Department of Computing

Probabilistic Models for Mining Imbalanced Relational Data

Amal Saleh Ghanem

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University of Technology

November 2009

Amal Ghanem
November 2009

Abstract

Most data mining and pattern recognition techniques are designed for learning from flat data files with the assumption of equal populations per class. However, most real-world data are stored as rich relational databases that generally have imbalanced class distribution. For such domains, a rich relational technique is required to accurately model the different objects and relationships in the domain, which can not be easily represented as a set of simple attributes, and at the same time handle the imbalanced class problem.

Motivated by the significance of mining imbalanced relational databases that represent the majority of real-world data, learning techniques for mining imbalanced relational domains are investigated. In this thesis, the employment of probabilistic models in mining relational databases is explored. In particular, the Probabilistic Relational Models (PRMs) that were proposed as an extension of the attribute-based Bayesian Networks. The effectiveness of PRMs in mining real-world databases was explored by learning PRMs from a real-world university relational database. A visual data mining tool is also proposed to aid the interpretation of the outcomes of the PRM learned models.

Despite the effectiveness of PRMs in relational learning, the performance of PRMs as predictive models is significantly hindered by the imbalanced class problem. This is due to the fact that PRMs share the assumption common to other learning techniques of relatively balanced class distributions in the training data. Therefore, this thesis proposes a number of models utilizing the effectiveness of PRMs in relational learning and extending it for mining imbalanced relational domains.

The first model introduced in this thesis examines the problem of mining imbalanced relational domains for a single two-class attribute. The model is proposed by enriching the PRM learning with the ensemble learning technique. The premise behind this model is that an ensemble of models would attain better performance than a single model, as misclassification committed by one of the models can be often correctly classified by others.

Based on this approach, another model is introduced to address the problem of mining multiple imbalanced attributes, in which it is important to predict several attributes rather than a single one. In this model, the ensemble bagging sampling approach is exploited to attain a single model for mining several attributes. Finally, the thesis outlines the problem of imbalanced multi-class classification and introduces a generalized framework to handle this problem for both relational and non-relational domains.

Acknowledgments

First and foremost, I would like to thank God for giving me the patience and guidance to complete my PhD study.

Then, I would like to thank the following people for helping me to complete this thesis.

First, I would like to thank my supervisors, Professor Svetha Venkatesh and Professor Geoff West. I thank them for guiding me to this interesting topic of my research and for their continuous support and motivation in all aspects of my research.

Also, I wish to express my gratitude to the University of Bahrain for its support and sponsorship during my post-graduate studies which has greatly contributed to the successful completion of this study.

Additional thanks go to my colleagues in IMPCA group for their discussions and opinions during my study, and especially my office mates, Annika and Thorsten. I would also like to thank my friends: Loujane, Aisha, Moza, Nee'ama, Alaya, Kawther, and Amal for their encouragement and support especially at tough times.

My loving thanks to my parents and my husband, to whom I dedicate my work.

I am thankful to my wonderful parents for their love and best wishes during my study abroad.

I owe my deepest gratitude to my mother for being the heart of my life. Thank you for your patience, love, prayers and care in every step of my life.

I am heartily thankful to my father who taught me the values of hardworking, patience and self-confidence. Thank you for your endless support, advices and encouragement to proceed from one step to the next.

Without my parents' encouragement and believe in me, it would have been impossible for me to finish this work.

Special thanks to my loving husband for his kindness, patience and endless support during my study. With his love and support, I enjoyed each moment in my study abroad and even loved my study more.

Finally, I wish to thank my lovely sisters: Ameena, Eman and Sarah and my young brother: Khalid, for their loving support. I thank them for all of their help during my study, especially when I was preparing to get married.

Contents

List of Figures

List of Tables

Relevant Publications

This thesis is based on some related work that has been published or documented elsewhere. The list of these publications is provided below.

- Ghanem, A. S., Venkatesh, S., and West, G. A. W. (2008). Learning in imbalanced relational data. In *Proc. International Conference on Pattern Recognition (ICPR)*. IEEE Computer Society.
- Ghanem, A. S., Venkatesh, S., and West, G. (2009). Classifying multiple imbalanced attributes in relational data. In *Proc. 22nd Australasian Joint Conference on Advances in Artificial Intelligence*, pages 220–229, Berlin, Heidelberg. Springer-Verlag.
- Ghanem, A. S., Venkatesh, S., and West, G. A. W. (2010). Multi-class pattern classification in imbalanced data. In *Proc. International Conference on Pattern Recognition (ICPR)*. IEEE Computer Society. to appear.

Chapter 1

Introduction

For years, relational database systems have been used for storing and representing the data records of many real-world domains. This is due to their ability to represent data in a compact and natural structure, and at the same time characterize the different relationships in the domain. Nowadays, the utilization of relational database systems ranges from major financial, medical and educational domains to small retailers.

The established and increasing use of these relational databases has highlighted the necessity for relational mining tools to analyse these rich domains and search for any interesting or important patterns within the data. Such tools are important in different domains, as they can significantly help in identifying the key factors for achieving further goals on the domain, and they can be also employed effectively for forecasting potential trends on the domain.

One of the methods that can be employed for mining relational domains is building a probabilistic model of the relational domain. This mainly concerns learning the different interactions between the relational variables that are linked in varied and complex forms. Such learning requires a rich probabilistic model that can handle the learning from the different configurations of the relational domain.

Probabilistic Graphical Models (Jordan, 1999, 2004) and especially Bayesian Networks (Pearl, 1988) have addressed this type of probabilistic modelling, and have been employed successfully in different domains. However, these models are designed to learn from single flat files that consist of a set of fixed attributes and a single static relationship that links the attributes. Therefore, for mining a relational domain using these methods, the data must be first converted into a flat file. This conversion has the major drawback of losing the different relationships between the domain variables, which play a significant role in mining and exploring these domains.

For the task of mining relational data, the Probabilistic Relational Models (PRMs) (Koller and Pfeffer, 1998; Getoor, 2002) were introduced as an extension of attribute-based Bayesian

Networks. PRMs handle the limitations of Bayesian Networks in learning from relational domains by utilizing the concepts of objects and object properties. Thus, PRMs learn from the entire relational database and describes the dependencies held in the domain, between attributes of the same table or related tables via foreign keys. Once a PRM model is learned, it can be used for data analysis to explore the important patterns and dependencies in the learned relational domain.

However, one of the critical tasks in mining relational domains is classification, which is the ability to obtain an accurate predictive model for classifying new instances of the learned domain. Although PRMs handle the problem of relational learning, it is based on the assumption shared by the majority of standard learning techniques of assuming balanced training data. Research has shown that the performance of most techniques degrades significantly for the case of learning with imbalanced data (Japkowicz and Stephen, 2002), where the majority of the data belongs to one class and only a few samples belong to the other classes. Consequently, the imbalanced class problem could severely limit the ability of PRMs to effectively mine imbalanced relational domains.

The imbalanced class problem is a common problem in real-world domains and a major issue in mining relational data. Although the imbalanced class problem has received considerable attention and several attempts have been proposed to handle this problem, these attempts mainly learn from a flat data representation. On the other hand, a few attempts have been proposed to handle the imbalanced class problem in relational domains, but they are practical only in certain contexts and lack the ability to fully explore and model the entire relational database as in PRMs.

The motivation behind the research in this thesis is to investigate the PRM learning technique for exploring and analysing imbalanced relational domains and obtaining extended models that can be employed in many different real-world applications.

1.1 Aims and approach

This thesis examines the problem of mining imbalanced relational domains via extending PRMs. The objectives are to:

- Explore the PRM concepts in mining real-world imbalanced relational domains and investigate the effectiveness of the learnt PRM models as predictive models in such domains.

- Investigate the specific problems of mining imbalanced relational data with: (1) multiple imbalanced attributes, and (2) for the case of multi-class imbalanced classification.
- Develop extended models of PRMs to handle the stated problems of mining imbalanced relational domains that can be applied to different domains.

The first part of this thesis tackles the exploration of the PRM concepts in mining real-world imbalanced relational domains. Therefore, the PRM was applied to a real-world university database for mining the records of undergraduate students. The objective of this application was to explore the PRM concepts and discover the main interactions in this domain. This goal was further supported by developing a visualization tool to illustrate the mining outcomes of the learned PRM model.

Although the learned PRM model from the university database confirmed the capability of PRMs in discovering the main dependencies in this domain, the learned model failed as a predictive model. The failure was a result of modelling a relational domain with imbalanced training data, in which one class of data is over-represented by a large number of samples compared to the other classes. The imbalanced class problem is common in many diverse domains and has been reported to have very poor performance with most well-known learning techniques.

The imbalanced class problem in relational data is even more complicated and a major issue in data mining. Therefore, this problem forms the basis of the research in this thesis and three models are introduced for mining imbalanced relational domains. The idea behind these models is the concept of utilizing the power of PRMs in modelling relational domains and extending it to handle the imbalanced class problem.

The first model, named PRMs-IM, extends the learning technique of PRMs to handle the imbalanced class problem in relational domains. The goal of this extension is to obtain a model from the relational training data that can be used effectively in classification, particularly for accurately classifying rare cases in the domain. This model is proposed for the problem of classifying one variable at a time, and specifically for the two-class case. The objective of this model is achieved by enriching the PRM with ensemble learning. This approach relies on the principle, reported in the literature, that an ensemble of a set of models often performs more effectively than individual models. Therefore, PRMs-IM handles the imbalanced class problem in relational domains by building an ensemble of relational models rather than a single one.

Building on PRMs-IM, a further model, PRMs-IM2, is introduced for classifying multiple imbalanced attributes in relational domains. PRMs-IM2 offers the opportunity to model several imbalanced attributes in one model rather than building an independent model for each attribute as proposed in PRMs-IM. To obtain a single model for all the imbalanced attributes, PRMs-IM2 employs the concept of the popular Bagging ensemble approach (Breiman, 1996). This allows PRMs-IM2 to model all the attributes simultaneously, and at the same time reduces the time required for training and inference.

Both PRMs-IM and its extension PRMs-IM2 are introduced to address the imbalanced class problem in relational domains for the special case of two-class classification. However, in several domains, there are many different pattern classes required to be classified rather than just two classes. Therefore, to handle the multi-class classification of imbalanced attributes, a further model is introduced, named Multi-IM. The approach followed in Multi-IM is based on an evaluation of the popular multi-class methods reported in the literature. Based on this evaluation, Multi-IM was proposed to utilize the strengths of the methods reviewed and avoid their shortcomings. Importantly, Multi-IM is proposed as a generalized framework for both relational and non-relational domains.

1.2 Contribution

The contributions of this thesis lies in developing a number of mining models applicable for learning from imbalanced relational domains. Furthermore, a visual data mining tool is proposed to aid the interpretation of the outcomes of the PRM learned models. This visualization tool provides a simpler means of illustrating the outcomes of the PRM models by mapping them onto a spatial representation.

The three models introduced in this thesis are:

- The PRMs-IM model, which is the core contribution in this thesis. PRMs-IM is proposed and demonstrated experimentally to extend PRMs modelling to handle the imbalanced class problem in relational domains by using an ensemble of PRM models.
- The PRMs-IM2 model, that deals with the problem of modelling multiple imbalanced attributes in relational domains. PRMs-IM represents the starting point of this

model, which is extended using the popular Bagging approach (Breiman, 1996) to model more than one imbalanced attribute in a single model.

- The Multi-IM model, as a generalized framework for both relational and flat domains for imbalanced multi-class classification. This model is also based on the concepts of PRMs-IM and extended for multi-class classification rather than the binary classification of PRMs-IM.

In addition, a demonstration of the developed models is presented on real-world databases to show their effectiveness in mining such imbalanced relational domains.

1.3 Thesis outline

The thesis is organized as follows:

A review of the related work is presented in Chapter 2. The review starts with an overview of the Probabilistic Relational Model (PRM) describing its semantics and learning algorithm. Following this, a detailed overview of research in the area of mining imbalanced relational domains is presented. This overview includes research on modelling imbalanced multiple attributes and imbalanced multi-class classification.

In Chapter 3, a real-world application of PRMs is introduced. The objective behind this application is to explore the PRM concepts in mining a real-world university relational database. In this chapter, the effectiveness of PRMs in capturing the most important dependencies in this relational domain is presented. Additionally, a visualization data mining tool is provided to illustrate the outcomes of the learnt PRM models.

Chapter 4 describes an extension of PRMs to handle the imbalanced class problem in relational domains. In this chapter, the effectiveness of PRMs in obtaining accurate classification models from imbalanced relational domains is investigated. Based on this investigation, an extension of PRMs named PRMs-IM is introduced for learning from relational imbalanced data by enriching PRMs with ensemble learning.

In Chapter 5, an extension of the work proposed in its preceding chapter is outlined for classifying multiple imbalanced attributes in relational data. The objective behind this extension, named PRMs-IM2 is to handle the problem of the previous approach of modelling a single imbalanced attribute at a time. PRMs-IM2 utilizes the concept of

PRMs-IM and the popular bagging ensemble approach to model more than one attribute in a single model.

Chapter 6 discusses the problem of learning from imbalanced multi-class data and proposes a new approach, named Multi-IM, to handle this problem. Multi-IM derives its fundamentals from PRMs-IM and extends it to a generalized framework for multi-class imbalanced learning for both relational and non-relational domains.

Finally, Chapter 7 provides a concluding summary of the work of this thesis, along with potential future directions.

Chapter 2

Background

This chapter discusses statistical methods for mining relational data in different domains. The chapter starts with the definitions of Graphical Models and Bayesian Networks and points to the effectiveness of these methods in modelling and mining real-world applications. The limitations of Bayesian Networks in modelling relational domains are then discussed, and an extension of Bayesian Networks for relational learning, namely Probabilistic Relational Models, is presented. The semantics and learning techniques of Probabilistic Relational Models are then explored. Subsequently, the chapter discusses the effectiveness of employing Probabilistic Relational Models for mining imbalanced relational data. This discussion starts with defining the nature of the imbalanced class problem and the evaluation measures; followed by a review of the proposed solutions for this problem. Finally, the chapter points to the necessity of improving Probabilistic Relational Models to handle the problem of imbalanced data distributions in relational domains.

2.1 Introduction

The majority of real-word data are stored in relational database systems as a set of relations related in various ways representing the domain structure. This relational data, in many cases acquired over many years, offers a rich ground for mining applications to gain a better understanding about the domain and how this data can be employed to achieve better performance. This mining task involves learning from a rich relational domain that cannot be easily represented as a set of simple attributes with a fixed relationship. For such domains, a rich relational technique is required to model the different objects and relationships in the domain.

An example of such modelling is offered by Probabilistic Graphical Models (Jordan, 1999, 2004), which provide a simple and yet very effective framework for modelling complex domains. The effectiveness of graphical models is achieved by combining probability and graph theory. The basic idea of graphical models is the concept of modularity, in which a large-scale model can be built out of a set of local models, and probability theory is used to combine these models into a single consistent system representing the original domain problem.

A graphical model of a given domain can be viewed as a graph topology describing the interactions between the domain variables, along with a joint probability distribution over these variables. This graph not only provides a natural and compact representation of the data, but also a solid grounding for learning and inference algorithms to be implemented. Inference can be carried out by using probability theory to compute the marginal or conditional probabilities of the variables of interest.

A popular approach of graphical models is the Bayesian Network (BN) (Pearl, 1988). A BN is a directed graphical model for describing the probabilistic relationships between a set of variables. BN provides a compact representation over the variables by utilizing the conditional independence between the variables, in which only few variables of the domain affect each other directly.

Despite the success of BNs reported in the literature in different applications, BNs cannot deal with complex domains that have varying numbers of objects and relationships, such as relational domains that could be instantiated in different configurations. BNs usually involve learning from a single flat file that consists of all the data as a set of pre-specified attributes with fixed relationships. Therefore, to apply BN learning to a rich relational database, the data must be converted first into a flat representation. This flattening of the relational data has several limitations, including: having redundant data, losing the structural representation of the data, losing the normalization benefits of the relational data and increasing the consistency-maintenance overhead. Moreover, Getoor (2002) discussed the problem of introducing statistical skew when flattening relational data, as records with multiple links would appear multiple times in the flat file, and hence result in incorrect statistics.

In order to learn directly from relational data, a number of approaches have been proposed, such as: FOIL (Quinlan and Cameron-Jones, 1993), TILDE (Blockeel and Raedt, 1998) and CrossMine (Yin *et al.*, 2004). These approaches are designed to construct classifiers based on rules or decision trees. Therefore, these models are proposed for the specific purpose of relational classification, i.e. learn the classification rules of a specific domain variable. As a result, these methods do not provide a tool for modelling the different interactions between the relational domain variables. Moreover, the resulting model of these methods can be only used to perform inferences about the modelled variable, and hence do not have the ability to infer about another variable (or set of variables) in the domain as in BNs.

The challenge in learning from relational data includes learning from the data in its relational format to discover the dependencies over the set of objects and relations, and then using the learnt model to do inferences about new objects of the relational domain given some observation. To address this task of relational learning and inference directly from relational domains, the Probabilistic Relational Models (PRMs) were first introduced by Koller and Pfeffer (1998) and then refined by Getoor (2002). PRMs have been introduced as an extension of the standard attribute-based BN to learn directly from the relational data without flattening the data as required in BN learning.

The relational learning is performed in PRMs by employing the concepts of objects, object properties, and interactions between objects. These concepts are utilized to specify the dependencies between the domain variables over the entire relational domain. Therefore, the PRM not only learns the correlations between the attributes in the same table as in the BN, but also learns the correlations between the attributes of different tables that are linked in different ways via foreign keys. The learnt PRM model provides a statistical model that can be used to answer many interesting inference queries about any aspect of the relational domain given the current status and relationships in the database.

In the PRM learning, an abstract model of the relational domain is built, which can be instantiated in different ways for each object of interest. The model specifies a template of the probability distribution over the relational database. The template includes: the relational schema of the domain that consists of the relational tables, attributes and links, and the probabilistic schema that describes the probabilistic dependencies between the domain attributes.

To understand the concepts of the PRM, consider a university relational domain with a set of tables, attributes, and relations describing a set of students and their units. Using this relational structure, the PRM constructs a probabilistic schema describing the dependencies between the domain attributes. The probabilistic schema consists of a set of nodes and directed links, representing the domain variables and the dependencies between them, respectively. Consequently, the probability distribution of each node depends only on the parents defined in the model, and hence the PRM specifies the probability distribution over the relational domain.

Using the PRM abstract (class) dependency model, new models can be created for new instances (students or units) of the relational domain. For example, for a specific unit u_i, a new model will be instantiated, such that it consists of the objects and relations related to

u_i. However, this model will be tied with the parameters of the class dependency model, and thus can be used for further inference or exploratory analysis.

PRMs have been successfully applied to different real-world applications, such as for selectivity estimation in databases (Getoor *et al.*, 2001d), for student modelling in virtual laboratories (Noguez *et al.*, 2007), in the medical domain for tuberculosis epidemiology study (Getoor *et al.*, 2004) and in the web domain for hypertext classification (Getoor *et al.*, 2001c). Furthermore, several extensions of PRMs were also proposed, including: the approach by Li and Zhou (2007) that extends PRMs to learn from incomplete data, the approaches by Getoor *et al.* (2001a) for modeling link uncertainty, and the approach by Getoor *et al.* (2000) to use class hierarchies in learning PRMs.

The several applications reported in the literature of PRMs indicate the effectiveness of this learning technique in modelling relational data and capturing the important dependencies in the domain. However, an essential task of mining relational domains is also the ability to use the learnt model for classification, i.e. use the learnt model to classify new instances not seen before. Cowell *et al.* (1993) show that even for perfect models that discover the correct dependencies in the domain, they are not necessarily practical for classification.

This task could be even more complicated in the case of imbalanced training data, where the data is overrepresented by one class as compared with other classes. The imbalanced class problem is a common real-world problem and has caused serious performance degradation for most standard learning techniques (Japkowicz and Stephen, 2002). This chapter reviews the PRM learning technique and its effectiveness to be used for mining and classification in real-world relational domains. In particular, the review will focus on the problem of mining imbalanced relational domains.

