relational data for multi-relational data mining: relational databases, graphs and first-order logic knowledge bases.

3.2.1 First-order logic Knowledge Bases

Here, the relational domain is represented as set of first-order logic statements. A first-order database language serves to describe a database structure and properties of the database constructs [54]. In this approach the database schema is defined as a set of formulas in the database language. The notion of a database schema plays the same role as the notion of theory in predicate calculus; a schema can contain formulas representing integrity constraints specifically for an application and the constraints specifically for a type of database, all expressed in the same database language.

The language used to represent instances is based on first-order logic, and is not restricted toward any particular type of database domain. In this language, the data can express the special requirements resulting from either relational, hierarchical or network databases. The database can be considered as an interpretation of some formulas given in this language, moreover, the same database language can be used to formulate user queries. Therefore, this kind of representation is used as a natural end-user interface to databases of different types.

3.2.2 Graph Based Representation

This approach represents the relational domain by means of a directed, attributed hypergraph $G_D = (V_D, E_D)$ with V nodes representing objects and E hyperedges representing relations, with one or more connected components. Each node $v_i \in V_D$ and edge $e_j \in E_D$ are associated with a type $T(v_i) = t_{vi}$, $T(e_j) = t_{ej}$. Each item type $t \in T$ has a number of associated attributes $X_t = (X_1^t, \dots, X_m^t)$. Consequently, each object v and link e are associated with a set of attribute values determined by their type $X_v^{t_v} = (X_{v1}^{t_v}, \dots, X_{vm}^{t_v})$.

Graph-based approaches to multi-relational data mining represent examples, background knowledge, hypotheses and target concepts as graphs. The objective of these approaches are the search for graph patterns which are frequent or which compress the input graphs, moreover, the distinction of positive and negative examples through the patterns available in the graphs is another goal of these techniques.

3.2.3 Relational Databases

Relational databases consist of a set of database tables with entities E and relations R . Items of a common type are stored in a separate database table with one field for each attribute. In this representation, the relational domain is defined by a schema, which describes entities, their attributes and relations between them. A common misconception is that a relational database management system is also a relational database, but soft-

ware such as Oracle, Microsoft SQL Server, PostgreSQL, and MySQL are not relational databases but tools in which one can implement relational databases. This work considers relational data represented using the relational schema, more information about this form of representation is given in the next section.

3.3 Concepts of Relational Databases

The fundamental assumption of the relational model [11] is that all data are represented as mathematical n -ary relations, an n -ary relation being a subset of the Cartesian product of n domains. In the mathematical model, reasoning about such data is done in two-valued predicate logic, meaning there are two possible evaluations for each proposition: either true or false. However, in relational domains there usually exists a third value such as unknown, which is often associated with the concept of NULL. Thus, although logic (which is inherently two-valued) is an important part of the relational model, a system that uses a form of three-valued logic can still be considered relational. In this form of representation the data is operated by means of a relational calculus or algebra, both being equivalent in expressive power.

A *Relational Database* is conformed by a collection of relations (frequently called tables), the term *Relation* is used here in its accepted mathematical sense.

Definition Given sets S_1, S_2, \dots, S_n , (not necessarily distinct), R is a *relation* on these n sets if it is a set of n -tuples each of which has its first element from S_1 , its second element from S_2 , and so on. We shall refer to S_j as the j th domain of R ; R is said to have degree n .

Relations are implemented by means of *Tables*, which organize data in rows and columns. All of the data stored in a column should be in the same domain (i.e. data type). In the relational model, it is stated that tuples should not have any ordering. This means both that there should be no order to the tuples, and that the tuples should not

impose an order of the attributes. This requirement is not universally achieved: All data stored in a computer has to have an order, as the memory of a computer is linear. Also, when the data is returned, there must be an order in which the data is returned because all transfer protocols are linear. The SQL standard requires columns to have a defined order, but the orders imposed must not impact performance, and they should never change the result of a query on the database.

A tuple usually represents some object and its associated data, regardless if that object is a physical one or is used to represent a concept. An important aspect of implementing tuples are the *key* attributes, these attribute represents a type of constraint which requires that the object isn't duplicated. In practice a key could be comprised of one or more attributes, in the case when a key covers more than one attribute is called a compound key. Other special attributes are *foreign keys*, which are not keys by the previous definition; Rather, a foreign key is a reference to a key in another table. Meaning that the referencing tuple has as part of its attributes, the values of a key in the referenced tuple that corresponds to the relationship.

This objects are all expressed in the relation schema, that defines the structure of the database [28]:

Definition A *relation schema* \mathbf{R} consists of a set of tables $\mathbf{R} = \{R_1, R_2, \dots, R_n\}$. Each table R is associated with attributes of three types: a single primary key $R.K$, a set of foreign keys $\mathbf{F}(R)$, and a set of descriptive attributes $\mathbf{A}(R)$. Each foreign key $R.F$ is associated to a table to which it points, $DOM[R.F] \in \mathbf{R}$. Each descriptive attribute $R.A$ is associated with a domain of possible values $\mathbf{V}(R.A)$.

From this definition we construe that a key requires that the cardinality of the relation should be equal to the cardinality of the relation projected over the columns of the key. Therefore, a key in this context refers to any set of attributes which uniquely span

the relation. Then, a database constitutes an instance of the relational schema:

Definition A *database* \mathbf{D} over \mathbf{R} consists of a set of tuples $\mathbf{T}[R]$ for each table R . For each $t \in \mathbf{T}[R]$:

- The primary key $t.K$ is unique within R .
- For each $F \in \mathbf{F}(R)$, $t.F$ is the primary key of some tuple in $\mathbf{T}[S]$ where $S = \text{Dom}[R.F]$.
- For each $A \in \mathbf{A}(R)$, $t.A$ is a value in $\mathbf{V}(R.A)$.

The second restriction refers to *referential integrity*, this database concept ensures that relationships between tables remain consistent. When one table has a foreign key to another table, the concept of referential integrity states that you may not add a record to the table that contains the foreign key unless there is a corresponding record in the linked table. More formally: Let R be a table and let F be a foreign key in R that refers to a table S with primary key K ; then for every tuple $r \in R$ there must be some tuple $s \in S$ such that $r.F = s.K$. Although referential integrity is enforced in most real world databases, in the models we've developed in this work we allow for foreign key attributes to take the value *null*, indicating that there is no related tuple in $\text{Dom}[R.F]$.

3.4 Implementing Relational Databases

Relational Databases are implemented using a special software that supports relational modelling and operations on relational data, this software is called a *Relational Database Management System*. When implementing the relational model using a RDBMS, several considerations have to be taken into account.

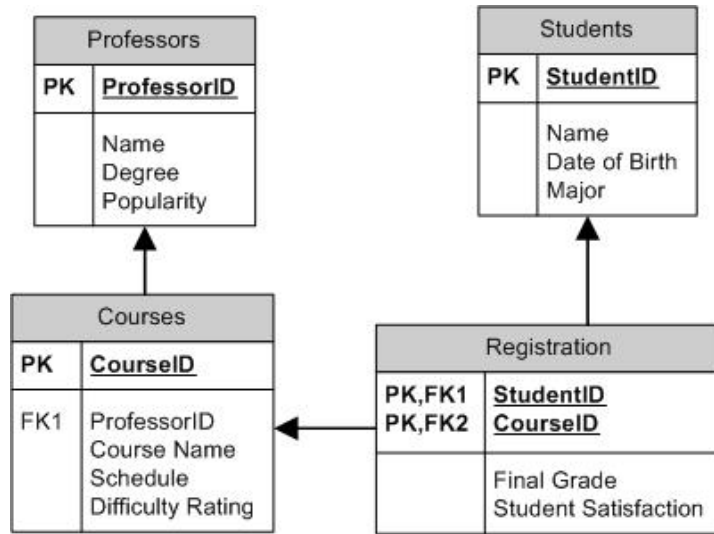


Figure 3.3: Physical Implementation of the Schema for the University Database

The natural relationship mechanism within the relational model is the primary-foreign key constraint. This binary relationship that has been explained in previous sections, can establish one-to-many and one-to-one relationship between two tables. Here, a data entity is related to another via the equality of at least one attribute value. Many-to-many relationships are modelled using a combination of two one-to-many relationships and three tables. This method establishes a many-to-many relationship between two tables. One of these tables is a *cross product* of the two tables involved within the many-to-many relationship. However, there are numerous methods used to establish a many-to-many relationship. Figure 3.3 illustrates this implementation with an example physical schema of the university domain, here, the table *registration* represents the many-to-many relationship between courses and students.

Other objects are used to implement relational databases, and although they are not intrinsic part of the relational database, its purpose is to help organize and structure the data:

- Constraints: They impose restrictions on the values that can be stored in the rela-

tions. These are usually defined in the form of expressions that result in a boolean value, indicating whether or not the constraint holds.

- **Stored procedures** A stored procedure is executable code that is associated with the database and usually store how to perform common operations, like inserting a tuple into a relation, or gathering statistical information about usage patterns. Stored procedures are not always considered part of a relational database, partially because they are not essential to the functioning of the database.
- **Indexes:** An index is a way of providing quicker access to the data in a relational database. Indexes can be created on any combination of attributes on a relation and they operate in a similar manner to how a book's index works, when tuples in a relation need to be looked up, the index is accessed and it locates the desired tuple.

3.5 Structured Query Language

In order to create, modify, retrieve and manipulate data from relational database management systems, the standard operating language is the Structured Query Language or SQL. This language is derived from the model proposed by Codd [11], and its first implementation was done by Chamberlain and Boyce [9]. Since then, it has been adopted as a standard by ANSI (American National Standards Institute) in 1986 and ISO (International Organization for Standardization) in 1987. The SQL interface is organized into the following groups:

- **Data Definition:** The *Data Definition Language* (DDL) allows the user to define new tables and associated elements. Most commercial SQL databases have proprietary extensions in their DDL, which allow control over nonstandard features.

- **Data Retrieval:** This is the most frequently used operation in transactional databases. When restricted to data retrieval commands, SQL acts as a declarative language. The main keyword is **SELECT**, which is used to retrieve zero or more rows from one or more tables in a database. Through the use of the **WHERE** clause the command restricts the output to the tuples that comply to a given condition; this command also provides for the ordering of the results, and for computing aggregate functions.
- **Data Manipulation:** These commands are used to retrieve and manipulate data and compose the standard *Data Manipulation Language* (DML). The functional capability of the commands is organized by the initial word in the statement, which is almost always a verb. The basic commands are **SELECT**, **INSERT**, **UPDATE** and **DELETE**.
- **Data Transaction:** The latter implementations of the SQL standard include a set of commands that allow the users to wrap around the DML operations. These *transactions* are a set of operations that must be executed completely or not at all.
- **Data control:** This group of SQL keywords, known as the *Data Control Language* (DCL), handles the authorization aspects of data and permits the user to control who has access to see or manipulate data within the database.

Multi-Relational Supervised Classification Based on Kernel Density Estimation

4.1 Supervised Classification

Supervised classification involves methods that automatically induce a predictive model from a set of training data. The goal is to estimate a function, that intends to predict a class label value for any input attribute vector using the data from a number of training examples. More formally, the supervised classification problem can be stated as:

“Given a finite set of classes G_1, G_2, \dots, G_g , known a priori, and a p dimensional input vector \mathbf{x} . *Supervised Classification* aims to find the relationship between the values of \mathbf{x} and the group G_i to which it belongs, based on the examples provided by the training data.”

In this setting, we assume that there is a series of *a priori* probabilities $\pi_1, \pi_2, \dots, \pi_g$, for each class label. Then, we can model the relationship between the attributes of any input vector and its class label by assuming that an element of the class $y \in 1, 2, \dots, g$ is an instance of the random variables with conditional probability function dependant on

the class $Fy(\mathbf{x})$. Given an instance of the input vector X , noted by x , the goal of classifier C is to determine its class label.

Definition A *Classifier* C is a function $C : R^p \rightarrow \{1, 2, \dots, g\}$, where $C(x)$ represents the class label assigned to the input vector \mathbf{x} .

The performance of a classifier C can be measured by the probability that a random instance from a group G_i is assigned to G_j , $j = 1, 2, \dots, g$, defined as:

$$e_{ij}(C) = Prob [C(\mathbf{x}) = j | \mathbf{x} \in G_i] \quad (4.1)$$

$$= \int_{R_j} f_i(\mathbf{x}) d\mathbf{x} , \quad (4.2)$$

where f_i is the i -th class density function and, $R_j = \{x : C(x) = j\}$.

We can estimate the probability of misclassification for any randomly selected member of a group G_i :

$$e_i(C) = \sum_{i \neq j}^g e_{ij}(C) \quad (4.3)$$

$$= \int_{\bar{R}_i} f_i(\mathbf{x}) d\mathbf{x} \quad (4.4)$$

Where \bar{R}_i is the complement of R_i ($i = 1, 2, \dots, g$).

In order to estimate the error rate of the classifier C , we compute:

$$e(C) = \sum_{i=1}^g e_i(C) \pi_i \quad (4.5)$$

Where π_i represents the prior probability of class G_i .

4.1.1 Multi-Relational Classification

The objective of multi-relational classification is to undertake classification tasks in relational databases. As was discussed in section 3.2, a schema for a relational database describes a set of tables $DB = \{T_1, T_2, \dots, T_n\}$, and a set of relationships between pairs of tables. In this setting, we have a database DB and a target relation T_{target} which includes a target variable y , representing the class label we aspire to predict.

Analogous to the propositional case, the multi-relational classification task aims to find a classifier function $C(x)$ which maps each tuple x of the target table T_{target} to its class label y . Therefore, the classifier function $C(x)$ now becomes a function dependant not only on the attributes associated to the tuples in the target table, but also to the attributes stored in the tuples of foreign key relations; in example: $y = C(x, DB, T_{target})$. In this scenario, the T_{target} table is considered the *target relation* while all others are *background relations*, which are in turn associated with the target relation by foreign key attributes.

4.1.2 Training Sample

In a supervised classification environment we have a set of data that constitutes the training sample and is used to build the classifier. In the attribute-value paradigm, the training sample is defined as a matrix \mathcal{L} whose elements are of the form (\mathbf{x}_j, y_j) , ($j = 1, \dots, n$); where \mathbf{x}_j is a p dimensional observation vector and y_j is its respective label.

In multi-relational data mining, the training data also includes the data in the tuples related to each instance in the target table. Therefore, for each tuple t in the target table, the feature space will be composed the descriptive attributes $\mathbf{A}(R)$ in the target relation T_{Target} and the descriptive attributes $\mathbf{A}'(R)$ stored in foreign key relations $T[S] \in DOM[R.F]$. The dimension of this feature space is not known beforehand, since each tuple can have a different number of related instances in foreign key relations.

4.1.3 Test Sample

The test sample contains the examples that are used to test the generated model. Its structure is the same as the training sample, and it is produced from a set of elements that are independently sampled from the same population from which the training sample was generated.

4.1.4 Error Rate Estimators

When we use a classification technique, it is important to verify that the generated model is atoned with the data. In order to evaluate the efficacy of a classifier, we have several estimators for the error rate associated to $e(C)$. The most widely used estimators are:

- Apparent error: When calculating this value, the training sample is used for building as well as for testing the classifier. Then, the error is estimated as the proportion of the misclassified instances in the training set. The expression to estimate this rate is:

$$e_A = \frac{1}{n} \sum_{i=1}^n Q(z_i, C(x_i)) \quad (4.6)$$

where

$$Q(u, v) = 0 \text{ if } u = v \text{ and } Q(u, v) = 1 \text{ for } u \neq v$$

Generally, this error rate is extremely optimistic and tends to produce bias when the number of samples is smaller than the number of predictor variables. In this case the obtained models respond to the noise present in the data rather than its structure [62].

- Test sample: In order to estimate the misclassification error, the training data is divided randomly into two sets: The first set will remain the training data and will be used to build the classifier, the remaining samples will constitute the test

sample. In order to estimate the error rate, the model obtained from the classifier is used to classify the instances in the test sample, and the error is the proportion of misclassified instances in this set. In this method, the usual distribution is $\frac{2}{3}$ of elements are assigned to the training sample, and $\frac{1}{3}$ is assigned to the test sample. This is repeated a given number of times and the average is reported.

- Cross validation: This is a more robust method to estimate the misclassification rate, here the set of target instances is partitioned in m sub-samples ($m = 10$ being the most used). Each sub-sample becomes then a test sample, and the classifier is built iteratively m times with the remainder $m - 1$ samples. Then, the error estimate is obtained as the average of classification errors for each sub-sample. The experiment is repeated a given number of times and the average is computed in order to estimate the error rate.

4.2 Multi-Relational Kernel Density Estimation

4.2.1 Introduction

At first glance, we can assume that the classification problem can be solved by correctly assigning a probability distribution $F_y(x)$ to each class label. Methods that apply this technique are called parametric, and have been efficiently employed when working with known distributions that adjust to each particular problem. However, it is not always possible to obtain these distributions, hence the use of nonparametric techniques is needed. In nonparametric classification, the main task is to efficiently estimate the probability density $f_y(\mathbf{x})$ for each class label. Popular non-parametric techniques include kernel density estimation, k-nearest neighbors, and gaussian mixture models.