The layout of this chapter is as follows. Section 2.2 introduces the basic elements, concepts and the learning algorithm for PRMs. Section 2.3 raises the problem of employing PRMs for mining imbalanced relational domains. Then, Section 2.4 introduces the imbalanced class problem and discusses previous work that has been performed in this field. Sections 2.5 and 2.6 continue the discussion about the imbalanced class problems, focusing on the classification of multiple imbalanced attributes and multi-class imbalanced classification. Next, a discussion about mining imbalanced relational domains is presented in Section 2.7. Finally, Section 2.8 summarizes the chapter.

2.2 Probabilistic Relational Models

The Probabilistic Relational Models specify a template for a probability distribution over a relational database. The template consists of two components: the relational component that describes the relational schema of the domain, and the probabilistic component that describes the probabilistic dependencies in the domain. The following sections describe the semantics of each schema and also describes the learning algorithm for PRMs. This chapter uses the same notations and conventions as those used by Getoor (2002) to illustrate the concepts of PRMs.

2.2.1 Probabilistic Relational Model representation

2.2.1.1 Relational schema

The relational schema of PRMs is used to specify the type of objects and relations in the relational domain. A relational schema $\mathcal{R}$ consists of a set of classes (tables) $\mathcal{X} = \{X_1, .., X_n\}$ and a set of relations $\{R_1, .., R_m\}$. Each class is associated with a key attribute $X.K$, a set of descriptive attributes $A(X)$ and a set of reference slots $R(X)$. A descriptive attribute A of class X (denoted as $X.A$) is a standard attribute that has a set of finite values $\mathcal{V}(X.A)$.

The set of reference slots of class X represents the set of foreign keys of the class. A reference slot ρ of class X is denoted as $X.\rho$. The relational schema also defines the range type of each reference slot, thus, for a reference slot $X.\rho$, the domain type (Dom[ρ]) is X and the range type (Range[ρ]) is Y for a class Y in $\mathcal{X}$.

For each reference slot ρ, an inverse slot ρ^{-1} can be defined as the inference function of ρ. PRMs also define the notations of a reference slot chain, $\tau = \rho_1, .., \rho_m$, represented as a sequence of one or more reference slots. The reference slot chain describes how objects from different classes are related indirectly.

Getoor (2002) used a simple relational schema of a university domain to illustrate the relational language of PRMs. In this chapter, the classical company database usually used to explain database systems, describing employees and departments relationships, will be used to illustrate the notations of PRMs. Fig. 2.1 shows the schema of the company domain.

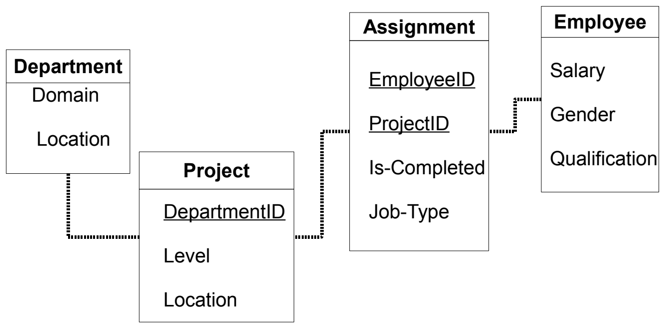


Figure 2.1: The relational schema of the company domain

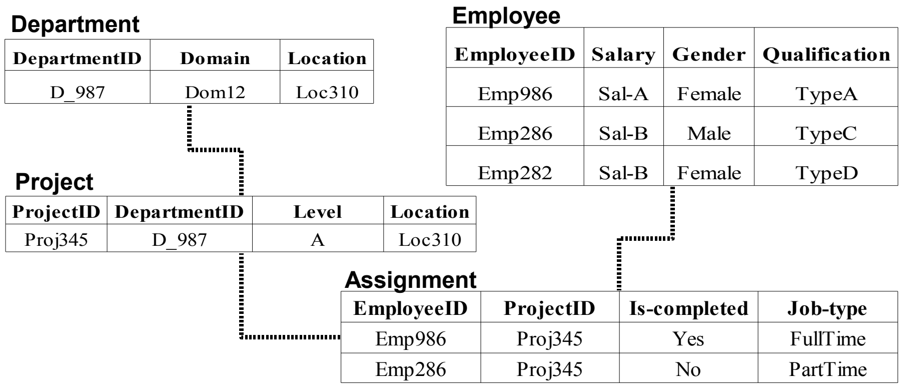
In this domain, the company schema consists of four classes: *Employee*, *Project*, *Department*, and *Assignment*. Each class is associated with a set of descriptive attributes. For example, the *Employee* class includes the ‘Salary’, ‘Gender’, and ‘Qualification’ attributes, and the *Project* class includes the ‘Location’ and ‘Level’ attributes.

Each descriptive attribute in the domain has a set of specified values, for example, the *Assignment* class has the descriptive attributes: ‘Is-Completed’ and ‘Job-Type’, which take values of {Yes, No} and {Full-Time, Part-Time}, respectively.

Additionally, to allow objects of one class to refer to objects of other classes, reference slots are used, such as in the *Assignment* class that has a reference slot ‘EmployeeID’ with range type *Employee*. The reference slots of the relational schema shown in the figure are underlined. An example of a slot chain linking objects of different classes is: ‘Assignment.ProjectID.DepartmentID’ denoting the department managing a particular assignment of a given project.

PRMs also specify an instance $\mathcal{I}$ of the relational schema $\mathcal{R}$ to denote an interpretation of the schema components. $\mathcal{I}$ specifies the set of objects of each class X and for each object it specifies the values of the descriptive attributes and the reference slots. Fig. 2.2 shows an example instantiation of the company domain. In this example, the instance shows that the department ‘D-987’ is running one project ‘Proj345’, with two employees ‘Emp986’ and ’Emp286’ working on this project.

Moreover, PRMs use the notation: *relational skeleton* (σ_r) to denote a partial specification of the relational instance $\mathcal{I}$. σ_r specifies the set of objects for each class and the relationships between the objects. However, it does not specify the values of the descriptive attributes. Each attribute in the skeleton is considered as a random variable. Given the skeleton σ_r, PRMs define the second component: the probabilistic schema.



Department

DepartmentID	Domain	Location
D_987	Dom12	Loc310

Employee

EmployeeID	Salary	Gender	Qualification
Emp986	Sal-A	Female	TypeA
Emp286	Sal-B	Male	TypeC
Emp282	Sal-B	Female	TypeD

Project

ProjectID	DepartmentID	Level	Location
Proj345	D_987	A	Loc310

Assignment

EmployeeID	ProjectID	Is-completed	Job-type
Emp986	Proj345	Yes	FullTime
Emp286	Proj345	No	PartTime

Figure 2.2: An instantiation example of the company schema

2.2.1.2 Probabilistic schema

A PRM probabilistic schema Π is composed of two components: the dependency structure $\mathcal{S}$ and the model parameters $\theta_{\mathcal{S}}$. $\mathcal{S}$ is defined by assigning a set of parents $Pa(X.A)$ for each descriptive attribute $X.A$. The parents of an attribute $X.A$ are other attributes of the domain that have direct influence on $X.A$.

The structure is represented graphically by a set of nodes, representing the domain attributes, and a set of directed links, showing the dependencies between the domain attributes. There are two types of formal parents of any attribute $X.A$: parents from the same class X or parents from related classes via the slot chain.

PRMs also address the dependency via a slot chain with a multi-set of values, i.e. the slot chain is referring to a set of objects rather than a single one. This multi-set of values is usually encountered in relationships of cardinality ‘zero or more’. For example, in the company example shown earlier, the ‘Salary’ of an employee can depend on the ‘Job-Type’ of the *Assignment* class. However, as each employee may be working on a different number of assignments (zero or more), the varying dependency of ‘Job-Type’ objects must be represented in a different way.

PRMs use the aggregation function γ to address this issue. As in database systems, this function combines the multi-set values into a single value, using functions such as: mean, mode, median, maximum, etc. Therefore, for a parent U with multi-set values, the child attribute $X.A$ will depend on some aggregated property of the multi-set values of U.

An example of dependency structure of the company example is shown in Fig. 2.3, showing

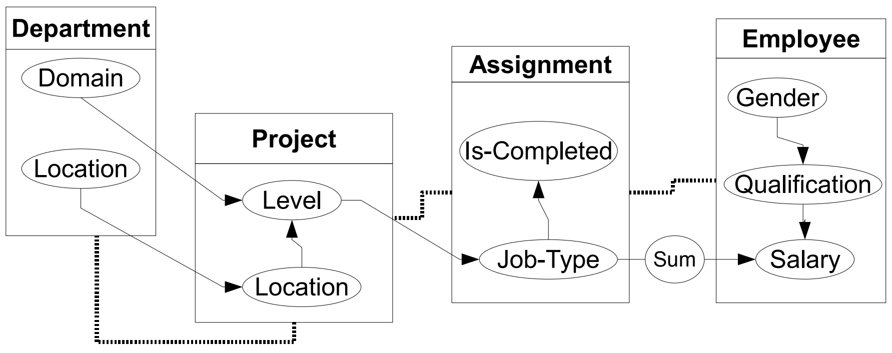


Figure 2.3: An example of a dependency structure of the company domain

the dependencies between the domain variables. In this example, this model shows that the employee 'salary' depends directly on two attributes: the 'qualification' attribute from the same class, and the sum of 'Job-Type' assigned to the employee from the related *Assignment* class .

By defining a set of parents $Pa(X.A)$ for each attribute $X.A$, the local conditional probability distribution (CPD) can be obtained. The CPD of an attribute $X.A$ specifies $P(X.A|Pa(X.A))$, which denotes the probability distribution over the attribute's value for each possible instantiation of the parents. The entire set of CPDs of the probabilistic model forms the model parameters $\theta_{\mathcal{S}}$.

By using the relational and probabilistic schema of PRMs, a PRM model can be summarized as follows:

Definition 1. *(reproduced from (Getoor, 2002)) For a relational schema $\mathcal{R}$ that consists of a set of classes $\mathcal{X} = X_1, .., X_n$, a Probabilistic Relational Model (PRM) Π is defined as follows:*

- *Each descriptive attribute $X.A$ has a set of parents $Pa(X.A) = (U_1, .., U_l)$, where each parent U_i is either from the same class X; or from a related class via the slot chain. If the slot chain is not single-valued, then aggregation γ is used.*
- *Each descriptive attribute $X.A$ has a conditional probability distribution (CPD) that specifies $P(X.A|Pa(X.A))$.*

Therefore, for a set of random variables in a relational skeleton, PRMs specify the probability joint distribution over the values of those random variables. As in BNs, this is obtained by defining a node for each random variable and specifying the parents and the

CPD for each attribute. The joint probability distribution of the PRM model can then be computed as follows (Getoor, 2002):

$$\begin{aligned} P(\mathcal{I}|\sigma_r, \mathcal{S}, \theta_{\mathcal{S}}) &= \prod_{x \in \sigma_r} \prod_{A \in \mathcal{A}(x)} P(\mathcal{I}_{x.A} | \mathcal{I}_{Pa(x.A)}) \\ &= \prod_{X_i} \prod_{A \in \mathcal{A}(X_i)} \prod_{x \in \sigma_r(X_i)} P(\mathcal{I}_{x.A} | \mathcal{I}_{Pa(x.A)}) \end{aligned} \tag{2.1}$$

Getoor (2002) states that his expression shares the same principle as the chain rule for BNs. But also points out two main differences: (1) the set of parents of a random variable is not fixed as in BNs and can change based on the relational skeleton, (2) the parameters in the PRM model are tied, thus, the CPDs of attributes of objects in the same class are identical.

As in BNs, it is important to ensure the coherence of the probability distribution, i.e. the probabilities of all instances sum to 1. In BNs, this is satisfied if the dependency structure is acyclic, thus, a variable does not depend on itself directly or indirectly (Pearl, 1988).

Similarly, PRMs employ the acyclic principle to ensure that these probabilities define a coherent probability distribution. However, in PRMs, it is important to ensure that the dependency structure is acyclic for any resulting skeleton. Getoor (2002) proved that if the class dependency structure is acyclic, then each attribute will never depend on itself (directly or indirectly), and any resulting relational skeleton is also acyclic.

2.2.2 Learning PRMs

This section describes the task of learning PRMs from relational data using the PRM language and semantics defined in the previous sections. For learning the PRM model, the input consists of the relational schema and training data. The relational schema specifies the domain classes, attributes of each class and the relationships held in the domain. On the other hand, the training data is a complete instance of the specified relational schema.

The PRM learning problem consists of two tasks: parameter estimation and structure learning. In parameter estimation, it is assumed that the PRM dependency structure is given, and hence it is required to define the CPDs of the domain attributes. In structure learning, the dependency structure is unknown, and therefore, it is required to learn both the dependency structure and the associated parameters from the training database. The following sections discuss each task of PRM learning.

2.2.2.1 Parameter Estimation

In parameter estimation, the input consists of the domain relational schema, a complete training instance $\mathcal{I}$, and the structure dependency $\mathcal{S}$, which specifies the set of parents for each attribute. Consequently, it is required to learn the structure parameters $\theta_{\mathcal{S}}$ by defining the CPD of each attribute in the domain.

In this task, the *likelihood function* is used to find the probability of the data given the model. In PRMs the likelihood of a parameter set $\theta_{\mathcal{S}}$ is defined as (Getoor, 2002):

$$L(\theta_{\mathcal{S}}|\mathcal{I},\sigma,\mathcal{S}) = P(\mathcal{I}|\sigma,\mathcal{S},\theta_{\mathcal{S}}). \tag{2.2}$$

Taking the log of this function:

$$\begin{aligned} l(\theta_{\mathcal{S}}|\mathcal{I},\sigma,\mathcal{S}) &= logP(\mathcal{I}|\sigma,\mathcal{S},\theta_{\mathcal{S}}) \\ &= \sum_{X_i}\sum_{A\in\mathcal{A}(X_i)}\left[\sum_{x\in\sigma(X_i)} logP(\mathcal{I}_{x.A}|\mathcal{I}_{Pa(x.A)})\right]. \end{aligned} \tag{2.3}$$

Moreover, using the theory of BNs, the *maximum likelihood* parameter estimation can be utilized to find the parameter $\theta_{\mathcal{S}}$ that maximizes the likelihood $L(\theta_{\mathcal{S}}|\mathcal{I},\sigma,\mathcal{S})$ for a given $\mathcal{I},\sigma$, and $\mathcal{S}$. This can be computed as the summation of terms corresponding to the domain attributes, and each of these terms can be maximized independently.

For many parametric models, maximum likelihood can be obtained via *sufficient statistics* that summarizes the data. For the case of multinomial CPDs, the sufficient statistics are the count of times of observing each of the different values of the attribute $X.A$ and its parents. This counting can be done in a straightforward manner using a standard database query.

Proposition 1 (reproduced from (Getoor, 2002))**.** *The maximum likelihood parameter setting $\theta_{\mathcal{S}}$ for multinomial CPDs can be defined as follows:*

$$P(X.A = v|Pa(X.A) = \boldsymbol{u}) = \frac{C_{X.A}[v, \boldsymbol{u}]}{\sum_{\acute{v}} C_{X.A}[\acute{v}, \boldsymbol{u}]}$$

where $C_{X.A}[v, \mathbf{u}]$ is the number of times counted in the training instance $\mathcal{I}$, for $\mathcal{I}_{X.A} = v$ and $\mathcal{I}_{Pa(X.A)} = \mathbf{u}$.

However, maximum likelihood can often overfit the data in many cases, thus in Bayesian networks a prior distribution over the parameters is used to avoid this problem. Using the same concept for BNs, PRMs have two assumptions: first, parameter independence, such that the priors over parameters $\theta_{X.A|v}$ are independent for different $X.A$ and v. Second, the prior over $\theta_{X.A|v}$ is a *Dirichlet* distribution (Heckerman, 1998). Therefore, in case of multinomial CPDs satisfying these two assumptions, with hyper-parameters $\alpha_{X.A}[\acute{v}, \mathbf{u}]$, PRMs define the posterior as follows (Getoor, 2002):

$$E_\theta[P(X.A = v|Pa(X.A) = \mathbf{u})|\mathcal{I}] = \frac{C_{X.A}[v,\mathbf{u}]+\alpha_{X.A}[v,\mathbf{u}]}{\sum_{\acute{v}} C_{X.A}[\acute{v},\mathbf{u}]+\alpha_{X.A}[\acute{v},\mathbf{u}]}$$

2.2.2.2 Structure Learning

Structure learning can be defined as learning a dependency structure that represents the data well. This task includes three important subtasks:

- Determining the set of legal dependency structures.
- Evaluating the candidate structures.
- Searching for a good structure with a high score.

Legal Structures. The set of possible structures is defined by the set of parents for each attribute $X.A$. There is an infinite number of possible structures even for a simple domain. However, only the legal structures, that provide coherent probability models, should be considered. In order to ensure that the structure candidate is legal, PRMs requires that the class dependency structure is acyclic, as described in the previous sections. Thus, only dependency structures that satisfy this requirement are considered as legal candidates.

Evaluating Different Structures. In this task, it is required to evaluate the different legal structures and find the one that represents the data well. PRMs adapt the Bayesian selection model for this task by using a scoring function. Therefore, the posterior probability of structure $\mathcal{S}$ can be defined as the posterior probability of the structure given the data I. This probability can be computed using Bayes rule, as follows (Getoor, 2002):

$$P(\mathcal{S}|\mathcal{I}, \sigma) \propto P(\mathcal{I}|\mathcal{S}, \sigma)P(\mathcal{S}|\sigma) \tag{2.4}$$

This score is composed of two main components: the structure prior probability and the probability of the data given that structure. For the first component $P(\mathcal{S}|\sigma)$, PRM assumes that the structure selection is independent of the skeleton, and thus $P(\mathcal{S}|\sigma) = P(\mathcal{S})$.

The second component represents the *marginal likelihood* (Getoor, 2002): $P(\mathcal{I}|\mathcal{S}, \sigma) = \int P(\mathcal{I}|\mathcal{S}, \theta_{\mathcal{S}}, \sigma)P(\theta_{\mathcal{S}}|\mathcal{S})d\theta_{\mathcal{S}}$. In the case of using the parameter independent Dirichlet prior, this integral is simply a product of integrals.

Proposition 2 (reproduced from (Getoor, 2002)). *For $P(\theta_{\mathcal{S}}|\mathcal{S})$ with parameter independence and Dirichlet with hyper-parameters $\alpha_{X.A}[v, \boldsymbol{u}]$, and a complete assignment $\mathcal{I}$, the marginal likelihood $P(\mathcal{I}|\mathcal{S})$ equals*

$$\prod_i \prod_{A \in \mathcal{A}(X_i)} \left[\prod_{u \in \mathcal{V}(Pa(X_i, A))} DM(\{C_{X_i.A}[v, \boldsymbol{u}]\}, \{\alpha_{X_i.A}[v, \boldsymbol{u}]\}) \right]$$

where $DM(\{C[v]\}, \{\alpha[v]\}) = \frac{\Gamma(\sum_v \alpha[v])}{\Gamma(\sum_v (\alpha[v]+C[v]))} \prod_v \frac{\Gamma(\alpha[v]+C[v])}{\Gamma(\alpha[v])}$, and $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$ is the Gamma function.

In this marginal likelihood each term corresponds to $P(X.A|\mathbf{u})$ where $\mathbf{u} \in \mathcal{V}(Pa(X.A))$. The term $P(X.A|\mathbf{u})$ depends only on hyper-parameters $\alpha_{X.A}[v, \mathbf{u}]$ and sufficient statistics $C_{X.A}[v, \mathbf{u}]$ for $v \in \mathcal{V}(X.A)$. Additionally, as for the prior over parameters values for each possible structure, the PRM learning algorithm assumes a uniform Dirichlet prior.

Structure Search. In this stage, PRMs employ the scoring function to evaluate the structure candidates and returns the high-scoring structure. Finding the best scoring structure is NP-hard (Chickering, 1996), and therefore PRMs use a heuristic search procedure. An example of such an algorithm is greedy hill-climbing search (Heckerman, 1998). This algorithm first considers the current structure, and then improves it iteratively by a set of local transformations and picks the one with the highest score. The transformations include:

add, delete or reverse an edge. To overcome the problem of the local-maxima (Heckerman, 1998), random restarts can be performed, i.e. perform a number of random steps whenever a local-maxima is reached, then continue the greedy hill-climbing process.