Kernel density estimation [22] is a relatively simple non-parametric density estima-

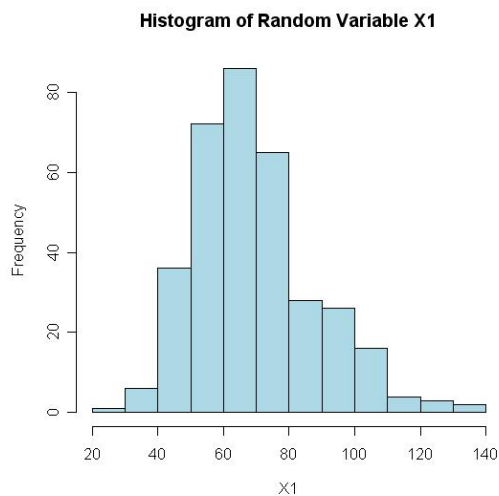


Figure 4.1: Histogram Example

tor that is frequently encountered and has been extensively used in the attribute-value paradigm. To understand kernel estimators, we first need to understand histograms, whose disadvantages provide the motivation for kernel estimators.

The construction of a histogram depends upon the the width of the bins (equal sub-intervals in which the whole data interval is divided) and the end points of the bins (where each of the bins start), the outcome is that the resulting model is not smooth, as illustrated by the example provided in figure 4.1. Kernel density estimation aims to improve this outcome. To remove the dependence on the end points of the bins, kernel estimators center a kernel function at each data point. Moreover, this estimation smooths out the contribution of each observed data point over a local neighborhood of that data point. The contribution of data point $x(i)$ to the estimate at some point x depends on how apart $x(i)$ and x are. The extent of this contribution depends on the shape of the kernel function adopted and its bandwidth. An example of kernel density estimation is shown in figure 4.2.

More formally, we can describe the process of kernel density estimation as [30]:

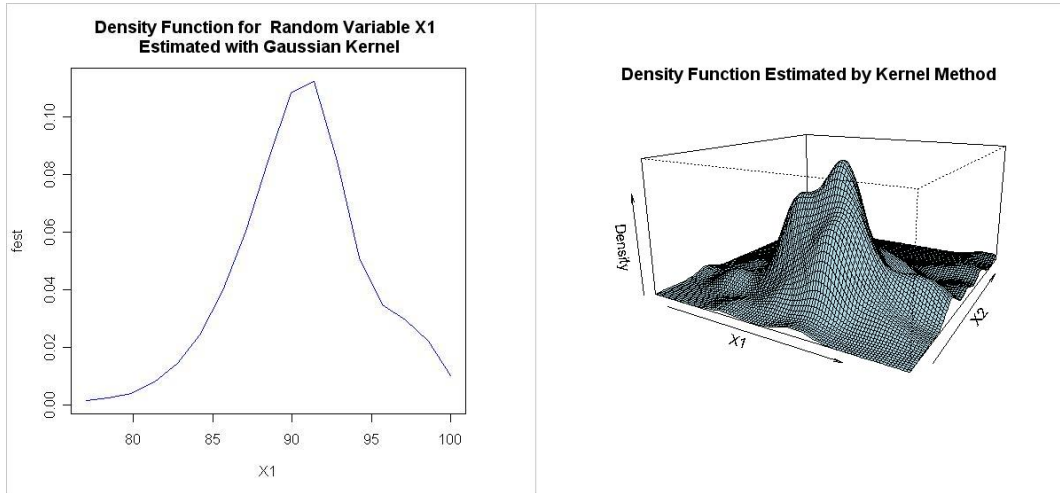


Figure 4.2: Example of Kernel Density Estimation in One and Two Dimensions

Given (x_1, \dots, x_n) , a random sample of the variable X with density function $f(x)$. Then, $f(x)$ can be estimated using the Kernel density estimation as follows:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right) \quad (4.7)$$

Where $K(\cdot)$ is a weight function, and h is called the bandwidth, which is a smoothing parameter. The function K indicates the contribution each point has over the estimated value.

$K(z)$ and h must satisfy the following regularity conditions:

$K(z)$ must be bounded and completely integrable in $(-\infty, \infty)$

$$\int_{-\infty}^{\infty} K(z) dz = 1$$

And,

$$\lim_{h \rightarrow \infty} h(n) = 0$$

Several kernel functions can be found in the literature, the most widely used in multivariate estimation are:

- Uniform Kernel:

$$K(z) = \begin{cases} 0 & \text{for } |z| > 1 \\ \frac{1}{2} & \text{for } |z| \leq 1 \end{cases} \quad (4.8)$$

- Gaussian Kernel:

$$K(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2} \quad (4.9)$$

- Triangular Kernel:

$$K(z) = \begin{cases} 1 - |z| & \text{for } |z| < 1 \\ 0 & \text{for } |z| \geq 1 \end{cases} \quad (4.10)$$

- Biweight Kernel:

$$K(z) = \begin{cases} \frac{15}{16} (1 - z^2)^2 & \text{for } |z| < 1 \\ 0 & \text{for } |z| \geq 1 \end{cases} \quad (4.11)$$

- Epanechnikov Kernel:

$$K(z) = \begin{cases} \frac{3}{4\sqrt{5}} \left(1 - \frac{z^2}{5}\right) & \text{for } |z| < \sqrt{5} \\ 0 & \text{for } |z| \geq \sqrt{5} \end{cases} \quad (4.12)$$

4.2.2 Multi-Relational Kernel Density Estimation

Multi-Relational kernel density estimation has been effectively used in the work presented by Shangai et al. [55], here the authors propose a form of kernel density estimation to directly and accurately compute the joint probability distribution of the variables within relational domains. However, this research does not explore the use of kernel density estimation for classification purposes. The authors successfully implement this technique in order to estimate the joint probability distribution in the framework of Relational Dynamic Bayesian Networks.

The authors assume independence between all the predictor variables corresponding to each foreign key relation, in order to approximate the joint distribution of the relational variables. Then, the marginal probabilities can be used to compute the joint distribution as:

$$P(X = x) = \prod_{R \in X} \left[\prod_{l,j \in X_R} P(X_{l,j} = x_{l,j}) \right] \quad (4.13)$$

where X represents the random attribute vector and x is a particular value, R is a relation in the DB . The drawback to this approach is that it can lead to inaccurate results when the independence assumption does not hold. Now, we can formulate the kernel density estimation technique proposed:

A kernel density estimator for a random vector X takes n samples and estimates X 's probability density function of f at X as:

$$\hat{f}(\mathbf{x}) = \frac{1}{n} \sum_i K(x; x_i) \quad (4.14)$$

where K is a non-negative kernel function.

The kernel function K represents a distribution over X based on the sample x_1, x_2, \dots, x_n and is typically a function of the distance between x_i and x . In multi-relational domains this approach can lead to a very large feature space, therefore, the kernel function is broken into a product of kernel functions that can be computed using any of the above mentioned multivariate kernel functions over the set of i related tables. Thus, the relational kernel function yields:

$$K(x; x_i) = \prod_R K_R(x, x_i) \quad (4.15)$$

Where $K_R(\cdot)$ is a multivariate density function that is estimated over the attributes of a relation within the relational database. Therefore, in order to estimate the probability density for the elements of the target relation, one must compute the product kernel for the density functions of all relations connected to the target relation by a foreign key

0. For each tuple $T.t$ in $T = T_{target}$
 1. For each class label $y = y_1, y_2, \dots, y_n$ compute the multi-relational probability density:
$$P(y_i, T.t) = \pi_j \hat{f}(T.t, y_i)$$

where:

$$\hat{f}(T.t, y_i) = \prod_R \frac{1}{n} \sum_{i=1}^n K_R(x, x_i)$$
 2. Assign the class label that maximizes $P(y_i, T.t)$

Figure 4.3: Algorithm for Multi-Relational Supervised Classification based on Kernel Density Estimation

relationship.

4.2.3 Supervised Classification using Multi-Relational Kernel Density Estimation

In order to use the kernel density estimation for the multi-relational classification task, we compute the conditional probability distribution of each class for every instance to be classified. Then, the class label that maximizes this probability is assigned to this instance.

This algorithm's complexity is dominated by the dimensionality of the attributes that compose the predictor space. The complexity for the algorithm is bounded by $O(n_{classes}n_{obs}n_{var})$, where n_{obs} is the number of examples, and n_{var} is the number of predictor variables in all the relations that compose the *DB*.

Multi-Relational Supervised Classification based on Gaussian Mixture Models

5.1 Introduction

Mixture models are used to model probability density functions, as a sum of parametrized functions. The resulting estimate of the probability function has the form

$$f_X(x) = \sum_{k=1}^K a_k h(x|\lambda_k) \quad (5.1)$$

where K is the number of components in the mixture model, and a_k is mixture proportion of the k -th component; and $h(x|\lambda_k)$ is a density function with parameter λ_k . These models try to mirror the behavior of populations which are composed by several distinct populations. They have been extensively used due to their flexibility, and hence, they have received a great deal of attention from researchers in both theory and experimentation.

5.1.1 Finite Mixture

The p dimensional random vector Y is distributed with a finite mixture if its density function $f(y; \Psi)$ can be expressed by:

$$f(y; \Psi) = \sum_{i=1}^k \pi_i f(y; \theta_i) \quad (5.2)$$

$$\begin{aligned} \pi_i > 0, i = 1, 2, \dots, k \\ \sum_{i=1}^k \pi_i = 1 \end{aligned}$$

Here, vector Ψ belongs to the parameter space Ω , which contains the unknown parameters of the model:

$$\Psi = (\pi_1, \pi_2, \dots, \pi_{k-1}, \theta^t) \quad (5.3)$$

where θ represents the vector of parameters $(\theta_1, \theta_2, \dots, \theta_k)$, of all known a priori parameters.

The $f(y; \theta_i)$ functions are probability density functions, and are called the components of the mixture; θ_i is the vector of unknown parameters for each density function. The mixture proportions or weights are noted by $\pi_1, \pi_2, \dots, \pi_k$.

The number k of components of the finite mixture is unknown and has to be determined taking into consideration the available data.

In particular, a mixture model can be composed of a series of multivariate gaussian density functions:

$$f(y; \Psi) = \sum_{k=1}^K \pi_k \phi_1(y; \mu_k; \Sigma_k) \quad (5.4)$$

where

$$\phi_1(y; \mu_i, \Sigma_i) = \frac{e^{[-\frac{1}{2}(y-\mu_i)^t \Sigma_i^{-1} (y-\mu_i)]}}{\sqrt{|2\pi \Sigma_i|}} \quad (5.5)$$

represents the gaussian density function with vector of means μ_i and covariance matrix Σ_i ($i = 1, 2, \dots, k$). In this case, the vector Ψ of unknown parameters is given by

$$\Psi = (\pi_1, \pi_2, \dots, \xi^t)^t \quad (5.6)$$

where ξ is composed of the elements of each vector μ_i and the elements that make up the covariance matrices $\Sigma_1, \Sigma_2, \dots, \Sigma_k$.

The density function defined in 5.4 is known as *Gaussian Mixture*, and is later used in order to build the classifier in a similar way to the Kernel Density Estimation presented in chapter 4.

5.2 The EM Algorithm

The unknown parameters of a Gaussian mixture model can be estimated by the EM algorithm; This algorithm has been thoroughly studied within propositional domains.

The idea behind this technique is the basis for its name, the goal of the Expectation-Maximization (EM) algorithm is to find the stationary points of the log likelihood function.

The original work about this algorithm is presented in Dempster [21]. Theory showing the monotone behavior of the likelihood and convergence of the algorithm is derived, proving that the likelihood function $L(\theta, Y_{sample})$ is increasing in each iteration. Many examples are sketched, including missing value situations, applications to grouped, censored or truncated data, finite mixture models, variance component estimation, hyperparameter estimation, iteratively reweighted least squares and factor analysis.

After this initial work on the EM algorithm, more work on the convergence of the algorithm was done by Wu [65]: He stated that the incomplete data can be represented with a graph of the exponential family that posses a compact parametric space, where all of the limiting terms in an EM succession are stationary points for the likelihood function. Another outcome of Wu's paper is the proof that, if the likelihood function is unimodal, and satisfies certain differential condition. Then, any EM succession converges to the only maximum likelihood estimator.

The Log Likelihood function of the gaussian mixture models can be unbounded and therefore have several local maximums and saddle points. For this reason the EM algorithm is dependant on the initial values, and no single solution is guaranteed.

The convergence rate of the EM algorithm is proportional to the ratio between the observed data an the missing data. If the populations that compose the mixture are well separated, then the EM successions exhibit a very fast linear convergence. On the other hand, if these populations are poorly separated, the expected converge is slowly linear.

An efficient EM algorithm for estimation of mixtures densities is presented in Red-

ner and Walker [52]. In this work, results show that the convergence of the algorithm improves for large samples and better component separation. Bilmes [5] also describes the maximum-likelihood parameter estimation problem and how the Expectation-Maximization (EM) algorithm can be used for its solution. It provides an implementation of the EM parameter estimation procedure for two applications: 1) finding the parameters of a mixture of Gaussian densities, and 2) finding the parameters of a hidden Markov model (HMM).

In general, the EM algorithm is an elaborate technique that provides a general method for finding the maximum-likelihood estimate of the parameters of an underlying distribution from a given data set, it is a stable algorithm that can be used when the data is incomplete or has missing values.

5.2.1 Definition of the EM Algorithm

As before, we assume that X is observed and is generated by some distribution, we call X the *incomplete data*. We assume that a complete data set exists $Z = (X, Y)$, for which we specify a joint density function:

$$p(Z|\Theta) = p(x, y|\Theta) = p(y/x, \theta)p(x|\Theta) \quad (5.7)$$

With this new density function, we can define a new likelihood function called the complete-data likelihood:

$$L(\Theta|z) = L(\Theta|X, Y) = p(X, Y|\Theta) \quad (5.8)$$

This likelihood function will be used to estimate Θ .

The EM algorithm first finds the expected value of the complete-data log-likelihood $\log p(X; Y|\Theta)$ with respect to the unknown data Y given the observed data X and the current parameter estimates. Thus, we define

$$Q(\Theta, \Theta^{(i-1)}) = E [\log p(X, Y|\Theta)|X, \Theta^{(i-1)}] \quad (5.9)$$

where

$$E [\log p(X, Y|\Theta)|X, \theta^{(i-1)}] = \int_{y \in \Upsilon} \log p(X, y|\Theta) f(y|X, \Theta^{(i-1)}) dy \quad (5.10)$$

In the best of cases, this marginal distribution is a simple analytical expression of the assumed parameters $\Theta^{(i-1)}$ and the data. In the worst of cases, this density might be very hard to obtain.

The evaluation of this expectation is called the E-step of the algorithm. The second step (the M-step) of the EM algorithm is to maximize the expectation computed in the first step. That is, to find:

$$\Theta^{(i)} = \underset{\Theta}{\operatorname{argmax}} Q(\Theta, \Theta^{(i-1)}) \quad (5.11)$$

These two steps are repeated as necessary. Each iteration is guaranteed to increase the log likelihood and the algorithm is guaranteed to converge to a local maximum of the likelihood function.

Due to the small computational cost that the EM algorithm has per iteration, we believe that it will be possible to extend this estimation to a relational setting; In the next section we present an extension to this algorithm in order to perform the task of supervised classification in relational domains.

5.2.2 Number of Components in the Mixture

The problem of determining the number of components is a very complex one, to which no definite solution has been found. Several authors have proposed different solutions in order to determine the number of components k in the mixture model. The main factors affecting the outcome of the proposed solutions are the separation between the different populations within the data, the sample size and the dimensionality of the data [62].

One of the most widely accepted alternatives to estimate this number, and the one used in this work, is the *Bayesian Information Criterion* (BIC) [56]. This criterion is defined as:

$$BIC = 2Q(\Theta, \Theta^{(i-1)}) - m_M \log(n) \quad (5.12)$$

where $Q(\Theta, \Theta^{(i-1)})$ is the likelihood obtained via the EM algorithm for the model and m_M is the number of estimated parameters and n is the number of observations.

Given any two estimated models, the model with the larger value of BIC is preferred. The BIC penalizes free parameters more strongly than does the Akaike information criterion [1]; The work by Roeder and Wasserman [53], shows that this criterion is consistent when using gaussian mixtures in order to estimate probability density functions. Several experiments have shown the efficacy of the criterion; An example of which is the work by Dasgupta and Raftery [15], where they employ this criterion in order to determine the

number of components for the mixture models estimated via the EM algorithm.

5.3 Multi-Relational Classifier Based on Gaussian Mixture Models

Gaussian mixture models have been previously used in classification tasks on propositional domains [29]. Experimentation includes the work by Daza [17], who provides an extensive reference in the application of Gaussian mixtures to the task of supervised classification in KDD experiments. His research focuses on the combination of classifiers based on Gaussian mixtures by means of Bagging and Boosting.

In order to extend this approach to a multi-relational setting, we build our technique upon the independence assumption presented in the previous chapter; Namely, in order to estimate the probability distribution of the relational variables, we assume independence between all the predictor variables corresponding to each foreign key relation. In a supervised learning setting, one wants to estimate the parameters of the probability model. A multi-relational classifier based on Gaussian Mixture models estimates the class posterior probability function as a mixture of Gaussian distributions. Thus, the final estimate is computed as the product of the probability densities estimated for each foreign key relation by means of a Gaussian Mixture model. Figure 5.1 illustrates the proposed algorithm.

This algorithm's complexity is greatly influenced by the complexity of the algorithm used to estimate the probability density. In order to estimate the complexity of our algorithm we determine a bound for the EM algorithm as $O(EM)$, which will be analyzed in Chapter 6. For the Multi-Relational Classification Algorithm Based on Gaussian Mixtures, the complexity is bounded by $O(n_{classes} \cdot n_k \cdot (O(EM) \cdot n_{obs} \cdot n_R))$, where n_k is the maximum number of components that will be estimated in the Gaussian mixture model, n_{obs} is the number of examples, and n_R is the number of relations in the *DB*.