Moreover, the PRM learning algorithm addresses two problems: the large number of structures, and the high cost of computing the sufficient statistics. These problems are handled in the PRM by using a phased heuristic search algorithm. This algorithm proceeds in phases, such that in each phase k, the PRM has a set of potential parents $Pot_k(X.A)$ for each attribute $X.A$. Thus, the space of structures is restricted to those parents in the potential list. Moreover, by this method, the PRM can compute the corresponding view of X.A, $Pot_k(X.A)$ in advance and then obtain the sufficient statistics for any subset of potential parents, which could potentially reduce the computation cost of the sufficient statistics.

The selection of the potential parents is performed in an iterative approach based on the slot chain. Thus, the PRM starts with an empty structure and then proceeds based on the links in the slot chain. Therefore, the search is first performed within the dependencies within the same class, then between objects that are directly related, then between objects that are indirectly linked by two reference slots, then by those that are three slots apart and so on. Thus, the PRM selects the new potential parents based on the current phase of the slot chain and re-iterates, stopping when no further improvement is made. This approach allows the algorithm to gradually search the large space of structures and at the same time paying more attention to dependencies between objects that are more closely related.

2.2.3 Inference in PRMs

PRMs do not require a new inference algorithm, as they can use existing inference algorithms. The inference can be made about new objects by transforming to a BN. The BN is constructed from the database using the learnt PRM class dependency graph, and using parameter tying. Thus, to perform an inference task about an object Obj_i, the PRM simply constructs a BN using the appropriate attributes of Obj_i and the set of objects related to Obj_i. Then, the class-level parameters for each attribute are tied to the resulting network. Given this BN, a standard BN inference algorithm can be used to perform the inference task.

2.3 Effectiveness of PRMs in mining imbalanced domains

PRMs offer a rich relational learning technique for learning from relational domains directly. The power of PRMs is derived from the popular BNs, and extended to learn the correlation in relational domains. However, it is important for a learning technique to obtain an accurate statistical model that shows the dependencies in the domain, so as to be able to employ this model for classification. Classification is an important task in real-world applications, and a good classification model is the one that achieves a high recognition rate in classifying unseen instances.

Generally, the research on learning statistical modelling, such as BNs, is focused only on obtaining models that best represent the data (Ezawa *et al.*, 1996). However, Ezawa *et al.* (1996) argues that these models may perform poorly when applied to specific goal, such as classification. Moreover, Cowell *et al.* (1993) show that even for perfect models that accurately capture the correlations in the domain, they are not necessarily practical for classification.

The classification task is often more complicated in the case of imbalanced training data. The imbalanced class problem is defined as having one class of data that is overrepresented by a large number of samples as compared to the other classes. Several investigations and experiments have reported that the classification performance of most well-known classifiers often degrades dramatically in the presence of imbalanced data (Japkowicz and Stephen, 2002; Batista *et al.*, 2004; Chawla *et al.*, 2004). This poor performance supports the earlier argument of Ezawa *et al.* (1996) and Cowell *et al.* (1993), and is often caused as a result of designing methods to learn from relatively balanced training data.

In imbalanced situations, the classification algorithms often get biased towards the majority class, generating more rules for the majority class, while ignoring those of the small/minority class. Consequently, most of the minority test samples are misclassified to be of the majority class, resulting in poor prediction for the minority samples. However, in several applications, the correct classification of the minority cases is more critical than the classification of the majority case. For example, consider the classical example of a medical application designed for detecting a certain disease that is usually rare when compared with the normal case. In such an application, it is more important to correctly detect people with the disease than detecting healthy people. Therefore, in this case, the required classifier is that which attains a high classification rate on the disease.

Several techniques have been proposed to handle the imbalanced class problem by either altering the training data or by biasing the learning algorithm internally. However, most of these methods are proposed to work within a set of certain assumptions or restrictions. A major restriction of these methods is learning from flat representation (non-relational data). However, as most of real-world data are nowadays stored as relational systems, the different interactions between the attributes in such domains needs to be considered when handling the imbalanced class problem. This indicates the need for additional research into methods to handle the imbalanced class problem in relational domains.

Learning from relational domains is gaining an increasing interest in machine learning, and a number of methods have been proposed for this task, such as FOIL (Quinlan and Cameron-Jones, 1993), CrossMine (Yin *et al.*, 2004) and TILDE (Blockeel and Raedt, 1998). However, these methods are designed to learn from relatively balanced domains, and hence are inadequate for handling the imbalanced class problem. Moreover, although the PRM has shown successful implementations to discover the correct relationships of different relational domains, the PRM is also restricted to learning from relatively uniform data distributions. Consequently, in real classification problems, the performance of the PRM could be degraded significantly as a result of not being modelled for the specific goal of classification on imbalanced domains.

In this thesis, the PRM classification on imbalanced data is explored. Therefore, the following sections will discuss the imbalanced class problem and the different approaches proposed to solve this problem in relational and non-relational domains. Furthermore, a review is also presented about two further aspects of the classification on imbalanced data. The first case is the multiple imbalanced attributes problem, in which the domain consists of more than one imbalanced attribute to be classified. The second case is the multi-class imbalanced problem, in which it is required to find a classification function that maps the input into an output of more than two classes.

2.4 Imbalanced class problem

A training dataset is characterized as imbalanced if there are significantly more examples of some classes compared to others. In a two-class problem, the imbalanced class problem is defined as having one class (majority) that is represented by a large number of samples, while the other class (minority) has only a few samples. Usually, in such situations it is often more critical to correctly detect the minority samples, as they represent the important rare cases, such as detecting oil spills in satellite images (Kubat *et al.*, 1998), unreliable telecommunication customers (Ezawa *et al.*, 1996), fraudulent telephone calls (Fawcett and Provost, 1997), and customer churn (Burez and Van den Poel, 2009).

Throughout this thesis, the term ‘minority’ will be used to denote the class with the small number of instances that are regarded as critical to be classified correctly, and ‘majority’ will denote the dominant class with the large number of samples.

Several approaches have been proposed to handle the imbalanced class problem. Most of these approaches were proposed for the special two-class problem and for single flat files. Generally, these approaches can be categorized into the following groups (Chawla *et al.*, 2004):

- **Data-level approaches**: At the data-level, the approaches attempt to re-balance the class distribution by re-balancing the training data. This re-balancing can be performed either by over-sampling the minority samples to match the size of the majority or by down-sampling the majority samples to match the size of the minority. Each method can be performed in a random manner or in a focused mode. The focused mode of re-sampling considers the data distribution. This approach first examines the distribution of the class, and then performs the sampling based on the information collected. A combination of the two sampling methods can be also performed. A detailed survey of the re-sampling methods and their behaviour is presented by Batista *et al.* (2004).

  Although re-sampling is the most common approach used in dealing with the imbalanced class problem, several shortcomings have been observed (Barandela *et al.*, 2003b; Japkowicz and Stephen, 2002). The random sampling approach could result in duplicate samples in the case of over-sampling, and in the case of down-sampling, there is a potential for losing useful and important data. Although, the focused mode of re-sampling offers a more advanced solution compared with random sampling, this approach results in an increased cost of data analysis.

- **Algorithm-level approaches**: At the algorithmic level, the proposed approaches try to bias an existing learning algorithm towards the minority class. Examples of this approach include: biasing the BN in favour of the imbalanced target attribute (Ezawa *et al.*, 1996), searching for the best intervals for both majority and minority classes (Kubat *et al.*, 1998), and vector quantization of the majority class (Nugroho, 2000).

  A popular example of this approach is cost-sensitive learning. In this approach, the mis-classification costs of the classes are varied, such that the algorithm is biased towards the minority class. This is achieved by increasing the mis-classification cost of the minority samples and reducing that of the majority samples (Pazzani *et al.*, 1994; Sun *et al.*, 2007). The main challenge of this method is setting the proper mis-classification costs, especially if this has to be carried out by non domain-experts (Japkowicz and Stephen, 2002). Experiments have shown that the algorithm-level approaches usually outperform the data-level approaches (Lee *et al.*, 2008; Japkowicz and Stephen, 2002).

2.4.1 The imbalanced class problems in relational domains

According to the rich literature on imbalanced learning techniques, most of the techniques were developed for learning from a single flat file, where the data is represented as a single data sheet that includes all the data attributes and corresponding values. However, the majority of real-world applications maintain their data as relational database systems instead of a single data file. In a relational database, the data would usually consists of several tables (relations) linked to each other in different ways via a set of foreign keys. Each relation consequently includes a set of attributes and the values of those attributes.

Therefore, learning techniques are required to address the imbalanced class problem in relational data, where attributes are linked to each other in different ways. Throughout this chapter, a target attribute will be used to denote the attribute of interest that is to be classified; and the corresponding relation of that attribute will be denoted as a target relation. Any relation other than the target relation in the database will be denoted as non-target relation.

Only few attempts have been made to handle the imbalanced class problem in multi-relational data. Proposed techniques include:

- **Cost-sensitive learning for structured data** (Sen and Getoor, 2008): Sen and Getoor (2008) proposed two approaches for cost-sensitive relational learning using Conditional Markov Networks (Taskar *et al.*, 2002). The main idea behind these approaches is based on the concept that exploiting the correlations in the relational domain can help in specifying the mis-classification costs of the individual samples. Consequently, these costs can be used in cost-sensitive learning to minimize the overall mis-classification cost. Sen and Getoor (2008) demonstrated that in relational domains, where data can be represented by a graph, such graphs can be employed effectively to identify the mis-classification costs of the examples. The mis-classification costs not only depend on the individual examples, but also on the relational costs, which are the mis-classification costs of related examples.

- **Learning from multiple views of a relational database (MVC-IM algorithm)** (Guo and Viktor, 2007): In this approach an ensemble (Dietterich, 2000) of classifiers is constructed, in which each classifier is trained on a different view of the relational database. This framework relies mainly on two observations: (1) different views of the relational domain would usually yield different results in terms of learning the minority class, (2) an ensemble of classifiers learned from these different subsets would often be more effective than the individual classifiers. Therefore, MVC-IM tries to achieve better prediction results of the minority class by combining the knowledge obtained from different views. This is achieved by firstly constructing multiple views of the relational database. Each view is essentially a relation (table) of the database aggregated with the records of the target attribute propagated from the target table. Next, a separate classifier is learned from each view. Finally, the view classifiers are combined using majority voting. Although this approach utilizes the ensemble approach that has shown to perform extremely well in different applications (Opitz and Maclin, 1999; Dietterich, 2000), it uses flat views of the database and hence does not exploit the relational structure of the data.

- **Multi-relational g-mean decision tree algorithm (Mr.G-Tree)** (Lee *et al.*, 2008): Mr.G-Tree algorithm is a tree-based algorithm proposed for handling the imbalanced class problem in relational domains. In this algorithm, the imbalanced class problem is addressed by employing the geometric mean (g-mean) (Kubat *et al.*, 1997) and the concept of best intervals proposed in the SHRINK algorithm (Kubat *et al.*, 1998). On the other hand, learning from relational data is handled by extending the propagation concepts of the relational learning technique: CrossMine (Yin *et al.*, 2004). For a relational database, Mr.G-Tree first defines the best interval/subset of each attribute in the target relation by employing the concepts of SHRINK. Then, using the g-mean measure, these intervals are used to define the splitting function of the root node of the decision tree. Then, the relational learning occurs in the Mr.G-Tree algorithm by propagating the records of the target attribute to a non-target relation in the domain. Consequently, the splitting function of the second node in the decision tree is defined using the same procedure of using the best intervals and g-mean measure. This process continues until all relations are explored. This algorithm is proposed for the special case of two-class classification. Moreover, this algorithm shares the same prorogation concept as that of MVC-IM, where target class labels are propagated to each relation. Therefore, as in MVC-IM, the learning in Mr.G-Tree algorithm does not fully exploit the different interactions between the different attributes in the domain.

	True Positive	True Negative
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

Table 2.1: The confusion matrix.

- **Multi-relational Naïve Bayesian classifier (R-NB)** (Xu *et al.*, 2008): In this approach, basic under-sampling and over-sampling methods are employed to handle the imbalanced class problem, and classification is handled by utilizing the Naïve Bayes classifier. However, this classifier is enhanced to learn from relational domains by dealing with each table separately using the attribute filter criterion and mutual information. Firstly, mutual information is computed between the target attribute and each attribute from each relation in the domain. Then, the attributes are ordered in descending order based on the mutual information, and the top attributes are selected for classification. In this approach, only the correlations between the domain relations and the target attribute are considered. Therefore, this approach is not suitable for classifying more than one target attribute. Moreover, this approach utilizes the over/under-sampling methods, which have major limitations in classification, as discussed earlier.

2.4.2 Evaluation measures

Accuracy is often used as the standard performance measure in machine learning, in which the percentage of accurately classified samples is computed. For a two-class (Positive, Negative) problem, the classification of the class samples can be viewed as a confusion matrix, shown in Table 2.1.

The confusion matrix can be used to define the following measurements:

- The traditional accuracy is calculated as:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.5}$$

- True Positive rate (TP_{rate}): the percentage of true positive examples correctly classified, defined as:

$$TP_{rate} = \frac{TP}{TP + FN} \tag{2.6}$$

- True Negative rate (TN_{rate}): the percentage of true negative examples correctly classified, defined as:

$$TN_{rate} = \frac{TN}{TN + FP} \tag{2.7}$$

- False Positive rate (FP_{rate}): the percentage of true positive examples misclassified to be of the negative class, defined as:

$$FP_{rate} = \frac{FP}{TN + FP} \tag{2.8}$$

- False Negative rate (FN_{rate}): the percentage of true negative examples misclassified to be of the positive class, defined as:

$$FN_{rate} = \frac{FN}{TP + FN} \tag{2.9}$$

However, for imbalanced data, neither the traditional accuracy or the other measurements defined above are practical, as the minority cases have little impact on the accuracy as compared to that of majority (Kubat *et al.*, 1998). For example, for a classification problem with the minority data represent only 2% of the data, and a classification algorithm that can successfully predict all the samples of the majority class, the accuracy would be 98%. Despite the high accuracy achieved in this problem, this measurement is really poor for applications where minority cases are more important to be correctly classified than the majority samples.

Therefore, to handle the evaluation of imbalanced classifiers, other methods were proposed. Considering the earlier two-class problem (Positive, Negative), where 'Positive' represents the minority class, the popular evaluation measures for imbalanced situation are:

1. **F-measure** (Sun *et al.*, 2007): this measure focuses only on one class. For example, if the focus is on the positive class, then the Positive Predicted value PP_{value} is used with the True Positive rate (TP_{rate}) to define the F-measure:

$$\text{F-measure} = \frac{2 \times TP_{rate} \times PP_{value}}{TP_{rate} + PP_{value}} \tag{2.10}$$

   The Positive Predicted value is defined as: $PP_{value} = TP/(TP+FP)$, which denotes the percentage of the predicted positive examples correctly classified. A high F-measure indicates that both TP_{rate} and PP_{value} are high.

2. **Geometric mean (G-mean)** (Kubat *et al.*, 1997): The G-mean measure is used in imbalanced classification algorithms to denote the average ratio of accuracy of both minority and majority classes. The G-mean is defined as:

$$G - mean = \sqrt{TP_{rate} \times TN_{rate}} \tag{2.11}$$

The G-mean is high when both accuracies of the majority and minority classes are high and when the difference between them is small. Using this measurement, the aim is to maximize the accuracy of both classes. The higher the G-mean, the better the classifier.

3. **Receiver Operating Characteristics (ROC)** (Fawcett, 2006): ROC curves are often used to evaluate the performance of imbalanced classification algorithms (Guo and Viktor, 2007; Ezawa *et al.*, 1996). This is due to the insensitivity of ROC curves to the change of class distribution. ROC curves are often used for classifiers that output the probabilistic scores of the classified classes, such as BNs and Neural Networks. In such classifiers, the predictions can be varied according to the threshold used. For each of these thresholds, the ROC curve visually demonstrates the trade-off between the FP_{rate} and the TP_{rate} by plotting them on the X-axis and Y-axis, respectively.

   A ROC curve shows the ability of the classifier to rank the positive examples relative to the negative examples. In a ROC curve, the point (0,0) represents the case of never predicting the positive examples correctly and the point (1,1) represents the case of always predicting the positive class. An ideal classifier is one that achieves $TP_{rate} = 1$ and $FP_{rate} = 0$. Thus, the closer the classifier curve is to the upper left corner of the graph, the better the model is. On the other hand, a classifier that resides along the main diagonal represents the performance of a randomly guessing classifier.

   The example in Fig. 2.4, shows the ROC curves for two classifiers. In the figure, the dotted line represents the performance of classifiers that assigns the samples randomly. In addition, in the figure, classifier A is better than B, because classifier A is located more closely to the upper left corner of the graph. The ROC curve offers a visual tool to obtain a good summary about the classifier performance. However, in order to compare several classifiers using ROC curves, the Area Under a ROC Curve (AUC) (Fawcett, 2006) is used. The AUC provides a single value representing classifier performance. A classifier with the highest AUC value is a better classifier.

   ROC curves measure the ability of a classifier to output good relative examples scores, i.e. the classifier is evaluated in terms of attaining accurate scores that discriminate between the two classes (Fawcett, 2006). Furthermore, an interesting property of a ROC curve is the insensitivity to the change in class distribution, as it depends only on the TP_{rate} and FP_{rate} and hence does not depend on the class distribution as in the G-mean and F-measure. Therefore, ROC curves and AUC

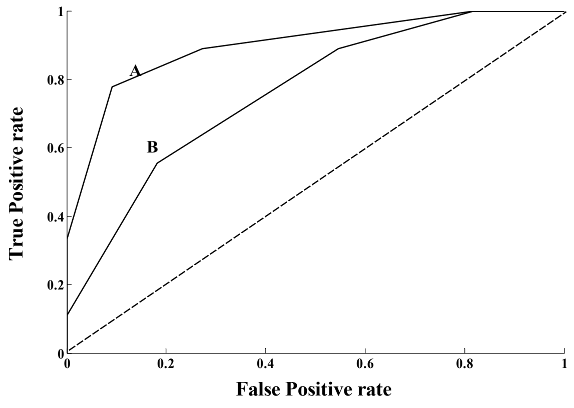


Figure 2.4: Examples of ROC curves.

values will be used as the evaluation measures throughout this thesis.

2.5 Classification of multiple imbalanced attributes

The previous section discussed the imbalanced class problem, and a number of techniques were presented to deal with such a problem. However, most of these techniques are proposed with the assumption of modelling a single imbalanced attribute at a time. However, in many real-world problems, it is of interest to classify two or more attributes rather than classifying a single attribute. Such problems include: face recognition application, where it is of interest to predict both face identity and head pose, and similarly in character recognition, in which both the font and character are required to be identified (Tenenbaum and Freeman, 2000).

A typical solution for classifying multiple attributes can be performed by constructing an independent model for each attribute. In this approach, it is assumed that the target attributes are independent, which is not the case in many domains. In addition, building independent models could significantly increase the number of models required for training and inference as compared with a single model of all the attributes. On the other hand, constructing a single model showing the different interactions between the target attributes not only reduces the number of models required, but could also improve the classification results by allowing the information of one attribute to reach a better conclusion about the other related attributes (Hiraoka and Mishima, 2002).

The classification of imbalanced relational attributes adds another level of complexity to the problem of classifying multiple attributes of a given domain. In such a situation, in addition to the problem of classifying multiple attributes, the learning technique must handle two other major tasks simultaneously: (1) capturing the different correlations between the relational attributes; (2) dealing with the imbalanced distribution in the training data that could degrade the classification performance.

Therefore, additional research is required to address the problem of classifying multiple attributes, and particularly for imbalanced attributes in relational domains. This section investigates this problem and reviews the different proposed approaches.

2.5.1 Solutions proposed for classifying multiple attributes

The problem of classifying multiple attributes can be viewed as building a classification model that utilizes the input of a given domain to find the class labels of a set of attributes (two or more attributes) of the domain. Unfortunately, this problem has not been well investigated in machine learning, despite its importance in many domains, especially in vision applications (Tenenbaum and Freeman, 2000).

Consider training data that consists of a set of attributes $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$, in addition to $\mathcal{A} = \{A_1, A_2, \ldots, A_m\}$, which represents the set of the target attributes that are of interest to be classified. Each target attribute has a set of class labels. The standard solutions proposed for classifying multiple attributes can be summarized as follows (Tenenbaum and Freeman, 2000; Hiraoka and Mishima, 2002):

1. **The independent approach:** In this approach, for each attribute A_i, an independent classifier C_i is constructed. Thus, the training data of C_i for classifying A_i, includes $\{X_1, X_2, \ldots, X_n, A_i\}$, i.e. the other target attributes rather than A_i are excluded from the training data. This approach relies on the concept that the attributes are often independent of each other and hence can be modelled independently.

   Although, this approach is the most common approach used for dealing with classifying multiple attributes, this approach has major shortcomings. For example, if the attributes are not in fact independent, then constructing independent models for this task has the major drawback of losing the different correlations between the attributes. Capturing the correlations in the domain can significantly improve the classification results, as the information discovered about one attribute can yield a better understanding about other related attributes. Moreover, as this approach is based on dealing with each attribute separately, it requires more training and testing phases than that of a single model.

2. **The combined approach:** In this approach, the multiple attributes $\mathcal{A}$ are combined in one complex attribute $\mathcal{B}$. Consequently, a single model is constructed to classify $\mathcal{B}$ using the training data $\{X_1, X_2, \ldots, X_n, B\}$. In this approach, the correlations between the attributes are lost, and it cannot be discovered if a target attribute A_i has an influence on another target attribute A_j. Moreover, for multi-class target attributes, the number of class labels of the combined complex attribute B grows rapidly, as each of the different combinations of the target attributes must be considered.

3. **The hierarchal approach:** In this approach, a similar method to that of decision trees is applied. This is achieved by assigning the root of the hierarchal structure to a classifier for a given attribute A_1, then for each class of A_1, a specialized classifier of A_2 is constructed, continuing the process until the last attribute has been considered. To illustrate the hierarchal approach, consider the domain defined earlier with input attributes $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$, but with only three target attributes $\mathcal{A} = \{A_1, A_2, A_3\}$. Each target attribute A_i has two classes: $\{A, B\}$.

   For training, the root classifier $C(Y_1)$ is constructed for classifying A_1. The training data of $C(Y_1)$ consists of the samples of $\mathcal{X}$ attributes and those of A_1. Subsequently, two classifiers are constructed for classifying Y_2, the first classifier $C(Y_2)a$ is trained on samples of Y_2 related to the records of $(Y_1 = A)$, and the other classifier $C(Y_2)b$ is trained on the samples of Y_2 related to the records of $(Y_1 = B)$. Likewise, for classifying the last attribute Y_3, four classifiers are constructed. Two classifiers $C(Y_3)a_1$ and $C(Y_3)b_1$ for the case of data of Y_3 related to data of $(Y_1 = A \ \& \ Y_2 = A)$ and $(Y_1 = A \ \& \ Y_2 = B)$, respectively. In addition, two classifiers $C(Y_3)a_2$ and $C(Y_3)b_2$ are created for the case of data of Y_3 related to data of $(Y_1 = B \ \& \ Y_2 = A)$ and $(Y_1 = B \ \& \ Y_2 = B)$, respectively. An illustration of this approach is shown in Fig. 2.5.

   For classifying new samples in the hierarchal approach shown in Fig. 2.5, the test sample is first classified by the root classifier $C(Y_1)$ for classifying the attribute Y_1. Accordingly, $C(Y_1)$ outputs the predicted class of Y_1. Based on the predicted class of Y_1, the second classifier for classifying the attribute Y_2 is evaluated. For example, if the predicted class of Y_1 is A, then the second classifier selected is that corresponding to the classifier trained on data related to class A of Y_1, i.e. classifier: $C(Y_2)a$. Consequently, the second selected classifier outputs the predicted class of attribute Y_2, and the process continues until the last attribute is classified.

   In this approach, the classification performance is significantly affected by the accu-

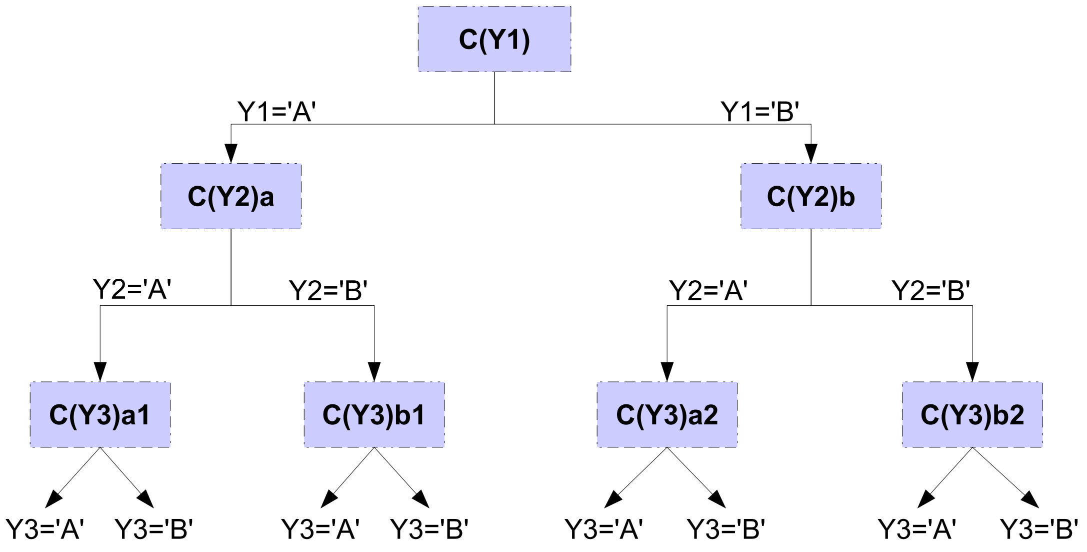


Figure 2.5: An illustration of the hierarchal approach for multiple attributes classification.

racy of the classifiers at the top of the hierarchy. This is due to the fact that the mis-classification committed by the top classifiers can not be corrected by the lower classifiers. Moreover, although in this approach, the top attributes are used to reach conclusions about the lower attributes, the classification of the lower attributes have no influence on that at the top of the hierarchy. In addition, a major concern in this approach is the number of classifiers required for the hierarchal structure, which grows enormously as the number of target attributes and classes increases.

A major drawback shared by these standard approaches is the inability to capture the interactions between the target attributes. Consequently, they lack the ability to use the information inferred about one attribute to improve that of the other target attributes. To address this issue, Hiraoka and Mishima (2002) proposed an approach for classifying two target attributes via mutual suggestion between the attribute classifiers. The idea behind this approach is based on using the classification results of one classifier to improve that of the other classifier and vice versa.

Consider training data that consists of $T = \{x_1, x_2, \ldots, x_n\}$, and two target attributes: C and S. In the approach proposed by Hiraoka and Mishima (2002), two classifiers are trained: F and G for classifying C and S, respectively. The F classifier is trained using as input: T and S, and outputs the class labels of C. The G classifier is trained using as input: T and C, and outputs the class labels of S. Thus, the F classifier outputs the conditional probability $P(C|T, S)$, and the G classifier outputs the conditional probability $P(S|T, C)$.

The two classifiers can be implemented using any standard classification method that produces the posterior probabilities of the class labels, such as Naïve Bayes.

For a new test sample t, a heuristic method is used to obtain the class of C and S attributes. Firstly, the test sample is classified by each classifier. Then, the output of each classifier is then used as a hint to update the inference results in the other classifier. This process continues until it converges to a stable result, when no improvement can be obtained.

The learning process in this approach is similar to the independent approach. However, it handles the problem of the separation between the target attributes, by classifying one attribute based on the data of the other target attributes, whereas in the independent approach, only the input data is considered in training and inference. The main contribution of this approach is in the inference phase, where the output of one classifier is used to update that of the other classifier.

The main concern in this approach, which is similar to that of the independent approach, is the number of classifiers required for training, especially for a large number of target attributes. Moreover, this approach has been proposed mainly for classifying two attributes. Therefore, it may not be practical in domains with more than two attributes, as it is not clear how to propagate the hints between more than two classifiers.

Moreover, for classifying multiple attributes, Tenenbaum and Freeman (2000) propose a framework to classify two attributes using a bi-linear model. For two attributes S and C, the framework initially specifies the training data by constructing an $M \times N$ matrix representing the class labels of the two attributes. Then, the framework fits a bi-linear model based on the observations in the matrix. Thus, a set of vector weights W_{ij} are computed to describe how the two attributes are linked together. For new test samples, the fitted models are used to predict the class labels of the attributes.

This approach relies on using bilinear models, which are simple and yet effective tools for modelling vision problems. However, this approach is designed for the special case of classifying two attributes, though Tenenbaum and Freeman (2000) argue that this framework can be easily extended to model mutiple attributes using multi-linear models. Different applications and extensions of this framework have been outlined (Grimes *et al.*, 2003; Vasilescu and Terzopoulos, 2002). However, this framework, and most of the extensions proposed, are based on the assumption of classifying independent target attributes, which is a valid assumption in vision problems, such as modelling the person identity and head pose in face recognition. However, this assumption would not be valid in a large number of real-world applications.

Despite the importance of the problem of classifying multiple imbalanced attributes in different real-world domains, this problem has not been well investigated and only few attempts are proposed to handle this problem. Moreover, most of the proposed techniques are designed for classifying two attributes.

2.6 Multi-class pattern classification from imbalanced data

Multi-class pattern classification techniques are widely used in different domains for various types of applications, such as text document classification (Stamatatos, 2008), bioinformatics (Al-Shahib *et al.*, 2005), and speech recognition (Even-Zohar and Roth, 2001). Multi-class pattern classification can be simply defined as building a system that correctly maps the input of a given problem to an output of one of more than two classes.

There is a rich literature of pattern classification techniques proposed for the two-class classification problem, such as support vector machines (SVM) (Hsu and Lin, 2002) that are designed specifically for binary functions. Along with binary classification, several algorithms have been also proposed for multi-class pattern classification, as it is often of interest in real-world domains to classify more than two pattern classes rather than classifying binary ones. Most of these algorithms are based on decomposing the multi-class problem into a set of two-class classification problems (Anand *et al.*, 1995; Rifkin and Klautau, 2004; Fürnkranz, 2002; Hastie and Tibshirani, 1998; Garcia-Pedrajas and Ortiz-Boyer, 2006).

Though these multi-class methods have been shown to perform extremely well in classifying multiple pattern classes in different domains, these methods were mainly proposed for learning from balanced datasets. However, many real-world domains have an imbalanced data distribution, where some classes of data have very few training examples as compared with the other classes. On the other hand, most of the approaches proposed to handle the imbalanced class problem are designed for classifying binary attributes. Therefore, this section presents a review of the challenges of multi-class classification in imbalanced domains, and discusses the different approaches proposed for this problem.

2.6.1 Multi-class pattern classification

Multi-class pattern classification can be defined as finding a function F that correctly maps the input features to an output of more than two classes. There are two ways to solve the multi-class problem (Lézoray and Cardot, 2008; Rifkin and Klautau, 2004): (1) by constructing a single machine that learns from all data at once, (2) by decomposing the multi-class problem into a set of binary classifiers and then combining them.

The first approach (all-at-once) considers all the training samples and classes at once and hence attempts to solve the multi-class problem as one single optimization problem (Hsu and Lin, 2002; Crammer and Singer, 2000; Weston and Watkins, 1999). The second approach is based on decomposing the multi-class problem into a set of smaller two-class problems, then constructing an independent binary classifier for each two-class problem, and subsequently combining the results of the binary classifiers (Anand *et al.*, 1995; Fürnkranz, 2002).

A number of experiments have been conducted in different domains to evaluate the performance of the two approaches of multi-class learning (Rifkin and Klautau, 2004; Hsu and Lin, 2002). These experiments have shown that the second approach is considered to be more effective and suitable for practical use, mainly for two reasons (Lézoray and Cardot, 2008; Rifkin and Klautau, 2004; Hsu and Lin, 2002; Ou and Murphey, 2007): (1) the first approach is slower as it needs to process all the data at once, while the second approach consists of a number of binary classifiers, where each classifier deals with only a small part of the data, (2) the classification is harder in the first approach, as the classifier needs to learn how to determine the separation boundaries of the large number of classes, whereas the binary classifiers have to only determine the boundary between two classes. Therefore, this chapter primarily focuses on the second approach and reviews the different decomposition techniques in the following sections.

2.6.1.1 Multi-class pattern classification using binary classifiers

This approach is based on splitting the multi-class classification problem into a set of binary classifiers. In this approach, for the K-class classification problem, N binary independent classifiers are built, each classifier trained independently using the relevant subset of the training data. The results of the N classifiers are then integrated to get the final output. The number of classifiers (N) depends on the modelling schema.

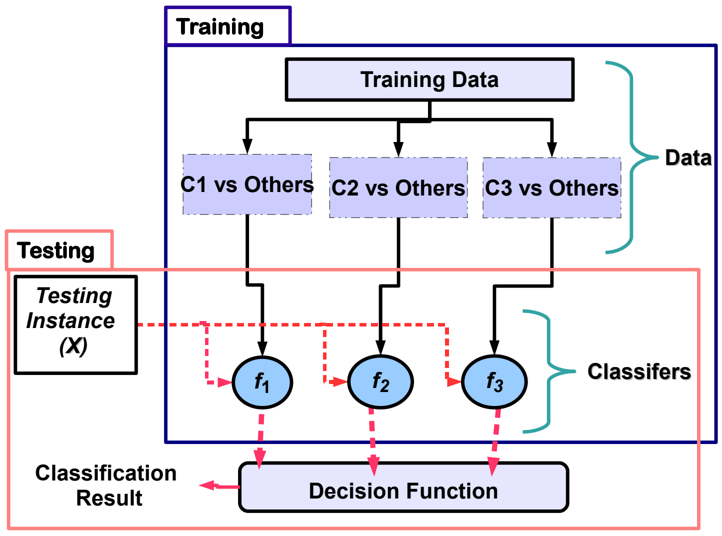


Figure 2.6: An illustration of One-Against-All (OAA) approach for three-class problem.

Several techniques have been proposed for decomposing the multi-class problem. In the following sections, a review of some of these techniques is presented. Moreover, illustrative diagrams are provided to demonstrate the concept of each technique. In these diagrams, rectangle, oval, and rounded rectangle shapes are used to represent the dataset/data subset, model/classifier, and the combination method, respectively.

Decomposing the multi-class problem can be generally categorized into three groups (Ou and Murphey, 2007):

1. **One-Against-All (OAA):**

   The OAA approach (Anand *et al.*, 1995; Rifkin and Klautau, 2004) splits the K-class problem into a set of K binary classifiers, with an independent classifier for each class. All classifiers are trained on the same data but with different class labels. The training data of classifier f_i consists of two sets: the samples of class C_i labelled as positives (1) and the samples of all the other classes labelled as negatives (0). Consequently, f_i has two possible outputs (1 or 0), to represent if the testing sample is of class C_i or not. The final classification result is obtained by combining the results of the K classifiers using a decision function: $F(x) = argmax_{i=1,....,K} f_i(x)$, in which the testing instance is classified to be of the class with the highest output value. Figure 2.6 presents the OAA method for the three class problem (C_1, C_2, C_3).

   In general, OAA has two shortcomings (Ou and Murphey, 2007; Murphey *et al.*, 2007; Tax and Duin, 2002): (1) imbalanced data: the training data of classifier f_i can be highly imbalanced, even if the original training data had an equal distribution of classes. This problem can be even worse if the data is originally imbalanced, (2) uncovered or overlapped classification regions: as each classifier is trained independently, there could be a situation where there are regions that are not covered by any classifier or covered by more than one. For example, consider the OAA system shown in Fig. 2.6. In this system, there could be a situation where each of the three classifiers rejects an instance and each outputs the probability score of class C_i to be zero, indicating that the instance certainly does not belong to any of the three classes. Similarly, there could be a situation where two classifiers claim that an instance belongs to their class with the same probability score. In these situations, other decision functions are required to resolve this ambiguity, such as classifying the instance randomly or assigning the instance to the class with the highest prior probability.

2. **One-Against-One (OAO):**

   The OAO approach, also known as the pairwise approach (Fürnkranz, 2002; Hastie and Tibshirani, 1998), decomposes the K-class problem into a set of binary classifiers, one for each possible pair of classes. The total number of classifiers will be $K(K-1)/2$. The training data of classifier f_{ij} consists of only the data of classes C_i and C_j and ignores all the others. This classifier is trained to discriminate between the two chosen classes only, and the output is (C_i or C_j) to indicate whether the input is of class C_i or C_j. Fig. 2.7 shows the OAO method for the three class problem (C_1, C_2, C_3).

   The simplest decision function to obtain the final result is by submitting the test sample to each of the binary classifiers, and then combining their predictions by majority voting, thus the test example is classified to be of the class that has the largest number of votes.

   The main advantages of OAO are: (1) fewer training examples are required for the binary classifiers, (2) OAO modelling does not suffer from the imbalanced issue, unless the two classes are originally imbalanced. However, the main concern in the OAO approach is the number of binary classifiers that grows in order of K^2 (Ou and Murphey, 2007).

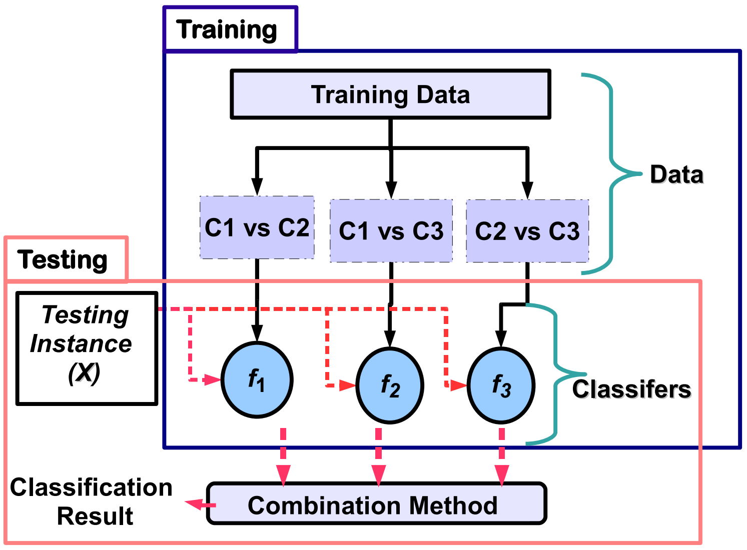


Figure 2.7: An illustration of One-Against-One (OAO) approach for three-class problem.

With regards to the problem of overlapped and unclassified regions, as with OAA, OAO can also face the same problem of overlapped regions, but it overcomes the problem of unclassified regions, due to the redundancy in the training of the pattern classes (Ou and Murphey, 2007). On the other hand, OAO can face another difficulty, where all classifiers disagree about the pattern class of a given instance (Tax and Duin, 2002). As an example of the disagreement situation, consider the OAO system shown in Fig. 2.7. In this system, there could be a situation where the classifiers (f_1, f_2, f_3) have the output (C_1, C_2, C_3), and thus disagree about which class the instance should belong to. Tax and Duin (2002) propose a solution to solve the overlapped and disagreement situations using combinations of approximate posterior probabilities.

Along with the majority voting method, several approaches were proposed for combining the binary classifiers. Lézoray and Cardot (2008) present a review of these approaches, and show that a new technique "stacking decoding" has shown substantial gain over the other approaches in the recognition rate. Stacking decoding (Savický and Fürnkranz, 2003) considers replacing the combination schema by a trainable classifier to resolve the conflict between the pairwise classifiers.

The idea behind stacking is derived from the conflict situations that could appear in majority voting. In majority voting, it is assumed that the relevant classifiers of class C_x would correctly predict C_x and hence provide more votes to C_x than the irrelevant classifiers (classifiers that have not been trained on classifying samples of class C_x). However, if the relevant classifier missclassifies the sample, then the

Classifier C1 vs.C2	C1 vs. C3	C2 vs. C3	Original Class
C1	C1	C2	C1
C2	C3	C2	C2
C1	C3	C3	C1
..	..	..	..
C2	C3	C3	C3

Table 2.2: An example of classification cases to demonstrate the concept of Stacking.

final output could be also misclassified. Table 2.2 (Savický and Fürnkranz, 2003) shows an example of this case for a three-class problem, where the third sample is missclassified by one of the relevant classifiers (C1 vs. C3) to be of class C3, and because the irrelevant classifier (C2 vs. C3) classifies the sample to be of class C3 as well, the final output predicts the sample to be of class C3. This example shows when the relevant classifiers make mistakes and how it affects the final output.