1. For each class label $y = y_1, y_2, \dots, y_n$ compute the probability density function:
 - 1.1 For each Foreign Key Relation $F \in R.F$
 - 1.2 For $k = 1, \dots, n_k$:

Apply the EM algorithm to the Foreign key relation, and store the likelihood in $l_F(\theta_j|x^n)$ where $x^n \in A(R.F)$

Estimate the BIC criterion for this value of k
 - 1.3 Select the optimal number of components:

$$k_{opt} = \operatorname{argmax}_k BIC$$
 - 1.4 Obtain the Multi-Relational probability density:

$$P(y_i|T.t) = \pi_j \prod_{i=1}^{n_R} l_F$$

where n_R is the number of relations in DB
2. Assign the class label that maximizes $P(y_i|T.t)$

Figure 5.1: Multi-Relational Gaussian Mixture Classification Algorithm

Experimental Results

6.1 Introduction

In order to test the efficiency of the classifiers proposed in previous chapters, we performed experiments focused on three relational data sets: The mutagenesis database which has been, widely used in Inductive Logic Programming (ILP) research [59]; the data for prediction of protein/gene localization and function from KDD Cup 2001 [10]; and the data released for the PKDD 2001 Discovery Challenge [13], from the medical domain for the prediction of thrombosis. The table 6.1 shows a summary for the characteristics of the data sets used; In the following sections we present the results obtained in experiments upon these data sets.

Relational Data Sets				
Data Set	No. Instances in Target R.	Number of Classes	No. Features in Target R.	No. Features in Foreign Key R.
Mutagenesis	188	2	5	5
Gene/Protein-Localization	862	15	6	2
Gene/Protein-Function	862	13	6	2
Thrombosis	1240	2	2	51

Table 6.1: Summary of Relational Data Sets

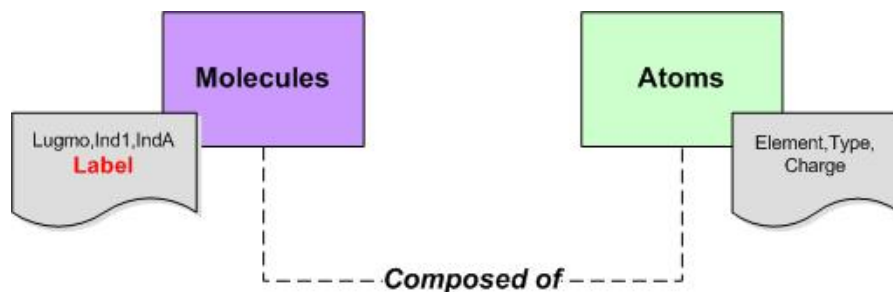


Figure 6.1: Relational Schema for Mutagenesis Data Set

6.2 Relational Data Sets

6.2.1 Mutagenesis Database

This database comes from the field of organic chemistry, the problem here is to predict the mutagenicity of a set of 230 aromatic and heteroaromatic nitro compounds. Mutagenicity is measured using the Ames test, its prediction is important as it is relevant to the understanding and prediction of carcinogenesis. These compounds occur in automobile exhaust fumes and sometimes are intermediates in the synthesis of thousands of industrial compounds.

This database, available from the Machine Learning Network **MLNet**, is one of the most widely used databases in ILP research. The most relevant results to the Machine Learning community of experimentation with this data are available in ([59], [60]). The original format of this database is **Prolog** syntax. In order to use this data set with our algorithms, we needed to translate it to the relational format. The database schema proposed for the domain is shown in figure 6.1.

The data set consists of 230 molecules and is drawn from the results obtained by Debnath et. al. [19], where two subsets of data are recognized: 188 compounds that could be fitted using linear regression, and 42 compounds that could not. The regression

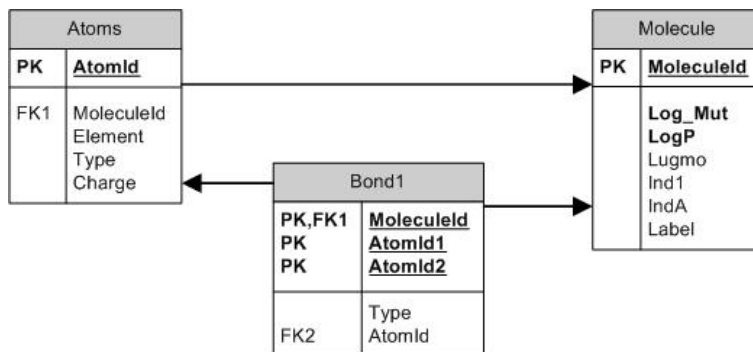


Figure 6.2: Physical Implementation of for Mutagenesis Database

friendly subset consists of 4893 atoms and 5243 bonds. In our experiments we selected the regression friendly subset, which is the one that has been treated in the Machine Learning literature. The implementation of the relational schema is shown in Figure 6.2.

6.2.2 Prediction of Protein/Gene Localization and Function

This database was released for the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001), in the framework of the KDD Cup competition. Figure 6.3 presents the relational schema proposed. The data is drawn from the field of bio-informatics, where there exists increasing interest in learning about the genes encoded in the sequences obtained in recent advances in genetics. Genes code for proteins, and these proteins tend to localize in various parts of cells and interact with one another, in order to perform crucial functions. This data set consists of a series of details about the various genes of one particular type of organism.

Two classification tasks are proposed in the challenge, to predict the functions and localizations of the proteins encoded by the genes. A gene/protein can have more than one function, but the algorithms we've proposed assume that each individual should be assigned to only one class label. In order to deal with multi-valued class attributes, we use the technique proposed by Atramentov [2]. Here, we predicted the membership of

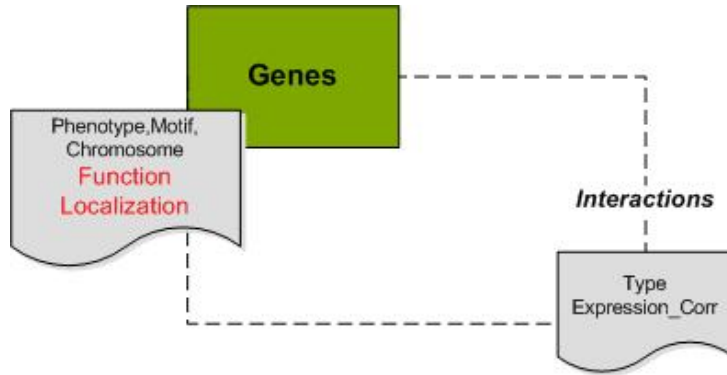


Figure 6.3: Relational Schema for the KDD Cup 2001 Data Set

each individual to each possible class, which means that we determine if a given protein has a function from all possible functions.

The overall accuracy of this approach is evaluated with this formula:

$$Accuracy = \frac{(truepositive + truenegative)}{(truepositive + truenegative + falsepositive + falsenegative)} \quad (6.1)$$

The original data is available in two variants. The first version consists of a single table, and is designed for propositional learning. The second format consists of two tables with 8 attributes in total, the target relation contains 862 genes. For our experiments we selected the second variant, because of its relational nature. However, the format in which the data is presented does not abide to the normal form [11]. In order to implement the relational schema in a physical database, we required that the attribute **GeneId** should be identified uniquely as a primary key for the target relation. Hence, we transformed the model into a normalized one presented in figure 6.4.

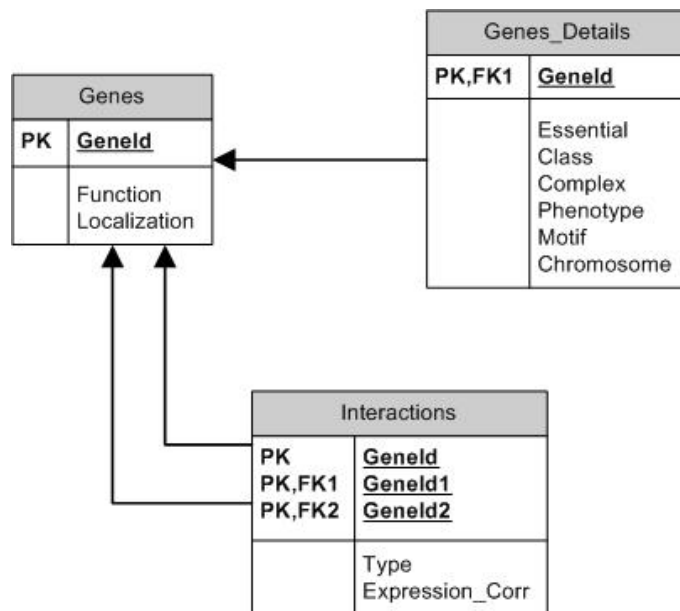


Figure 6.4: KDD Cup 2001 Database

6.2.3 Medical Domain - Prediction of Thrombosis

The database was collected at Chiba University hospital. The data portrays the results of a study made on patients suffering from collagen diseases. Collagen diseases are autoimmune diseases in which patients generate antibodies attacking their own bodies. The mechanisms of such diseases are only partially known, that is why there exist interest in obtaining more information on its classification. Medical studies have shown that thrombosis is one of the most important and severe complications of collagen diseases, and one of the major causes of death. Thrombosis is an increased coagulation of blood, that clogs blood vessels. The physicians that donated the data have found that this complication is closely related to anti-cardiolipin antibodies.