Therefore, stacking decoding was proposed to resolve the situation when the relevant classifiers make mistakes. In stacking, a new training set $D^{Stacking}$ is composed of the outputs of the binary classifiers and the original class value for each testing sample. $D^{Stacking}$ is then fed to a trainable meta classifier to produce the final classification result. There are two approaches to generate $D^{Stacking}$, either by considering the probability estimations of the different classifiers or by using the predicted output values of the classifiers to be fed to the meta classifier. The experiments conducted by Lézoray and Cardot (2008) have shown that the latter approach produces better results than the former.

3. **P-Against-Q (PAQ) model:** This approach relies on using M classifiers, each of which is trained for P classes against Q classes, where P and Q are one or more classes. Examples of PAQ modeling schemes include:

   - **Error-Correcting Output Code (ECOC):**
     This approach suggests using error correcting codes for the multi-class problem (Dietterich and Bakiri, 1995). Initially, a codeword (string of M bits) is assigned to each class, where M represents the number of binary classifiers. In the codeword for a class J, the i^{th} bit can be either (+1,-1) to indicate whether the i^{th} classifier is trained to classify the given class or not, respectively. This can be implemented as a Matrix ($K \times M$) for K-class problem, in which row i

represents the codeword of class C_i.

Then, for each testing instance x, a codeword is constructed from the results of the binary classifiers, in which the i^{th} bit of the codeword represents the output result of classifier i on x. The output codeword of x is compared to the rows of the matrix. If the output does not match exactly one of the rows, then the class with the minimum Hamming distance (Dietterich and Bakiri, 1995) is chosen.

The success of ECOC relies on good row and column separation (Ou and Murphey, 2007), so that each codeword is well presented. Different approaches have been proposed in this category of error-coding, including the unified approach by Allwein *et al.* (2001). In this approach the matrix has three values {-1, 0, 1} with 0 meaning "don't care". For the OAA approach, the diagonal elements are set to +1 and all the other elements are set to -1. For the OAO approach, in which each bit of the codeword corresponds to pair classes (C1,C2), the +1 is used for row C1, -1 is used for row C2 and 0 for all the other rows.

- **One-Against-Higher-Order (OAHO):**

  This approach (Murphey *et al.*, 2007) was originally designed to handle the imbalanced problem in multi-class pattern recognition. In this modelling approach, $(K-1)$ classifiers are constructed for an ordered class list=$\{C_1, C_2, ..., C_K\}$. The first classifier (C_1, C_{2+}) is trained with two sets of data, C_1 represents the samples of class C_1 being labelled as (1), and C_{2+} represents all other samples from the other classes being labelled as (0). The second classifier is trained on the data (C_2, C_{3+}) with two sets of data representing the samples of class C_2 as (1) and the samples of higher ordered classes $(C_3, ..., C_K)$ as (0), and so on, until the last classifier classifies C_{K-1} against C_K.

  The testing process in OAHO is hierarchical, as the testing instance is tested first by the first classifier (C_1, C_{2+}). If the instance is classified as C_1, then the example will be labelled as class C_1, and the process ends, otherwise the second classifier is used, and so on until the last classifier that classifies C_k against C_{k-1}.

  This hierarchical process in OAHO implies that the top level classifiers should be as accurate as possible, as any mis-classification caused by the top level cannot be corrected in the next steps. The classes can be ordered either randomly, based on class importance or based on the data size of each class. In order to resolve the problem of imbalanced classes in multi-class classification, the classes can be ordered based on the size of the training data of each class (Murphey *et al.*, 2007). Therefore, the class list is ordered to have the classes with the largest number of examples first. Thus the smaller classes are combined

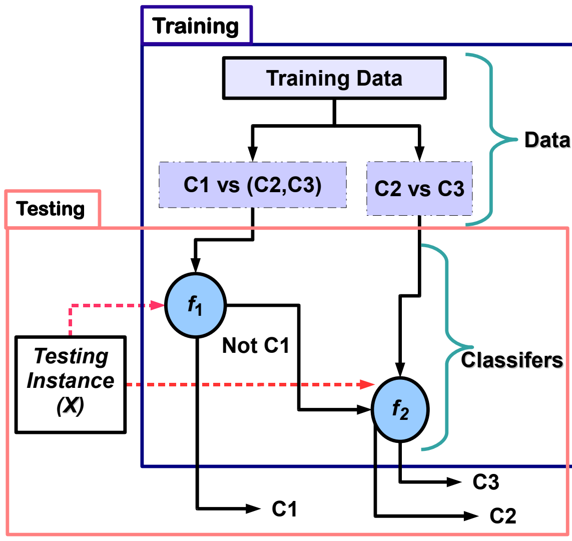


Figure 2.8: An illustration of One-Against-Higher-Order (OAHO) approach for three-class problem.

together and compared against the largest class at the higher levels of the hierarchy. Fig. 2.8 shows the OAHO method for the three class problem with order (C_1, C_2, C_3).

2.6.1.2 Combining two strategies

The All-and-One (A&O) method (Garcia-Pedrajas and Ortiz-Boyer, 2006) is proposed as a combination of the two popular multi-class techniques: OAA and OAO. This method uses both the OAO and OAA methods to combine the strengths of both methods and avoid the problems of each. This method is based on studying the characteristics of both methods, and examines the situations in which the methods fail. The study found that for a high percentage of classifications using OAA, the second best ranking output of OAA represents the accurate output when the first output fails. With regards to OAO, the study showed that the independent binary classifiers are highly accurate on their own. However, when these classifiers are combined, the results are incorrect, because the approach includes the irrelevant classifiers that have not been trained on the class of the given instance.

In A&O, both OAA and OAO methods are trained. For a new testing sample, the OAA method is used to select the first two candidate classes (C_i, C_j) and then the corresponding

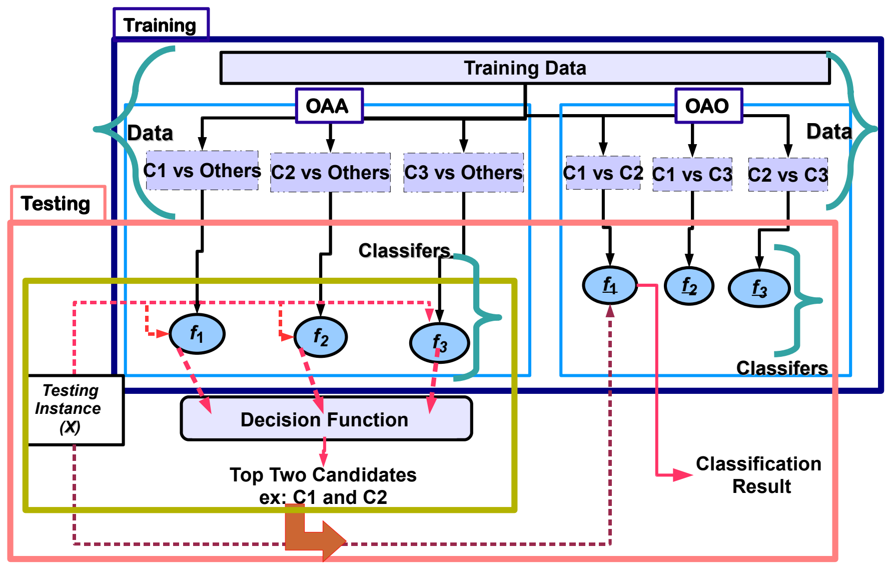


Figure 2.9: An illustration of All-and-One (A&O) approach for three-class problem.

OAO binary classifier f_{ij} is used to determine the final output. The main drawback of this approach is the number of classifiers to be trained. However, once the classifiers have been trained, the testing step needs only the K classifiers of OAA and one classifier of OAO, in comparison to the $K(K-1)/2$ classifiers of OAO (Garcia-Pedrajas and Ortiz-Boyer, 2006). Fig. 2.9 shows the A&O method for the three class problem (C_1, C_2, C_3).

2.7 Mining imbalanced relational domains

This chapter has primarily focused on the problem of mining imbalanced relational domains and reviewed the main work performed in this field. In terms of relational learning, the review presents the effectiveness of employing the Probabilistic Relational Models (PRMs) in learning from relational domains. The review also investigates the applicability of utilizing PRMs for mining imbalanced relational domains and points to the need for improving PRMs to handle the imbalanced class problem.

The imbalanced class problem was also discussed and a number of techniques were presented. The review particularly addressed three main problems: (1) the imbalanced class problem, (2) modelling multiple imbalanced attributes, and (3) multi-class imbalanced classification. However, as the main concern of this chapter is handling the imbalanced class problem in relational domains, most of the presented techniques were not practical and could be only implemented in certain contexts.

In terms of the first problem of handling the imbalanced class problem, the review shows that the imbalanced class problem has been well investigated in flat domains, whereas, few attempts are proposed to handle the problem in relational domains. The relational imbalanced techniques are proposed using different techniques and were applied in different domains. However, the common concept shared by these methods is the idea of propagating the records of the target attribute into the other non-target relations to perform the learning. An exception is the work proposed by Sen and Getoor (2008). As discussed before, this propagation hinders the ability to fully exploit the different interactions between the domain variables as achieved by PRMs. Moreover, these methods are only practical for applications with only one target attribute.

For the work proposed by Sen and Getoor (2008), the authors have shown that their approach can often lead to significant reductions in mis-classification costs. Yet, a major concern of this approach, as in any cost-sensitive learning algorithm, is the challenge of setting the suitable mis-classification costs of the individual samples (Japkowicz and Stephen, 2002).

In terms of modelling multiple imbalanced attributes, although this problem is highly relevant to different real-world problems, only a few approaches are proposed, especially for the particular case of classifying two *balanced* attributes. None of the proposed methods address the imbalanced problem or the relational learning for classifying multiple attributes.

As for the last problem of multi-class imbalanced classification, a large number of techniques are presented for handling multi-class classification. However, none of the presented methods was designed to learn from relational data. In terms of the imbalanced problem, OAHO is the only method that was proposed to handle the imbalanced problem for multi-class classification and proved to outperform the other methods for imbalanced multi-class learning (Ou and Murphey, 2007). However, OAHO includes the challenge of arranging the class list in the best order to minimize propagating the classification errors to the next levels of the hierarchy, as mistakes made at the top level cannot be corrected by the following levels.

As for the other multi-class methods: OAA and OAO, each method has its strengths in multi-class classification as pointed out by several papers (Rifkin and Klautau, 2004; Lézoray and Cardot, 2008; Garcia-Pedrajas and Ortiz-Boyer, 2006), yet they have major limitations when dealing with the imbalanced problem (Ou and Murphey, 2007). As for the ECOC approach, the experiments carried out by Ou and Murphey (2007) using multiple neural networks for the 10-class handwritten digit recognition problem and show that both OAA and OAO approaches achieve better recognition rates than ECOC. The new method All-and-One (A&O) combines the strengths of both OAA and OAO, but its performance on imbalanced multi-class domains has not been reported.

In summary, this review signifies the need for additional research on mining imbalanced relational domains. At the same time, the review demonstrates the potency of PRMs in learning from relational domains, which shows the gap in this learning technique to handle the imbalanced class problem in relational domains.

2.8 Conclusion

This chapter has reviewed the problem of mining relational domains. The review discussed the effectiveness of Graphical Models and Bayesian Networks in modelling real-world domains and their shortcomings in modelling relational domains. The chapter then discussed an extension of Bayesian Networks to handle relational learning, named Probabilistic Relational Models (PRMs). PRMs share the same underlying techniques of Bayesian Networks, but employ the concept of objects to handle the relational learning and inference. Despite the success of PRMs in modelling relational domains, they cannot handle the common imbalanced problem in many relational domains. Consequently, the chapter reviewed the imbalanced class problem, and particularly for the case of multi-class imbalanced learning and modelling multiple imbalanced attributes. The review showed that most of the presented techniques were designed to be employed in certain contexts and hence are not practical for relational domains. Conclusively, the review signifies the need for additional research for learning from relational imbalanced domains and pointed to the potency of improving PRMs for this task. The following chapters explore improving PRMs to handle the learning from imbalanced relational domains.

Chapter 3

PRMs for Mining a University Relational Database

The previous chapter pointed to the need for effective relational learning techniques to cope with the increasing use of relational database systems in different domains. Moreover, the previous chapter presented the Probabilistic Relational Model (PRM) as a relational learning technique, which has been applied successfully in several applications of different real-world relational domains. This chapter explores the PRM concepts in the educational domain to analyze a university relational database. The aim of this analysis is to explore the effectiveness of the PRM in capturing the most important dependencies in this relational domain, in particular for first year undergraduate students. Moreover, in this chapter, the significance of these dependencies is discussed in terms of improving the success rates of first year undergraduate students. Additionally, a visualization tool is provided to the domain users to illustrate the benefits that could be gained from the dependencies indicated by the PRM models.

3.1 Introduction

Chapter 2 described the concepts of the Probabilistic Relational Model (PRM) as an extension of Bayesian Networks (BNs) to the relational domain. PRMs have been successfully applied in several real-world relational domains such as: for selectivity estimation in databases (Getoor *et al.*, 2001d), for student modelling in virtual laboratories (Noguez *et al.*, 2007), in the medical domain for tuberculosis epidemiology study (Getoor *et al.*, 2004) and in the web domain for hypertext classification (Getoor *et al.*, 2001c).

This chapter explores the use of the PRM in the educational domain and particularly for investigating a university relational database. Universities usually use large relational database systems to record varied information about students, instructors, units and courses. This data can be used for mining interesting information such as: studying the retention factors of courses and modules, identifying students at risk in early stages to offer suitable support, and defining the best set of enrollment requirements of each degree.

An important part of university data is that related to the first-year of undergraduate study, due to the significance of this year in shaping students' attitudes towards learning (Astin, 1993). Mining the data of undergraduate students has been intensively explored in the literature by using different learning techniques such as Bayesian Networks and regression models. For example, regression analysis was applied by Bennett (2003) and Wetzel *et al.* (1999) to study the retention factors of undergraduate degrees, by Goold and Rimmer (2000) to investigate the performance factors of first year computing and by Budny *et al.* (1997) and Zhang *et al.* (2004) to study the influence of curriculum units on engineering student retention. On the other hand, Bayesian Networks were applied by García *et al.* (2007) to model students' learning styles and by Druzdze and Glymour (1994) for retention factors study.

Despite the success of Bayesian Networks and regression analysis in many real-world applications, these techniques are inadequate for modelling complex relational domains as discussed in the previous chapter. Therefore, this chapter adapts the relational learning technique (PRMs) to model a university relational database and in particular to analyze a first-year student database. PRMs, as in BNs, provide a compact and natural graphical representation that show the important dependencies in the domain. These dependencies could potentially point to interesting information about improving students' performances in their first year.

A toy university relational database was used by Getoor (2002) to illustrate the PRM conventions and semantics. The toy dataset consisted of three tables with two attributes and three/four tuples per table. In this chapter, a real university database is used to explore the effectiveness of a PRM in learning from a real relational university database with a larger number of tables, tuples and attributes that consists of about nine tables, twinty-nine attributes and thounsands of tuples.

In this chapter, the PRM will be employed on two undergraduate relational databases. The first database focuses on the data related to the students and their performances in first-year units. The goal of using this database is to explore the dependencies that could affect student performance in their first year units.

The second database focuses on the data of first year undergraduate students along with their performance in high school. This database will include the related data about students, their results in high school and their performances in first year in university. The aim of using this database is to investigate to which degree the results of high school and personal information such as address, age, and gender would affect student performance in first year in university. Moreover, a visualization tool is presented to illustrate the significance of the dependencies discovered by the PRM in this database.

The remainder of this chapter is structured as follows: Section 3.2 describes the first dataset explored in this chapter and discusses the important dependencies indicated by the PRM to improve student performance. This is followed by a description of the second dataset in Section 3.3. This section also presents the main dependencies discovered by the PRM in this domain and presents a visualization tool to demonstrate the results of the PRM model. Finally, section 3.4 concludes the chapter.

3.2 Dataset I: First-year relational database

This study dataset is derived from the student relational database used at Curtin University. The Curtin University dataset consists of thousands of tables and relationships holding student data from 1999 onwards. The extracted study dataset focused only on the data of the first year students of two undergraduate degrees: Bachelor of Computer Science (BCS) and Bachelor of Commerce (BCom). For each of these degrees, a separate dataset was extracted to include the data of first year students for the years 2001-2006. Importantly it was verified that the study plan of the first year did not change over these years enabling aggregation of all the data to form a large dataset.

Each dataset is organized into a set of tables and attributes describing the students' personal, academic and financial information, which is organized as follows:

- The `Personal_Info` table, which includes the student personal attributes:
  - Age: discretized into groups of {16-19, 20-29, 30-40},
  - Gender: {Male, Female},
  - Is_international: {Yes, No},
  - Is_English_home_language: {Yes, No},
  - Country_of_Birth and Citizen_Country: each takes values from a predefined set of countries.
- The `Academic_Info` table that holds the financial and academic attributes:

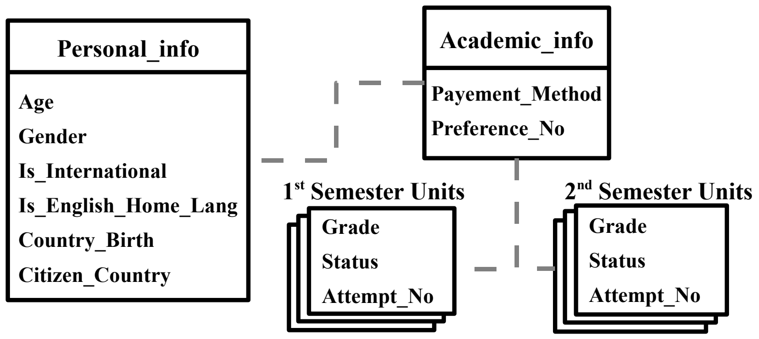


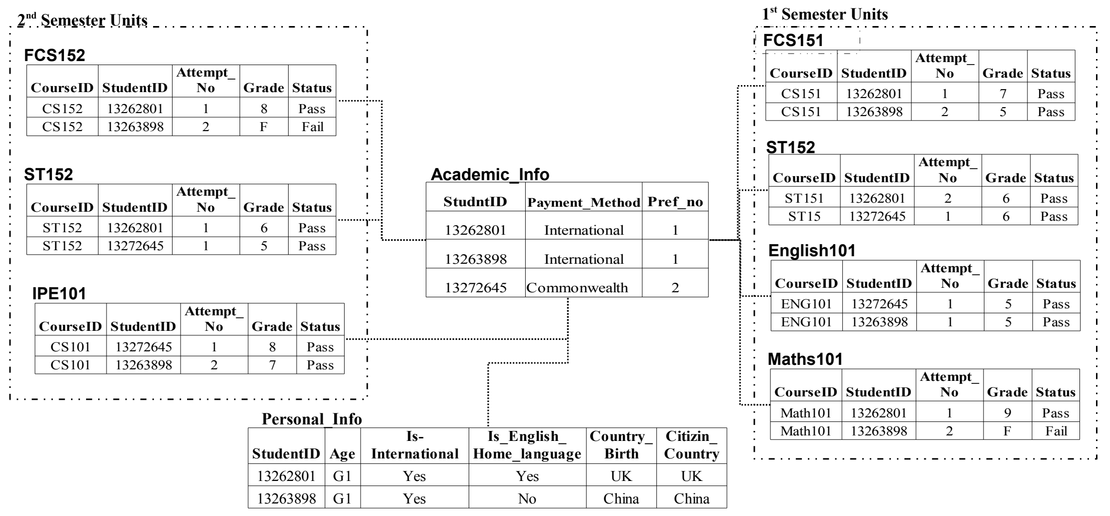
Figure 3.1: The relational database schema of first year undergraduate students.

  - Payment_Method: {Commonwealth, International}, representing how the student is charged,
  - Preference_No: {1, 2, 3}, describing the student's preference of study, where Preference_No = 1 indicates that the corresponding degree is the student's first preferred degree to study and so on.

- Number of tables representing the first year units, each unit table holds the student's performance of the unit:
  - Status: {Pass, Fail},
  - Grade: {F, 5, 6, 7, 8, 9}, representing the categories of {0-49, 50-59, 60-69, 70-79, 80-89, 90-100},
  - Attempt_No: {1, 2, 3}, indicating the number of times the student repeated the unit.

The `Academic_Info` table is associated with two key attributes: `Student_ID`, which links the `Personal_Info` of a specific student with the corresponding record of the table: `Academic_Info`; and `Student_Study_No` key attribute, which links the student records of the table `Academic_Info` with the related records from the units tables. Fig. 3.1 shows the relational database schema of first year undergraduate students.

The first year curriculum of the BCS and BCom degrees are shown in Tables 3.1 and 3.2, respectively. Fig. 3.2 shows an example instantiation of the BCS dataset. In this example, the instance shows the academic and personal information for a number of students, along with the student's units in first and second semesters.



2nd Semester Units

FCS152

CourseID	StudentID	Attempt_ No	Grade	Status
CS152	13262801	1	8	Pass
CS152	13263898	2	F	Fail

ST152

CourseID	StudentID	Attempt_ No	Grade	Status
ST152	13262801	1	6	Pass
ST152	13272645	1	5	Pass

IPE101

CourseID	StudentID	Attempt_ No	Grade	Status
CS101	13272645	1	8	Pass
CS101	13263898	2	7	Pass

Academic_Info

StudntID	Payment_Method	Pref_no
13262801	International	1
13263898	International	1
13272645	Commonwealth	2

Personal_Info

StudentID	Age	Is-International	Is_English_ Home_language	Country_ Birth	Citizin_ Country
13262801	G1	Yes	Yes	UK	UK
13263898	G1	Yes	No	China	China

1st Semester Units

FCS151

CourseID	StudentID	Attempt_ No	Grade	Status
CS151	13262801	1	7	Pass
CS151	13263898	2	5	Pass

ST152

CourseID	StudentID	Attempt_ No	Grade	Status
ST151	13262801	2	6	Pass
ST15	13272645	1	6	Pass

English101

CourseID	StudentID	Attempt_ No	Grade	Status
ENG101	13272645	1	5	Pass
ENG101	13263898	1	5	Pass

Maths101

CourseID	StudentID	Attempt_ No	Grade	Status
Math101	13262801	1	9	Pass
Math101	13263898	2	F	Fail

Figure 3.2: An example of the BCS dataset.

Semester	Unit Code	Unit Name	Prerequisite
I	ST151	Computer science 1	-
	Maths101	Mathematics 1	-
	FCS151	Foundation of computer science 1	-
	English101	Technical communication 1	-
II	ST152	Computer science 2	ST151
	FCS152	Foundation of computer science 2	FCS151
	IPE101	Introduction to programming	-

Table 3.1: The BCS first year curriculum.

3.2.1 Applying PRMs to the students dataset

This section presents the results of applying PRMs to the BCS and BCom datasets. The PRM models are learned by implementing the PRM techniques described in Chapter 2. In this chapter, the dependencies between the attributes are shown in different styles of arcs (bold, dashed, ...). The different styles of arcs are used in this chapter to denote the importance of each of the dependencies discovered by the PRM in terms of improving student performance, and does not imply that a certain arc-style is of different characteristics than others.

Semester	Unit Code	Unit Name	Prerequisite
I	BIS100	Business information systems 1	-
	ACCT100	Accounting 1	-
	LFW100	Legal framework 1	-
	ECON100	Economics 1	-
II	MGT100	Management 1	-
	MKT100	Marketing 1	-

Table 3.2: The BCom first year curriculum.

Attribute	Learned parent(s)
FCS152.grade	English101.grade, Maths101.grade
ST152.status	FCS152.status
IPE101.Attempt-number	FCS152.Attempt-number
ST151.Attempt-number	Academic_Info.Pref_No
FCS151.Attempt-number	Academic_Info.Pref_No
Personal_Info.Is_English_Home_Language	Personal_Info.Citizen_Country
Personal_Info.Is_International	Personal_Info.Citizen_Country

Table 3.3: A summary of the main PRM dependencies learned from the BCS dataset.

3.2.1.1 Bachelor of Computer Science (BCS)

The PRM dependency structure learned from the BCS dataset is shown in Fig. 3.3. The learned model shows the dependencies within the same table and between different tables. Table 3.3 shows a summary of the main dependencies learned from this dataset, especially with regards to the dependencies affecting units grades.

In the seven units in the BCS dataset, the model has repeatedly pointed to the correlations between the `grade`, the `status` and the `attempt-number` within the same unit. An explanation is that each of these attributes refers to the student success at the unit, and hence they are related.

In terms of the dependencies between the units, the model shows the dependence of the `FCS152` grade (studied in semester II) on the grades of `English101` and `Maths101` (studied in semester I). Also, the model shows correlations between the `FCS152` and the other two units of semester II (`ST152`, `IPE101`), as indicated by the links directed from `FCS152 status` and `attempt-number` to `ST152 status` and `IPE101 attempt-number`, respectively.

Therefore, the model shows the correlations between the Maths and English in semester I and `FCS152` in semester II, and then the correlations between `FCS152` and the other two

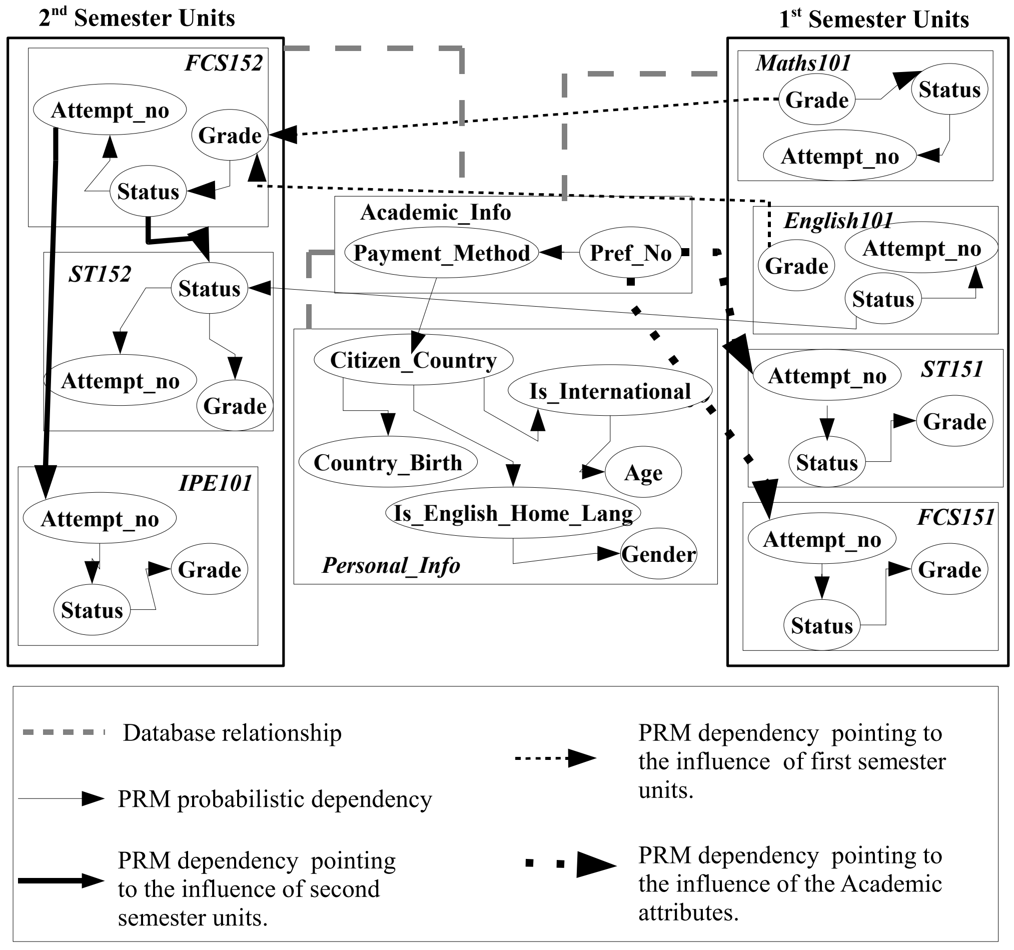


Figure 3.3: The PRM structure learned from the BCS dataset.

units of semester II (ST152, IPE101). The conditional probabilities of the model show that a student with high grades in Maths and English has a higher probability of passing the second semester unit: FCS152, and consequently higher probability of passing the other two units in semester II.

These probabilities indicate that the performance in semester II units is strongly related to the Maths and English results achieved in semester I. As both Maths and English are first semester units, then the identified dependencies and probabilities about Maths and English could suggest setting these two units as prerequisites for the second semester units. The dependencies of the second semester units ST152 and IPE101 on the other second semester unit FCS152, could suggest studying FCS152 first or at least at the same semester given that the FCS152 suggested prerequisites (Maths and English) are completed.

In order to investigate the PRM suggestions of setting Maths and English as prerequisites

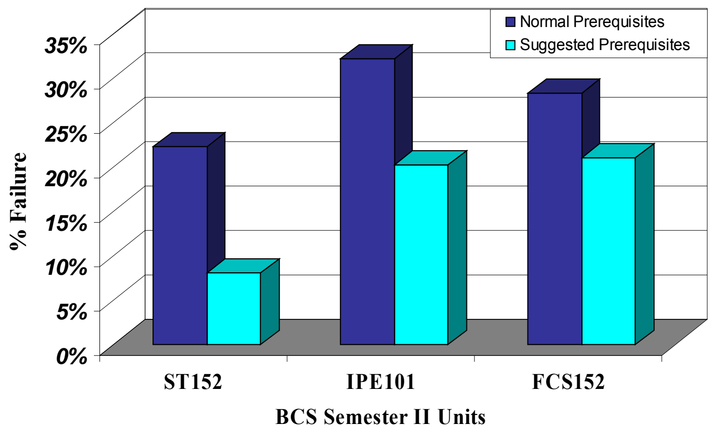


Figure 3.4: The comparison of students performance in semester II obtained by following the normal BCS prerequisites and the PRM suggested prerequisites.

for the second semester units, the BCS dataset was analyzed to draw a comparison between two sets of students: the set of students who enrolled into semester II units after passing the curriculum prerequisites shown in Table 3.1, and the set of students who enrolled into semester II units after passing the curriculum prerequisites in addition to passing Maths and English as suggested by the learned PRM model.

The BCS dataset is restricted by the curriculum prerequisites shown earlier in Table 3.1. As a result, the existing data of semester II units (`ST152` and `FCS152`) represent the set of students who must have passed the corresponding prerequisites. This restriction of the curriculum prerequisites limits the space of the data analysis to only examine the impact of adding Maths and English to the existing prerequisites, but not the analysis of removing the current perquisites and setting only those pointed to by the PRM model.

Fig. 3.4 shows the comparison of student success in semester II between the normal prerequisites and the suggested prerequisites. The comparison shows that the failure percentage in semester II units is less in the case of students passing Maths and English compared to passing only the curriculum prerequisites. This comparison supports the findings of the PRM model which recommend adding `Maths101` and `English101` as prerequisites for the second semester units.

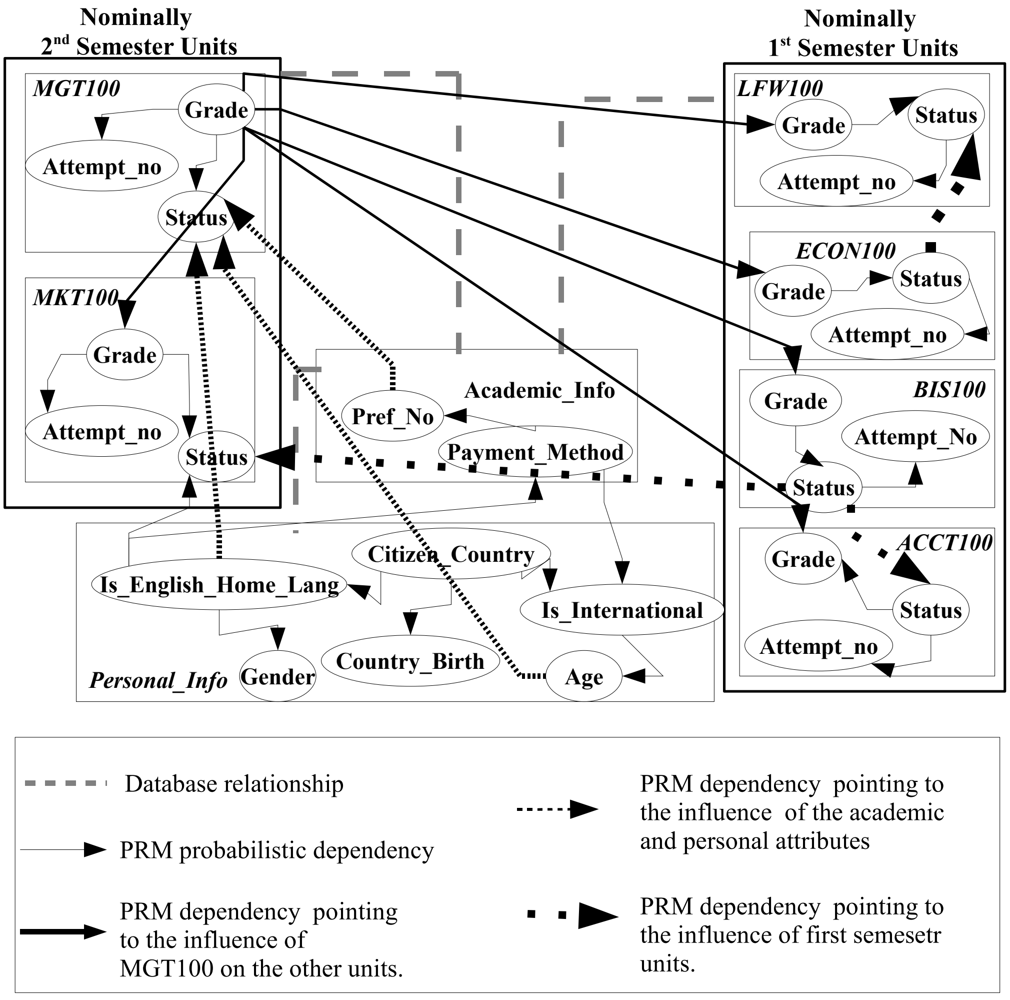


Figure 3.5: The PRM structure learned from the BCom dataset.

3.2.1.2 Bachelor of Commerce (BCom)

The PRM model learned from the BCom dataset is shown in Fig. 3.5 and a summary of the main dependencies is presented in Table 3.4. The dependency model shows the MGT100 grade as the parent of all other units grades. The learned parameters of this model showed that the better a student performs in MGT100, the higher the probability of achieving higher grades in all other first year units.

To recap from the BCom curriculum shown in Table 3.2, the first year of BCom does not include any prerequisites (i.e. the units can be studied in any semester). Hence, the dataset of BCom consists of all the first year units regardless of which semester the units were studied. This could explain why MGT100, as a second semester unit, does not depend

Attribute	Learned parent(s)
LFW100.grade	MGT100.grade
LFW100.status	ECON100.status
ECON100.grade	MGT100.grade
BIS100.grade	MGT100.grade
ACCT100.grade	MGT100.grade
MKT100.grade	MGT100.grade
MKT100.status	BIS100.status
ACCT100.status	BIS100.status
MGT100.status	Academic_Info.Pref_No
Personal_Info.Is_English_Home_Language	Personal_Info.Citizen_Country
Personal_Info.Is_International	Personal_Info.Citizen_Country

Table 3.4: A summary of the main PRM dependencies learned from the BCom dataset.

on first semester performance. The dependence of first year units on the `MGT100` could indicate the potential of setting `MGT100` as a prerequisite for the other first year units and hence the inclusion as a first semester unit.

Moreover, the model shows correlations between the `status` of `BIS100` and the status attributes of `ACCT100` in semester I and `MKT100` in semester II. In addition, the model also shows correlations between the status attributes of `ECON100` and `LFW100`

These dependencies found between the BCom units could suggest placing some units as prerequisites for others. For instance, as BCom curriculum does not specify prerequisites and the `MGT100` unit appears to be the main parent of all the other units, this could indicate that `MGT100` can be a prerequisite for the other BCom units and need to be placed as a first semester unit.

Moreover, the PRM model has also identified other units in semester II which depend on `ECON100` and `BIS100`, which could also point to using these two units as prerequisites for their identified semester II children-units and hence these two units should remain in the first semester with `MGT100`. Table 3.5 shows the suggested BCom plan by the learned PRM model.

In order to check the influence of the PRM suggested prerequisites, the BCom dataset was analyzed. Fig. 3.6 shows a comparison between the performance of two sets of students, the first set of students were those who followed the normal BCom curriculum and the second set of students were those who had followed the implicit prerequisites indicated by the learned PRM models, as shown in Table 3.5. In comparison to the normal prerequisites, it was found that with the suggested prerequisites, the overall percentage of

Semester	Unit Code	Prerequisite
I	BIS100	-
	ECON100	-
	MGT100	-
II	ACCT100	MGT100, BIS100
	LFW100	MGT100, ECON100
	MKT100	MGT100, BIS100

Table 3.5: The PRM suggested prerequisites for BCom.

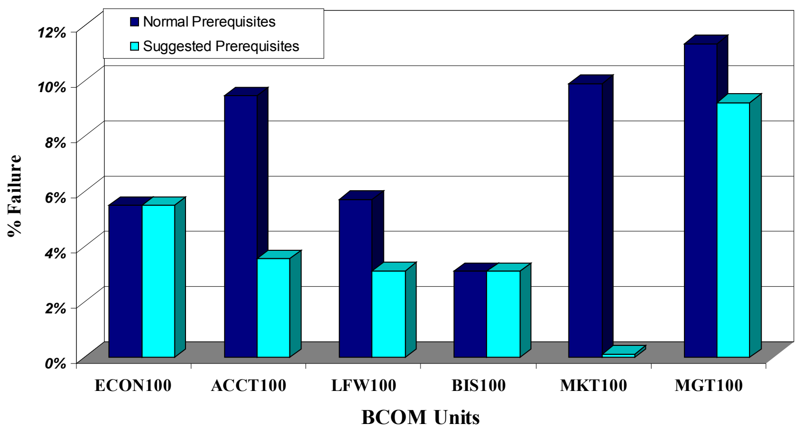


Figure 3.6: The comparison of students performance in semester II obtained by following the normal BCom prerequisites and the PRM suggested prerequisites.

failure of the BCom units is reduced. However, in both cases, the failure percentage was similar for `BIS100` and `ECON100`, as these units remain the same in both cases. Hence, this confirms the finding from the PRM model suggesting the reordering of the units and placing `BIS100,ECON100` and `MGT100` as first semester units and prerequisites for the second semester units.

3.2.1.3 Similarities between the degrees

In both the BCS and BCom learned models, the PRM successfully identified many of the dependencies that would be considered reasonable. For instance, in the `Academic_Info` table, the attributes `is_English_home_language` and `is_international` depend on the student's country of origin, and the unit status and number of repetitions depend on the unit grades. Moreover, the findings from the PRM models agree with Wetzel *et al.* (1999) in that the most significant factors affecting student performance are the academic factors

Preference No.	Attempt No.		
	First	Second	Third
First	0.913	1.00	0.00
Second	0.087	0.00	1.00

Table 3.6: The learned conditional probabilities of ST151 Attempt_no.

with personal and financial factors being of less importance.

Both BCS and BCom models pointed to the effect of the student's preference of study on the student's performance in first year units. In the BCS PRM model, the grades of the main units in computer science: `ST151` and `FCS151`, taken in semester I, depend probabilistically on the student preference of course to study, shown earlier in Fig. 3.3 by the thick dotted links directed from `Preference_No` to the grades of `ST151` and `FCS151`. Table 3.6 shows the learned parameters of `ST151`, which shows that the student will pass the unit at the first attempt with probability ≥ 0.9 if Computer Science was the student's first or second preference for a course, while this probability drops to zero when Computer Science is the third preference.