The classification task posed by this data consists in predicting the possibilities of the occurrence of thrombosis in collagen disease patients [4]. The data is provided in relational format, the schema is shown in Figure 6.5. The target relation contains records of 1240 patients admitted to the outpatient clinic for collagen diseases, the implemented

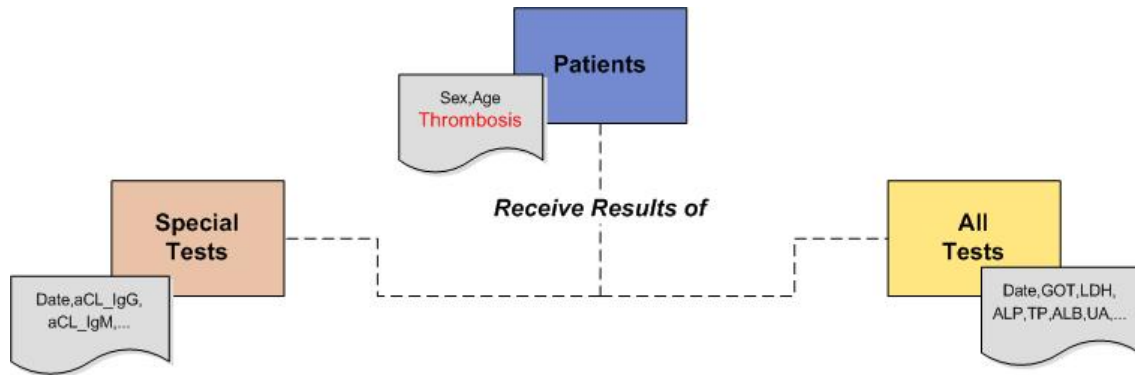


Figure 6.5: Relational Schema of Thrombosis Data

database is shown in figure 6.6.

6.3 Implementation Details

In order to carry out the experiments, we developed a computational environment in the Java programming language, making it platform-independent. Within this setting, the code required to carry out the knowledge discovery tasks was developed. Our implementation provides for the execution of the following data mining tasks for relational domains:

- Data preprocessing: Handling of missing values by means of Naive Bayes predictor 6.1.
- Classifier independent miss-classification error estimation, using the cross-validation technique.
- Supervised classification using Kernel Density Estimation.
- Supervised classification using Gaussian Mixture Models.

The relational data was implemented using the MS Access RDBMS. In order to access the data from our programming environment we used the JDBC Database Access

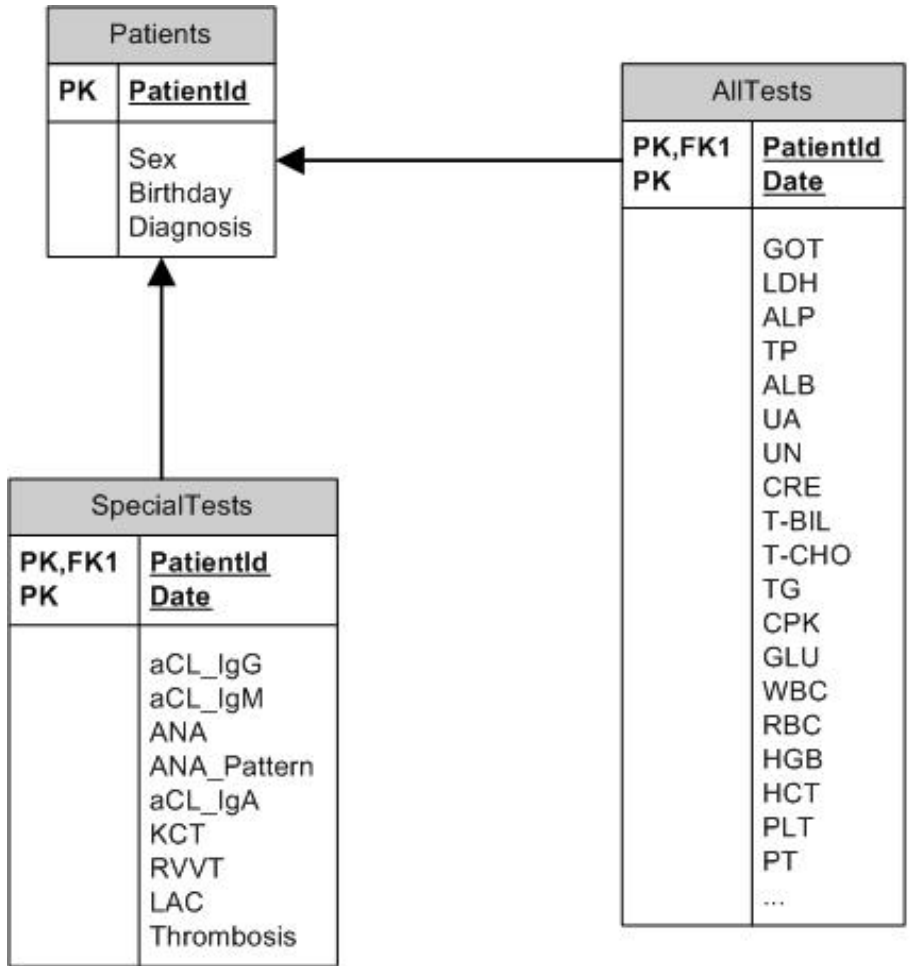


Figure 6.6: Thrombosis Database

API, this interface allows our algorithms to be independent of the choice of RDBMS. In order to operate the data, we abided by the ANSI SQL 1999 standard *Data Manipulation Language*; although internally some XML manipulation is performed, the interface to the database engine requires compliance with the SQL3 standard only.

6.4 Multi-Relational Kernel Density Estimation

In order to test this technique, we performed experiments on the described data sets using the general procedure for supervised classification; namely, construct a classifier using a

Multi-Relational Classifier Based on Kernel Density Estimation			
Data-Set/	Accuracy		
Kernel Function	Uniform Kernel	Gaussian Kernel	Bi-Weight Kernel
Mutagenesis	80.47%	81.71%	78.35%
Gene/Protein - Localization	70.89%	70.23%	69.45%
Gene/Protein - Function	79.86%	81.33%	77.67%
Thrombosis	90.05%	91.28%	90.91%

Table 6.2: Experimental Results for Multi-Relational Kernel Classifier

sample of the available data as Training Data and test the resulting classifier on the test set provided. The miss-classification error for the resulting model was computed using the cross-validation technique described in section 4.1.4, with $m = 10$ sub-samples.

The data sets used presents significant challenges because the data contains many *missing* attribute values; If not treated, this problem affects the predictor power of the classifier greatly. For our experimentation, we integrate a simple approach to treating this problem. A Naive Bayes model for each attribute in a table is built, based on the remaining attributes and excluding the class attribute [48]. Then, the missing attribute values are completed with the most likely value predicted by the Naive Bayes predictor. Then, for each tuple r from table R , the missing attribute values $X.A$ in the tuple are replaced with

$$v_{NB} = \operatorname{argmax} v_m \in \operatorname{DOM}(X.A) \prod_{\forall X_i \in DB} \prod_{\forall X_i.A_j, A_j \neq A_m} \prod_{\forall r_n \in X_i} P(X_i.A_j/v_m) \quad (6.2)$$

This estimate takes the first product over all relations in the database; The second product is taken over all the attributes in that relation, excluding the target attribute and

0. Initialization of m , s and p for all kernels:

$\mu_i =$ a vector in learning examples $\mu_i \neq \mu_j \forall j$

$$p_i = \frac{1}{\text{number of kernels}}$$

$\sigma_i =$ standar deviation of the i^{th} feature

$$\Rightarrow \theta_i = (\mu_i, \sigma_i)$$

1. Repeat until likelihood convergence

M-Step: maximization

$$\mu_j = \frac{\sum_n p(\theta_j|x^i)x^i}{\sum_n p(\theta_j|x^i)}$$

$$\sigma_j^2 = \frac{1}{d} \frac{\sum_n p(\theta_j|x^i)\|x^i - \mu_j\|^2}{\sum_n p(\theta_j|x^i)}$$

$$p_j = \frac{1}{N} \sum_n p(\theta_j|x^i)$$

where:

d is the dimension of x ;