In BCom, the model also indicated that the results of `MGT100`, the unit affecting all other units in BCom, is controlled by three personal factors: the student's age, preference of study and if English is the home language, as shown in Fig. 3.5 by the dashed links directed to the `MGT100` grade. The probability distribution shows that students with age category (16-19), with English being the home language and having Commerce as first or second preference, have a higher probability of achieving better results in `MGT100`.

3.2.1.4 The mining outcomes of PRMs on the students database

PRMs have successfully pointed to a number of findings for the BCS and BCom degrees that could increase the success rates of first year students. For example, both BCS and BCom models point to the importance of the student's preference of study on the student's performance, which suggests focusing on the student's preference of study in the enrollment.

In the BCS model, the dependencies between the units suggest some additional prerequisites to those defined in the curriculum in order to increase the success rates of first year students. For example, the effects on the semester II units by Maths and English studied in semester I could indicate allocating more resources to the Maths and English units, as well as considering these units as prerequisites for second semester units in addition to the curriculum-defined prerequisites.

As the BCS study dataset was from an English-based university, this could explain the significant effect of the English unit on the students' performance. Considering the effect of Maths, this finding agrees with that of Byrne and Lyons (2001), who applied a regression analysis to study the factors affecting computing students, and found that Maths and Science results are critical factors for a computing student's performance. In comparison to regression analysis, PRMs learn the probabilistic interactions between the domain variables without any prior knowledge, whereas regression analysis requires defining which variable to investigate. Considering the description of the second semester units, in particular the `FCS152` unit, which covers the theoretical aspect of computer science including set theory, relations, logic, induction etc., indicates that this unit needs the techniques learned in Maths. In addition, this could recommend considering Maths and English, when selecting students for the Computer Science degree.

Referring to the BCS curriculum prerequisites, for example: the `ST151` unit is a prerequisite for `ST152` because `ST151` provides an introduction to Unix and Java programming that is required for studying the data structure concepts in `ST152`. However, the PRM model for BCS did not draw any relationships between the curriculum perquisites, but has strongly identified both English and Maths as the implicit prerequisites for semester II units. This could raise the question: are `ST151` and `FCS151` indeed prerequisites for `ST152` and `FCS152`, respectively, as defined by the BCS curriculum? However, this question could not be investigated in this study, as the BCS dataset is restricted with the set of students who passed the curriculum prerequisites and then enrolled into semester II units.

For the BCom model, the impact of the dependencies could suggest some changes in the BCom curriculum in order to improve the student's performance in the first year. For instance, the PRM model suggests setting `MGT100` as a prerequisite for the second semester units, as `MGT100` performance influences the performance of the other units. As other dependencies were also identified in the BCom, presented earlier, this could suggest moving `MGT100` to semester I with `BIS100` and `ECON100` and moving `ACCT100` and `LFW100` to semester II with `MKT100`, as shown in Table 3.5.

In conclusion, the learned PRM models have pointed to a number of significant dependencies that could help the University to improve the success rates in first-year program. Particularly the influence of the suggested prerequisites by the PRM models, which could help the University to refine the degree curriculum by identifying the implicit performance prerequisites of each unit and consequently help the students to perform better in subsequent units.

3.3 Dataset II: Advanced undergraduate relational database

3.3.1 Dataset description

This study dataset is also derived from the student relational database used at Curtin University focusing on first year students of the Bachelor of Computer Science (BCS) and the Bachelor of Commerce (BCom). In this dataset a single database is extracted to include all the information about BCS and BCom students for the years 2001-2006. Moreover, this dataset not only includes the data of the students in the University, but also includes related data of high school performance.

This dataset is organized into a set of tables and attributes organized as follows:

- The `Student` table, which includes the student personal attributes:
  - Age: discretized into groups of {16-19, 20-29, 30-40},
  - Gender: {Male, Female},
  - Suburb: which takes values from a predefined set of suburbs.
  - Student_ID: Student ID, which represent the primary key of this table.
- The `Academic_Info` table holding the academic attributes:
  - Course: {BCS, BCom}, representing the student's course: {Bachelor of Computer Science, Bachelor of Commerce}, respectively,
  - FT/PT: representing the study mode: {Full Time, Part Time}, respectively,
  - Preference_No: {1, 2, 3}, describing the student's preference of study, where Preference_No = 1 indicates that the corresponding degree is the student's first preference degree to study and so on.
  - Student_ID: Student ID, which represent the primary key of this table in combination with the course.
- The `School` table holding the high school attributes:
  - SchoolCode: which takes values on a predefined set of schools, and represents the primary key in this table.

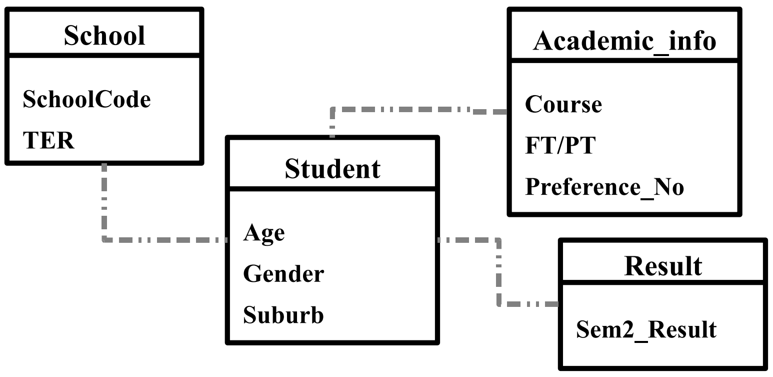


Figure 3.7: The relational database schema of dataset II.

  - TER: (Tertiary Entrance Rank) taking values of {6, 7, 8, 9}, representing the categories of {60-69, 70-79, 80-89, 90-100}.

- The `Result` table that represents the performance in first year at university:
  - Sem2_Result: representing the result of semester II in the first year, which takes values of {Terminated, Conditional, Good Standing}.
  - Result_ID: result ID, which represent the primary key of this table.

The `Student` table is the main table in this dataset, it is associated with a key attribute: `Student_ID`, which links the `Student` record of a specific student with the corresponding records of the other tables. Fig. 3.7 shows the relational database schema of this dataset. Although this dataset represents an advanced dataset of the one described in the previous section, yet small numbers also occur in some instances as in the previous dataset.

3.3.2 Applying PRMs to dataset II

The structure of the learned PRM of dataset II is shown in Fig. 3.8. The structure shows dependencies between the same tables and between different tables. The learned PRM model has identified a number of dependencies that could be considered reasonable and correct. For example, the dependence of a student's study mode (Full Time or Part Time) on the student's age, as usually full time students are younger than part time students who would probably be working and thus studying part time. Moreover, the dependence of the student's course of study on the student's preference number, as it is likely that students would be enrolled in the course of their choice.

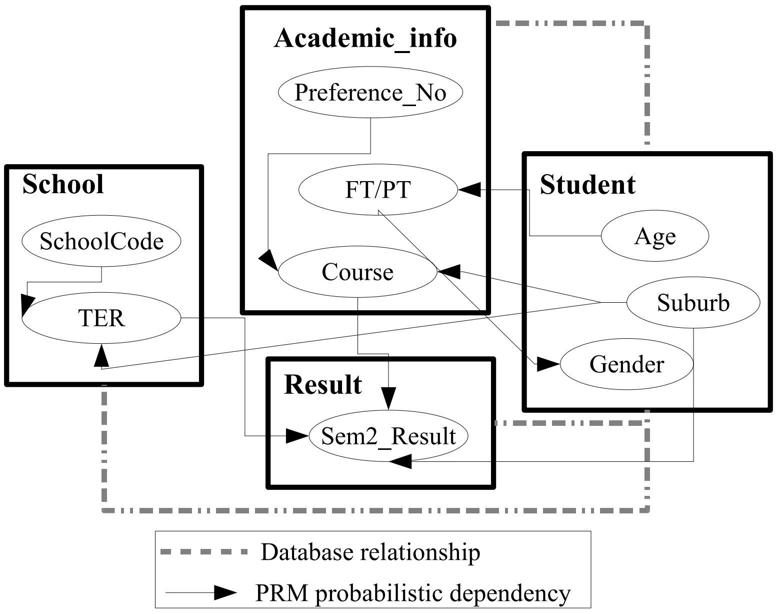


Figure 3.8: The PRM structure learned from dataset II.

In addition, there were other dependencies that are interesting and need further analysis. For instance, the dependence of the TER on the student's school, which indicates that the student's score is strongly influenced by which school the student attended. The most interesting dependency was the dependence of student results in semester II of university on three factors from different related tables, which are: student's course of study, TER and suburb.

It was found from the conditional probabilities that for certain suburbs, students studying BCom are more likely to obtain 'Conditional' or 'Terminated' results in semester II even if they have scored high results (TER) in their high school. On the other hand, it was found that students from other suburbs studying BCom perform very well in semester II regardless of their TER in high school.

The PRM model learned from dataset II was of great interest to the domain experts, who provided the database. Domain experts were mostly interested in the dependencies discovered by the PRM about the influences on student performance in semester II by the course of study, suburb and TER, as these outcomes can help the University to guide resource allocation to improve student performance. For example, this could help the University to investigate how well a student would perform in University given the student comes from a specific suburb with a certain TER. This investigation could potentially help the University to identify students at risk in the early stages and provide the required assistance. Moreover, this finding could help the University to manage resource allocation, thus more resources could be invested for students who need more help and less resources provided for others.

3.3.3 Spatial visualization of the PRM outcomes

The graphical representation of the PRM offers an insight into the main dependencies in the domain and how the domain variables interact with each other in different ways. Once these dependencies are discovered, further analysis is required to gain a greater understanding about these dependencies and how can they be employed to achieve the greatest efficiency. Moreover, this analysis is essential for the domain users who are usually not expert on PRMs and hence not familiar with the notations of the PRM models.

Further analysis of the PRM outcomes could be achieved in various forms, such as: consulting the domain experts about the dependencies or obtaining more statistics from the dataset to better understand the dependencies. Another method that can be used to facilitate getting the picture of the dependencies is via visualization, due to the significance of visualization as an effective and easy way for delivering the ideas behind the data.

This section explores visualizing the outcomes of PRM models by providing a visualization tool that can be effectively used by the users within the domain. This visualization tool is constructed in a way that it presents the results of the PRM dependencies in a simple, easily explored and interactive representation.

The primary focus of this visualization in this chapter is related to visualizing the outcomes revealed by the PRM on dataset II. In particular, visualizing the influence of the TER, suburb and course factors on student results in semester II in University, as these dependencies are of significant interest to the domain experts.

The aim of visualizing this particular outcome is to provide the University with a graphical tool that be can be used easily to obtain direct feedback about student performance given the different factors discovered by PRMs and especially by the ‘Suburb’ factor. This visualization representation should spatially plot the students’ performances based on their suburbs and should be flexible to allow the users to navigate between the suburbs to display the different results. Plotting the performance spatially can help the University to recognize any potential correlation between the suburbs, especially of the suburbs closer to the University.

For this visualization task, the Keyhole Markup Language (KML) (2009) has been employed. KML is an XML-based language developed for expressing geographical data on two-dimensional maps or three-dimensional earth browsers, such as Google Earth and Google Maps. KML provides a tool that can be used to explore the data in new ways, by pinpointing different locations, adding images, polygons or even 3D models.

The KML file simply consists of a set of features, such as: place-marks, polygons, 3D models, images, etc. Each feature in KML is specified by a longitude and a latitude to specify the location of the feature. Other features may require additional information. Once the KML files are created with the relevant place-marks, models and images, they can be viewed in Google Earth or Google Maps and easily navigated.

KML has been used for the visualization task in this section because it can be easily used to mark the different suburbs in dataset II on Google Earth and then add the corresponding images or 3D models representing the students' performances. Moreover, the KML files can access web pages to obtain information and can be easily distributed in very small files called KMZ files. The KMZ file represents a zipped file of the relevant KML file. The KMZ file includes a single KML file and any other images, icons or models used in the KML.

In this chapter, Google Earth (GE) has been chosen to display the geographical representation of the KML files, due to the power of GE as a desktop program that could display different views of the Earth. The views of GE can range from satellite images and maps to 3D model buildings. Moreover, GE offers the ability to view very rich geographical scenes of the Earth from different angles, in which users can save the interested places and share them with other users. More importantly, and for the particular task of the visualization in this chapter, representing the student performances as 3D models can be easily achieved in GE using KML files.

Therefore, a KML file was implemented to display the geographical representation of the students results for each of the BCom and BCS courses. Initially, for each of the degrees, a KML file was created such that it includes a set of place marks, representing the suburbs, and image features, representing student performance as histograms. By loading this KML file into GE, the suburbs are shown as yellow pins, as shown in Fig. 3.9. The student performances can be viewed as an image in a separate window by clicking on the suburb pin, as shown in Fig. 3.10.

The BCS student performances in Fig. 3.10 are represented as a histogram showing the 'Probability of passing semester II' for each of the 'TER' scores from high schools for each

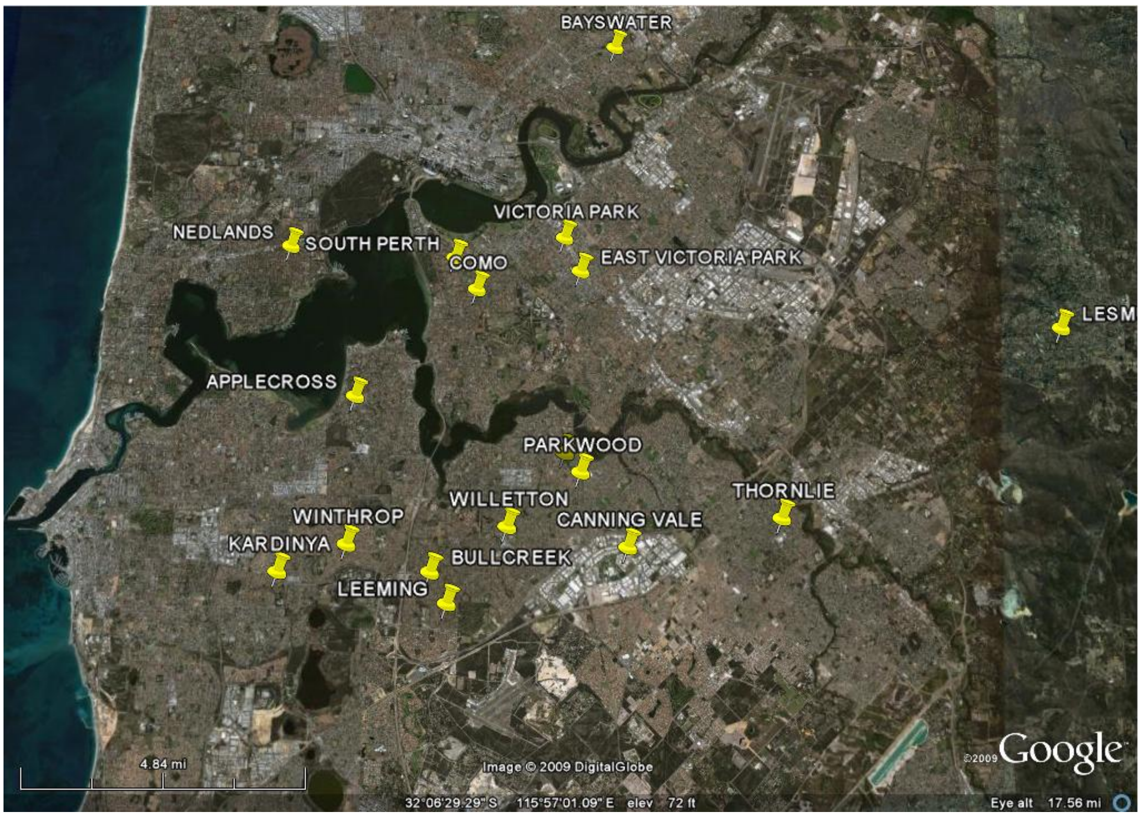


Figure 3.9: Snapshot of the geographical representation for using place-marks in Google Earth.

suburb. For example in Fig. 3.10, if the BCS student is from Parkwood suburb with a TER of '70's' or '90's', then the probability of obtaining 'Good Standing' in semester II is 100%, however the probability of 'Good Standing' drops to 80% for a TER of '80's' with a probability of 20% of obtaining 'Conditional' result. The high probability of 'Good Standing' for TER of '70's' or '90's' is due to the small number of samples in this category.

Although the representations shown in Fig. 3.9 and 3.10 show geographically student performance based on the suburbs, this representation lacks the ability of viewing the performances of more than one suburb at a time to get immediate feedback about the major differences between the suburbs. Therefore, an advanced representation was created using 3D models by using Google SketchUp (2009). Fig. 3.11 and 3.12 shows snapshots of the implementation of KML files with 3D models to show the results of semester II for each suburb for the different TER results of high schools.

Using the KML file with 3D models in GE provides a flexible tool to geographically visualize student performance of different suburbs. The student performances are plotted as a 3D histogram model. Each 3D model is a stacked bar chart for each of the suburbs

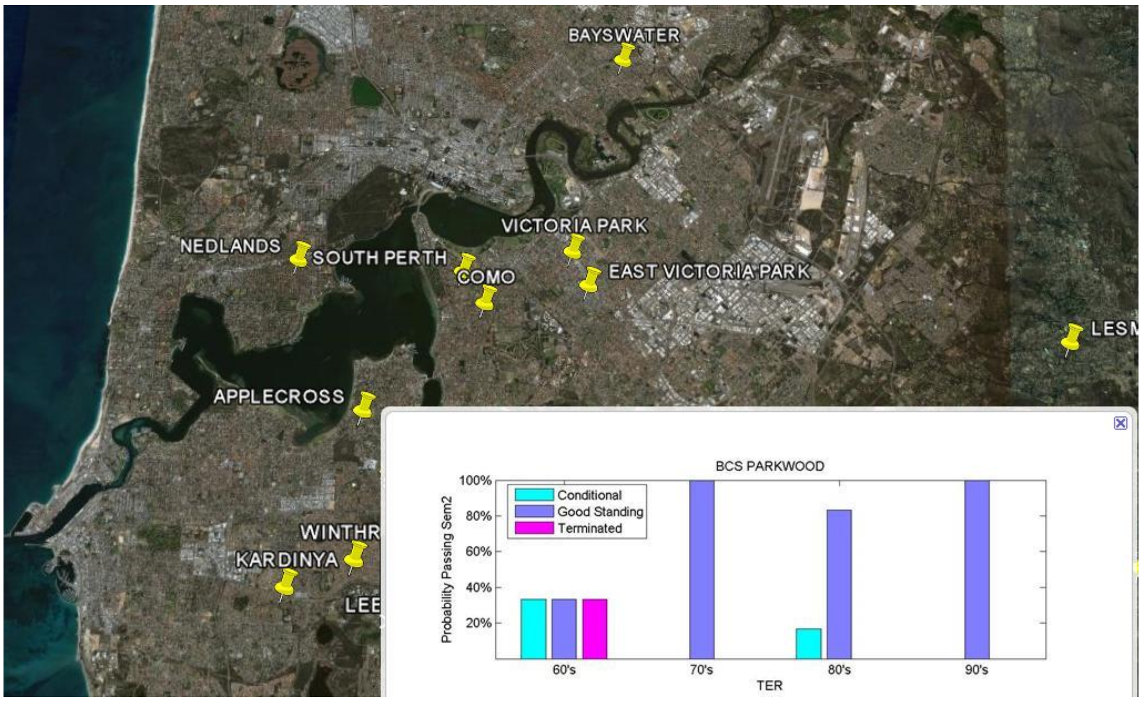


Figure 3.10: Displaying the students' performances for a specific suburb.

of each course showing semester II results based on TER results. The navigation and the preview of the 3D histograms can be easily achieved through GE navigation tools.

Domain users can answer several queries using this visualization task, for instance: what are the best/worse suburbs in terms of student performance? Whether the near suburbs have similar/different student performance? Or what is the difference, in terms of student performance, between the suburbs nearby the University compared with those far away?