N is the number of learning examples

E-Step: expectation

$$\begin{aligned} p(\theta_j|x^n) &= \frac{p(x^n|\theta_j^{old})p(\theta_j^{old})}{\sum_i p(x^n|\theta_i^{old})p(\theta_i^{old})} \\ &= \frac{N(\mu_j^{old}, \sigma_j^{old}, x^n)p_j^{old}}{\sum_i N(\mu_i^{old}, \sigma_i^{old}, x^n)p_i^{old}} \end{aligned}$$

where:

$N(\mu, \sigma, x)$ is a normal multivariate distribution

Figure 6.7: EM Algorithm

the class label; Then, the third product is taken over all the tuples in relations connected trough foreign key associations. The classifier is built as described after the missing values are filled in this manner. The results obtained with this approach are shown in Table 6.4

6.5 Multi-Relational Gaussian Mixture Models

In order to implement the technique described in chapter 5, we implemented a library of functions in JAVA. Our version of the EM algorithm assumes that the co-variance matrix

Multi-Relational Classifier Based on Gaussian Mixture Models	
Data-Set	Accuracy
Mutagenesis	83.94%
Gene/Protein - Localization	76.72%
Gene/Protein - Function	89.45%
Thrombosis	96.68%

Table 6.3: Experimental Results for Multi-Relational Gaussian Mixtures Classifier

for the predictor features has a diagonal structure, with varying volume; This means that the predictor features in each foreign key relation are independent, and we estimate the co-variance for each pair of features. This results in a simplification of the EM algorithm, the implemented algorithm is shown in Figure 6.7. The probability density function was estimated using mixture models within the range of $k = 1, \dots, 6$ components, the best number of components in the generated model model was selected using the BIC criterion.

Since our implementation of the EM algorithm simplifies the computations for the covariance matrix, and reduces the number of parameters that will be estimated, the complexity of the algorithm is also significantly reduced. The expectation step has a complexity of $O(n_{obs})$, and for the maximization step, the bound is determined by $O(n_{obs} \cdot n_M)$, where n_{obs} represents the number of examples and n_M is the number of parameters that needs to be estimated. In past studies of the numerical properties for this algorithm, it has been shown that its convergence is slowly linear at the worst case scenario [65]; Therefore, our implementation presents a bound of $O(n_{obs}^2 \cdot n_M)$ at the worst case.

The results obtained with this classifier are shown in Table 6.3. This results show a significant improvement from the results obtained using kernel density estimation, however this increase in accuracy comes at the cost of a significant increase in running time.

Running Time (in milliseconds)				
Data-Set	MR-KDE	MR-KDE	MR-KDE	MR Gaussian
	Uniform	Gaussian	Bi-Weight	Mixture
Mutagenesis	7,832	7,727	8,103	40,513
Localization	8,985	9,122	9,240	68,269
Function	9,123	9,526	9,384	70,343
Thrombosis	12,430	12,853	12,678	57,972

Table 6.4: Running Time for the Implemented Methods

Comparative Evaluation					
Data-Set	MR-KDE	MR-KDE	MR-KDE	MR Gaussian	Best Reported
	Uniform	Gaussian	Bi-Weight	Mixture	Accuracy
Mutagenesis	80.47%	81.71%	78.35%	83.94%	86% [57]
Localization	70.89%	70.23%	69.45%	76.72%	72.1% [10]
Function	79.86%	81.33%	77.67%	89.45%	93.6% [10]
Thrombosis	90.05%	91.28%	90.91%	96.68%	99.28% [13]

Table 6.5: Performance Comparison with Best Known Results

Figure 6.4 shows a comparison of the running time for the implemented methods. These experiments were carried out under the MS Windows platform. Using a DELL® PrecisionTM 690 workstation, which had Intel® XeonTM CPUs with speeds of 3.00 GHz and 2.99 GHz, under a 64 Bits environment and with 16 GB of RAM memory. Both the accuracy and running accounted in this thesis results from performing 50 repetitions of each experiment and then the average of these values is reported.

Conclusions and Future Work

In multi-relational data mining, algorithms are faced with the task of learning from relational databases and from data describing complex/structured objects. One of the main tasks of data mining is the task of supervised classification; and several techniques have been proposed in the literature in order to solve this problem. Most of the approaches in multi-relational data mining are based on the ILP paradigm, and recent developments are working with probabilistic relational models. This thesis presents two new algorithms for estimating probability distributions over relational data sets. The models are based on traditional non-parametric techniques, and contrary to most of the previous relational approaches, our approach is based on notions from relational algebra. By incorporating this relational representation of the data, we provide a framework which has the potential for a wider use than other representations that build on first-order logic bases or graph based representation, because this approach is more generalized across the database community.

The first algorithm focused on multi-relational Kernel density estimation, this technique for estimating the probability distribution over relational data was described in [55]. Another related work is done by Woznika et. al. [64], here they present a general framework for kernel-based learning over relational schemata in a similar manner to the one used in this thesis. However, their model is based upon a newly defined attribute type call instance-set and their algorithm used a relational distance measure based on kernels in order to perform the classification task. Therefore, the approach proposed here presents a novel solution to this problem.

Experimental evaluation of this method shows that it performs in an efficient manner. However, it still lacks the predictive performance that has been obtained in the best known published results of the three relational data sets used in the experiments. This result can be attributed to the characteristic of the databases, they possess a high dimensional feature space, and the number of observations available to the learner is relatively small for non-parametric techniques; This conclusion is confirmed by the outcome obtained with the Thrombosis data set, which contains the larger number of observations of the evaluated databases and achieved the best result for this technique. Therefore, this technique is better suited for data sets that possess a large number of observations in the training sample, in order to improve the predictive performance of the classifier.

Supplementary experiments with this algorithm can study the effect that the parameter settings for the elementary kernels used in this thesis have over the classifier. Another interesting route for future work can be the study and creation of new kernel functions that can build upon the relational properties from the set of predictor variables. This new kernel function should try to express the expanded relational feature space as reduced space, such as the mapping proposed by Cumby and Roth [14].

The task of learning probability distributions using Gaussian mixture models has been extensively studied within propositional domains, still, it has not been used in a relational setting. In this Thesis we present an algorithm to approximate the probability density function of a relational database using Gaussian Mixture models, via the EM algorithm. The model generated is used for the task of supervised classification, and an empirical evaluation shows that this method compares well to other published results that use the same relational data sets.

This technique improves the results obtained with our previous approach, this improvement can be attributed to the fact that the technique is slightly parametric in nature.

The models are generated making some assumptions about the behavior of the density function, and in our implementation we also assume independence between the predictor features.

Further work in this direction should include alternative structures for the covariance matrix estimated with the EM algorithm, our implementation explores one structure but more complex models can also be generated if the independence assumption is bypassed. Comparisons between methods for selecting the optimal number of components is also a possibility of further research, alternatives include the Akaike Criterion, and the Mutual Information Criterion. Another interesting area to explore consist the incorporation of query optimizing techniques. Recent work in this direction has proven to speed up multi-relational data mining algorithms [63] by means of identifying common subexpressions in sets of relational queries and data mining query scheduling.

In conclusion, this Thesis has explored non-parametric techniques and their predictive performance for the task of supervised classification in multi-relational domains. The methods proposed consist of original solutions to this data mining task, and the experimental evaluation of the algorithms show an efficient performance within benchmark relational data sets. These techniques constitute a first step towards the exploration of non-parametric techniques and their extension to a multi-relational setting. Moreover, this work deals with relational data stored directly in relational database systems, providing an expression language that is intuitive to SQL programmers and taking one step closer to bridging the gap between traditional KDD techniques and the database community.

Bibliography

- [1] H. Akaike. Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*. Akademiai Kiado, Budapest., 1973.
- [2] A. Atramentov. Multi-relational decision tree algorithm - implementation and experiments. Master's thesis, Computer Science, Iowa State University, 2003.
- [3] A. Atramentov, H. Leiva, and V. Honavar. A multirelational decision tree learning algorithm: Implementation and experiments. In *In Proceedings of the 13th International Conference on Inductive Logic Programming*. Springer-Verlag, 2003.
- [4] P. Berka. The pkdd discovery challenges on thrombosis data. In *European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2001.
- [5] J. Bilmes. A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical report, University of Berkeley, ICSI-TR-97-021, 1997.
- [6] H. Blockeel. *Top-down induction of first order logical decision trees*. PhD thesis, Department of Computer Science, Katholieke Universiteit Leuven, 1998.
- [7] H. Blockeel and L. De Raedt. Relational knowledge discovery in databases. In *Proceedings of the 6th International Workshop on Inductive Logic Programming*, volume 1314. Springer-Verlag, 1997.

- [8] U. Bohnebeck, T. Horvath, and S. Wrobel. Term comparisons in first-order similarity measures. In *Proceedings of the Eight International Conference on Inductive Logic Programming*, pages 65–79. Springer, 1998.
- [9] D.D. Chamberlain and R.F. Boyce. Sequel: A structured english query language. In *Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) workshop on Data description, Access and Control*, pages 249–264. International Conference on Management of Data, Ann Arbor, Michigan, 1974.
- [10] J. Cheng, C. Hatzis, H. Hayashi, M. Krogel, S. Morishita, D. Page, and J. Sese. KDD cup 2001 report. *SIGKDD Explorations*, 3(2):47–64, 2002.
- [11] E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.
- [12] D. Cook and L. Holder. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research*, 1:231–255, 1994.
- [13] I. Coursac, N. Duteil, and N. Lucas. pkdd 2001 discovery challenge - medical domain. In *PKDD Discovery Challenge 2001*, volume 3, 2002.
- [14] C. Cumby and D. Roth. On kernel methods for relational learning. In *The Twentieth International Conference on Machine Learning (ICML). Washington, DC USA*. AAAI Press, 2003.
- [15] A. Dasgupta and A.E. Raftery. Detecting features in special point processes with clutter via model-based clustering. *Journal of the American Statistical Association*, 1998.
- [16] C. Date. *Introducción a los Sistemas de bases de datos*. 1993.
- [17] L. Daza. Combinación de clasificadores basados en mezclas gaussianas. Master’s thesis, Mathematics Department, University of Puerto Rico at Mayaguez, July 2002.

- [18] L. De Raedt, H. Blockeel, L. Dehaspe, and W. van Laer. Three companions for data mining in first order logic. *Relational Data Mining*, pages 105–139, 2001.
- [19] A.K. Debnath, R.L. Lopez de Compadre, G. Debnath, A.J. Shusterman, and C. Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *J. Med. Chem.*, 34:786–797, 1991.
- [20] L. Dehaspe and H. Toivonen. Discovery of frequent DATALOG patterns. *Data Mining and Knowledge Discovery*, 3(1):7–36, 1999.
- [21] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal Royal Statistical Society series B*, 39:1–38, 1977.
- [22] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, New York, 2000.
- [23] S. Dzeroski and N. Lavrac. An introduction to inductive logic programming. *Relational Data Mining*, pages 48–73, 2001.
- [24] S. Dzeroski and N. Lavrac. *Relational Data Mining*. Springer-Verlag, 2001.
- [25] U. Fayyad. Knowledge discovery in databases: An overview. *Relational Data Mining, Springer-Verlag*, pages 28–47, 2001.
- [26] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uhturudsamy. Advances in knowledge discovery and data mining. *AAAI Press/MIT Press*, 1996.
- [27] L. Getoor. Multi-relational data mining using probabilistic relational models: research summary. In *Proceedings of the First Workshop in Multi-relational Data Mining*, 2001.
- [28] L. Getoor. *Learning Statistical Models from Relational Data*. PhD thesis, Computer Science, Stanford University, June 2002.

- [29] T. Hastie and R. Tibshirani. Discriminant analysis by gaussian mixtures. *Journal Royal Statistical Society series B*, 58:155–176, 1994.
- [30] R. Herbrich. *Learning Kernel Classifiers*. The MIT Press, 2002.
- [31] S. Karmer and G. Widmer. Inducing classification and regression trees in first order logic. *Relational Data Mining*, pages 140–159, 2001.
- [32] N. Ketkar, L. Holder, and D. Cook. Qualitative comparison of graphbased and logicbased multirelational data mining: A case study. In *4th international Workshop on Multi-Relational Mining, Chicago, Illinois*, 2005.
- [33] M. Kirsten and S. Wrobel. Relational distance-based clustering. In *Proc. Fachguppentreffen Maschinelles Lernen (FGML-98)*, pages 119–124. Techn. Univ. Berlin, Technischer Bericht 98/11, 1998.
- [34] M. Kirsten and S. Wrobel. Extending k-means clustering to first order representations. In *Proceedings of the Tenth International Conference on Inductive Logic Programming*, pages 112–129. Springer, 2000.
- [35] J. Knobbe, H. Blockeel, A. Siebes, and D. Van der Wallen. Multi-relational data mining. In *Proceedings of Benelearn*, 1999.
- [36] J. Knobbe, A. Siebes, H. Blockeel, and D. van der Wallen. Multi-relational data mining using uml for ilp. In *Proceedings of the First Workshop in Multi-relational Data Mining*, 2001.
- [37] J. Knobbe, A. Siebes, and Van der Wallen. Multi-relational decision tree induction. In *Proceedings of the 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases*, 1999.
- [38] S. Kramer. Demand-driven construction of structural features in ILP. *Lecture Notes in Computer Science*, 2157:132+, 2001.

- [39] S. Kramer, N. Lavrac, and P. Flach. Propositionalization approaches to relational data mining. *Relational Data Mining*, pages 262–291, 2001.
- [40] S. Kramer, B. Pfahringer, and C. Helma. Stochastic propositionalization of non-determinate background knowledge. In *International Workshop on Inductive Logic Programming*, pages 80–94, 1998.
- [41] H.P. Kriegel and S. Schonauer. Similarity search in structured data. In *Proceedings of the 5th Conference on Data Warehousing and Knowledge Discovery, Prague, Czech Republic*, 2003.
- [42] M. Krogel, S. Rawles, F. Zelezny, P. Flach, N. Lavrac, and S. Wrobel. Comparative evaluation of approaches to propositionalization. In *Proceedings of the 13th Int. Conference on Inductive Logic Programming*. Springer-Verlag, 2003.
- [43] N. Lavrac, S. Dzeroski, and M. Grobelnik. Experiments in learning nonrecursive definitions of relations in linus. Technical report, Jozef Stefan Institute, 1991.
- [44] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.
- [45] S.H. Muggleton. Inverse entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3–4):245–286, 1995.
- [46] S.H. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19(20):629–679, 1994.
- [47] J. Neville. *Statistical Models and Analysis Techniques for Learning in Relational Data*. PhD thesis, Computer Science, University of Massachusetts Amherst, September 2006.
- [48] J. Neville, D. Jensen, and B. Gallagher. Simple estimators for relational bayesian classifiers. Technical report, University of Massachusetts, 03-04, 2004.
- [49] Z. Pawlak. Information systems theoretical foundations. *Inf. Syst.*, 6(3):205–218, 1981.

- [50] J. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann, San Francisco, CA, 1993.
- [51] R. Quinlan. Foil: A midterm report. In *Proceedings of the 6th European Conference on Machine Learning*, volume 667. Springer-Verlag, 1993.
- [52] R.A. Redner and H. Walker. Mixture densities, maximum likelihood and the em algorithm. *SIAM Review*, 26:195–239, 1984.
- [53] K. Roeder and L. Wasserman. Practical density estimation using mixtures of normals. *Journal of the American Statistical Association*, 92:894–902, 1997.
- [54] H. Rybiski. On first-order-logic databases. *ACM Trans. Database Syst.*, 12(3):325–349, 1987.
- [55] S. Sanghai, P. Domingos, and D. Weld. Relational dynamic bayesian networks. *Journal of Artificial Intelligence Research*, 24:759–797, 2005.
- [56] G. Schwartz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.
- [57] 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. Technical report, PRGTR-08-99, Oxford University Computing Laboratory, Oxford, 1999.
- [58] A. Srinivasan and R.D. King. Feature construction with inductive logic programming: A study of quantitative predictions of biological activity aided by structural attributes. In S. Muggleton, editor, *Proceedings of the 6th International Workshop on Inductive Logic Programming*, pages 352–367. Stockholm University, Royal Institute of Technology, 1996.
- [59] A. Srinivasan, S.H. Muggleton, and R.D. King. Comparing the use of background knowledge by inductive logic programming systems. In *Proceedings of the Fifth International Workshop on Inductive Logic Programming*, 1995.

- [60] A. Srinivasan, S.H. Muggleton, M.J.E. Sternberg, and R.D. King. Theories for mutagenicity: a study of first-order and feature based induction. Technical report, PRG-TR-8-95, Oxford University Computing Laboratory, 1995.
- [61] P. Turney. Low size-complexity inductive logic programming: The East-West challenge considered as a problem in cost-sensitive classification. In L. De Raedt, editor, *Proceedings of the 5th International Workshop on Inductive Logic Programming*, pages 247–263. Department of Computer Science, Katholieke Universiteit Leuven, 1995.
- [62] A. Webb. *Statistical Pattern Recognition*. Oxford University Press Inc., 1999.
- [63] M. Wojciechowski and M. Zakrzewicz. *Advances in Knowledge Discovery and Data Mining*, volume 3518/2005, chapter On Multiple Query Optimization in Data Mining, pages 696–701. Springer Berlin / Heidelberg, 2005.
- [64] A. Woznica, A. Kalousis, and M. Hilario. Kernel-based distances for relational learning. In *3rd Workshop on Multi-Relational Data Mining. Seattle, Washington*, 2004.
- [65] C.F. Wu. On the convergence properties of the em algorithm. *The Annals of Statistics*, 11:95–103, 1983.