Answering such queries can help the University to manage resource allocation, thus more resources could be invested where it is needed and less resources for others. Examples of such queries and answers are shown in Fig. 3.13 and Fig 3.14. In these figures, although the histograms show the performances of BCS students from nearby suburbs, the performances are not similar and vary to some extent between the suburbs. On the other hand, the histogram shown in Fig. 3.15 shows relatively similar performances of BCom students from three nearby suburbs, especially for the TER of values: {60's, 70's, and 90's}.

Therefore, by this visualization tool, users can simply view the outcomes revealed by the PRM model in an interactive way and navigate through the conditional probabilities of the PRM model as 3D histogram charts plotted geographically on GE. It is hoped that this visual representation would help the University to gain a better understanding about

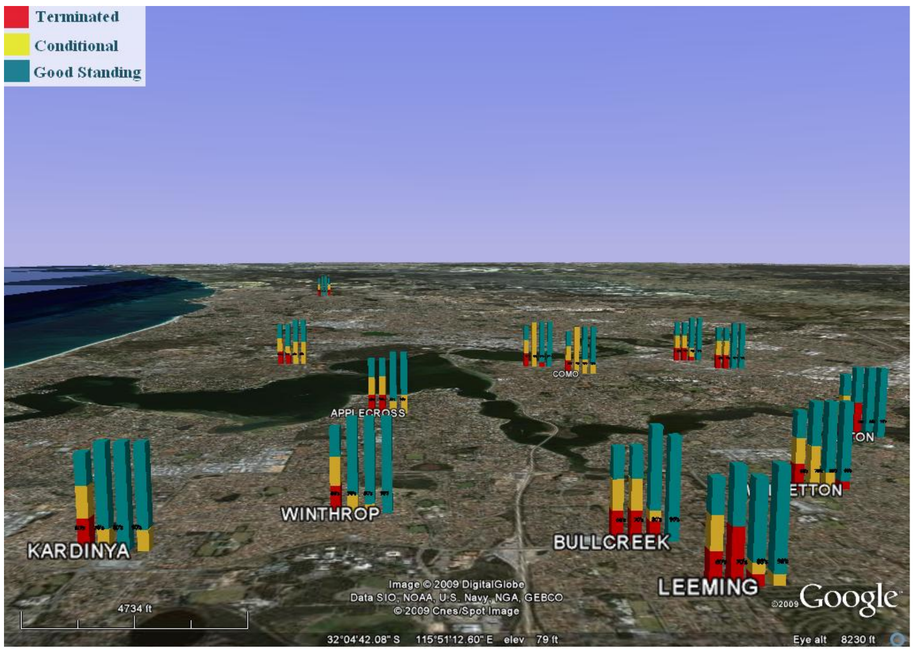


Figure 3.11: Snapshot of the geographical representation for performances of BCS students.

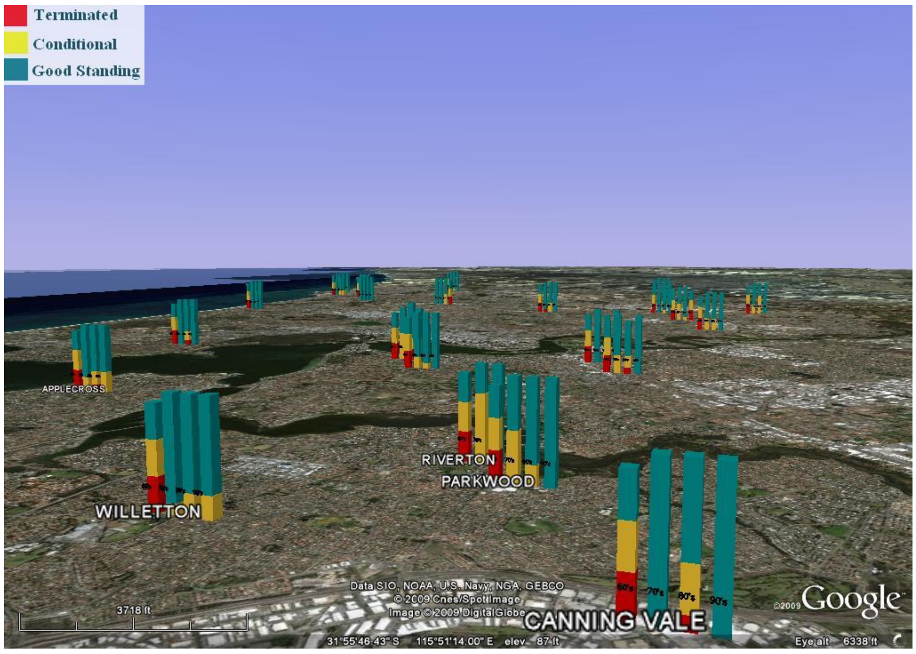


Figure 3.12: Snapshot of the geographical representation for performances of BCom students.

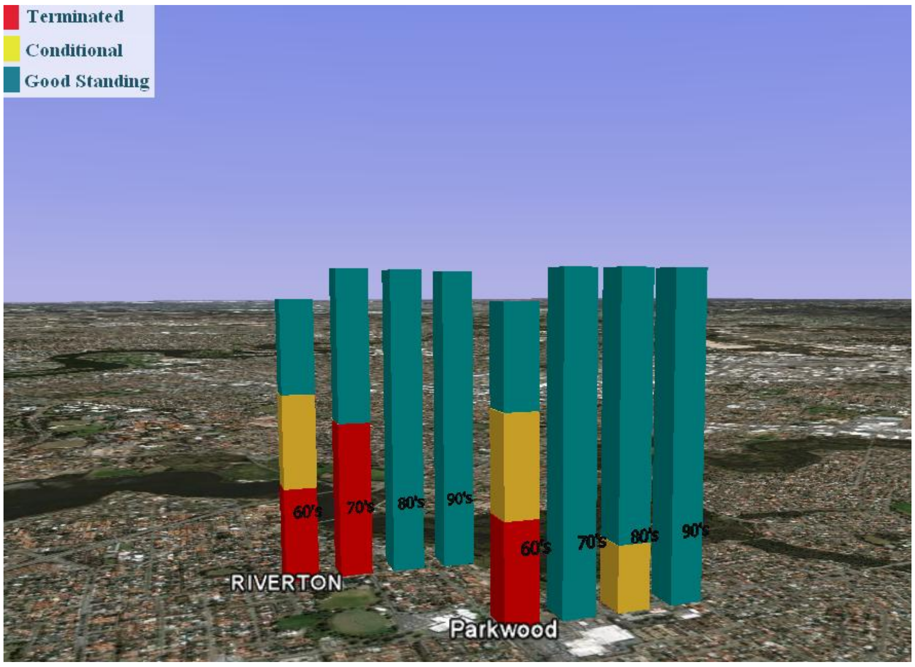


Figure 3.13: The performances of BCS students of two nearby suburbs.

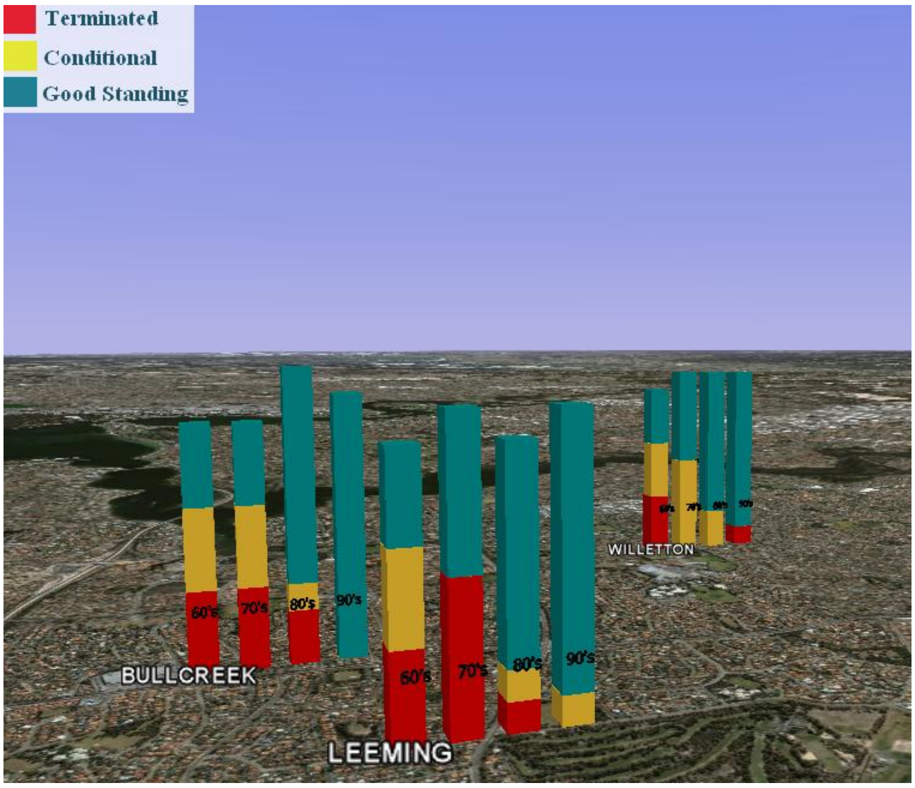


Figure 3.14: The performances of BCS students of three nearby suburbs.

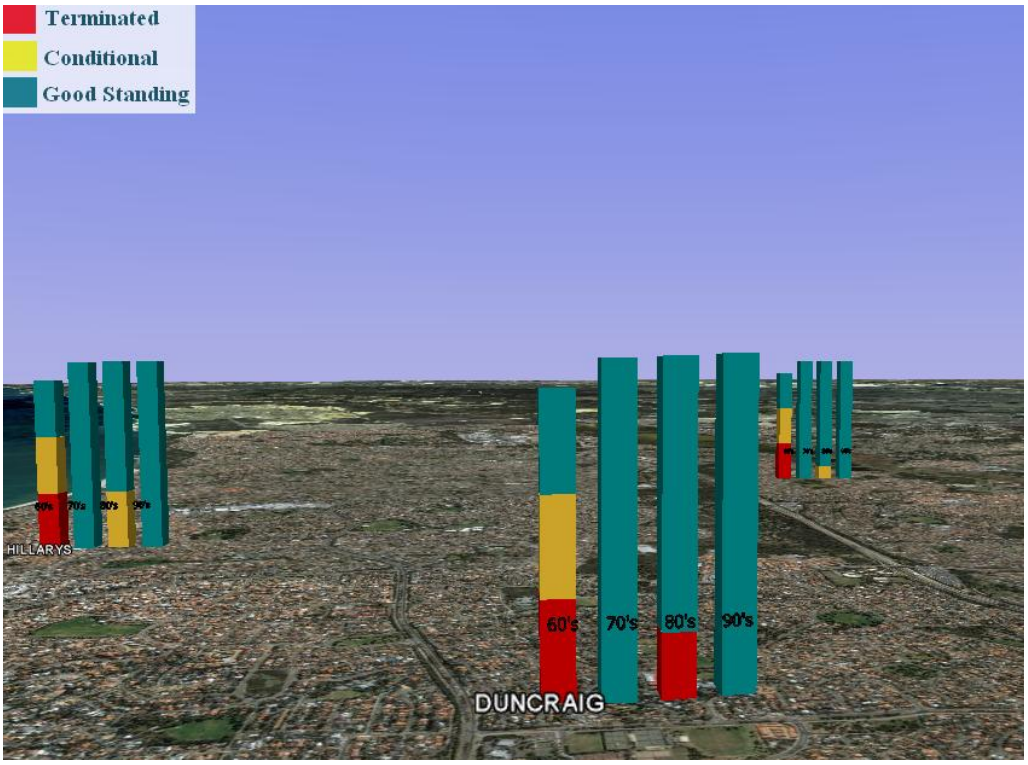


Figure 3.15: The performances of BCom students of three nearby suburbs.

the dependencies discovered by PRMs and note the correlations across suburbs.

3.4 Conclusion

In this chapter, PRMs were applied to two real-world University datasets. The goal of these applications was to explore the effectiveness of PRMs in capturing the most important dependencies in the University undergraduate domain, and in particular for Computing and Commerce undergraduate students. The models learned form the undergraduate datasets pointed to a number of recommendations to improve the success rates of first year undergraduate students. In the Computing and Commerce datasets, PRMs have successfully identified implicit prerequisites within the first year of the degree programs. In the second undergraduate dataset, PRMs have pointed to interesting factors affecting student results in semester II in University. These factors were of great interest to the domain experts and a visualization tool was constructed to illustrate the spatial significance of the dependencies indicated by PRMs. This chapter explored the concepts of PRMs on a University dataset to gain more understanding about the domain. Although PRMs can produce indications of causes and effects, a major concern in statistical learning is the ability to use the learned model to make reliable predictions about new instances. The next chapter discusses the effectiveness of PRMs in obtaining predictable models from real-world relational data.

Chapter 4

Mining Imbalanced Relational Data

The previous chapter presented two mining applications of PRMs in real-world relational domains. Each of the PRM applications revealed a probabilistic model describing the significant dependencies in the relational domain. An important task in data mining and statistical modelling is using the learned models for classification. This chapter investigates the effectiveness of PRMs in obtaining accurate classification models from real-world data. In particular, this chapter explores the effectiveness of PRMs in learning from imbalanced data that is common in many real-world applications and which has reported significant deterioration of the performance of traditional classification techniques. This chapter addresses the learning from relational imbalanced data by enriching PRMs with ensemble learning. The performance of the proposed method is shown on a number of real-world relational imbalanced datasets, including a university student dataset to identify students at risk.

4.1 Introduction

Several pattern classification techniques have been proposed for different types of classifications problems, such as decision trees (Quinlan, 1993), neural networks (Mitchell, 1997), Bayesian Networks (Pearl, 1988) and support vector machines (Cortes and Vapnik, 1995). Most of these techniques have been designed on the assumption of learning from single flat files with relatively balanced data distribution. However, in real-world domains, the data is often presented as relational database systems with imbalanced data distribution for which one class of data has a very large number of samples while the other classes are represented by only a few samples.

The imbalanced class problem is an important issue in pattern classification, as it can result in a significant deterioration in the performance achieved by most well-known learning techniques, particularly for detecting the minority samples (Japkowicz and Stephen, 2002). This is due to the fact that these methods get biased towards the majority samples, and hence classify most of the minority samples to be of the majority class. As a result, although the learned model can achieve high classification results for detecting the majority samples, it significantly fails to correctly classifying the minority samples.

The imbalanced class problem is crucial in many real-world applications as it is more important for the users to accurately predict the rare cases rather than those of the majority, such as in detecting fraudulent telephone calls (Fawcett and Provost, 1997), the detection of oil spills in satellite radar images (Kubat *et al.*, 1998) and identifying unreliable telecommunication customers (Ezawa *et al.*, 1996). The imbalanced class problem is even more complicated in the case of relational data as most learning techniques are proposed for learning from flat data representations and hence are inadequate for handling the relational learning.

There have been several attempts to solve the imbalanced class problem on flat data representation (Batista *et al.*, 2004; Nugroho, 2000; Ezawa *et al.*, 1996; Pazzani *et al.*, 1994), as discussed in Chapter 2. However, these methods have been proposed on the assumption of using non-relational data. In contrast, although a number of techniques have been proposed for relational learning, such as FOIL (Quinlan and Cameron-Jones, 1993), CrossMine (Yin *et al.*, 2004) and TILDE (Blockeel and Raedt, 1998), they assume learning from balanced data.

Therefore, this chapter presents an extension of PRMs, named PRMs-IM, to learn from imbalanced relational data. The goal of this technique is to obtain high overall performance in learning from imbalanced relational data, and particularly for correctly classifying the minority samples in relational domains.

Although PRMs have been successfully applied in several domains for relational learning and have shown superior results (see the previous chapter on learning from the student relational data), PRMs performance can be hindered by the imbalanced class problem. Ezawa *et al.* (1996) argues that in general, research on learning Bayesian Networks focuses on obtaining models that best represent the data. However, these models may perform very poorly when applied to a specific goal, such as the classification task. Cowell *et al.* (1993) also shows that although the learned model may perfectly represents the relationships in the data, it is not necessarily the best model for classification.

As in Bayesian Networks, PRMs which are the extended version of Bayesian Networks for relational learning, can successfully discover the correct relationships of the relational domains. However, in real classification problems, the performance could be degraded as a result of not being modelled for the specific goal of classification.

Therefore, in order to solve the imbalanced class problem in relational data, PRMs-IM proposes an approach consisting of the construction of an ensemble of PRM models. Thus, instead of training one model on imbalanced data, a set of PRM models are trained on balanced relational subsets extracted from the training data. Then, the results of the learned PRM models are combined to obtain the final classification result.

PRMs-IM can be viewed as a two-process approach that employs PRMs for relational learning and utilizes ensemble learning to handle the imbalanced class problem. PRMs have been employed for the relational learning process due to its effectiveness in learning from relational data, as presented in the previous chapter. For the other process, ensemble learning is implemented to train the classification models on balanced subsets of the data. Ensemble learning is implemented to benefit from its improved performance that has been reported in several studies, which not only improve the training time of the algorithm but also the robustness and performance of the classification (Kamel and Wanas, 2003).

In comparison to the methods reviewed in Chapter 2 for learning from imbalanced relational data, PRMs-IM is the only method based on a relational learning technique, except for the work by Sen and Getoor (2008) which employs conditional Markov networks (Taskar *et al.*, 2002) for learning from imbalanced structured data using cost-sensitive learning. However, using cost-sensitive learning includes a major challenge of setting the proper cost for each misclassification. Furthermore, PRMs-IM makes use of the database in its relational format without reverting to intermediate steps of flat views of the relational database as proposed by the other relational approaches discussed in Section 2.4.1.

In this chapter, PRMs-IM is evaluated on a number of real-world domains with imbalanced data, including the student dataset discussed in the previous chapter. The results are shown in terms of the Receiver Operating Characteristics (ROC) curves and the Area Under a ROC Curve (AUC) values, which are usually used for measuring imbalanced learning algorithms (Fawcett, 2006). The evaluation of PRMs-IM has shown promising results as a relational learning technique for imbalanced data and in particular for correctly classifying the minor samples.

This chapter is organized as follows: Section 4.2 introduces ensemble learning. Next, in Section 4.3 the PRMs-IM is presented, followed by the results of evaluating PRMs-IM on real-world domains in Section 4.4. Finally, Section 4.5 concludes the chapter.

4.2 Ensemble learning

Building a classification system usually involves the process of evaluating several classification techniques for the specific classification problem at hand. Then, the classifier with the best performance is used as the chosen classification system. However, although one classifier can have the best overall recognition performance, the samples misclassified by the best classifier may be correctly classified by the other classifiers. This is because the samples misclassified by the different classifiers are not necessarily the same (Kittler *et al.*, 1998).

Therefore, the overall performance accuracy can be improved through integrating several classifiers instead of using just one single classifier. This method of using multiple classifiers for classification is often referred to as ensemble learning (Opitz and Maclin, 1999; Dietterich, 2000), combining multiple classifiers (Kittler *et al.*, 1998), classifier fusion (Ma *et al.*, 2008), and hybrid methods (Barbosa and Freire, 2007). Throughout this thesis, the term 'ensemble method' will be used to refer to the concept of using multiple classifiers for solving classification problems.

The basic idea behind ensemble learning relies on the concept that using multiple classifiers could lead to classification results that are better than any of the individual classifiers. This improvement in the classification results is attained as misclassification errors committed by each classifier are often in different regions of the input space (Kittler *et al.*, 1998). Thus, it is hoped that the errors made by one classifier would be corrected by the other classifiers. In other words, ensemble learning utilizes the different components that would likely disagree about their predictions. Note, there is little to be gained by using identical classifiers with the same outputs.

Ensemble techniques have been widely employed in several domains as an effective and practical solution for different complex classification problems (Czyz *et al.*, 2004; Barbosa and Freire, 2007; Mashao and Skosan, 2006; Kedarisetti *et al.*, 2006). Different machine learning techniques have been proposed to perform this type of learning, and it has been shown in several studies that these methods outperform a single best classifier (Opitz and Maclin, 1999; Dietterich, 2000). Popular methods using this approach are bagging (Breiman, 1996), boosting (Freund and Schapire, 1996) and stacking (Savický and Fürnkranz, 2003).

The ensemble method can be simply defined as a learning algorithm that constructs a set of base classifiers and then combines the classifier outputs to classify new samples. Therefore, ensemble learning consists of two components: constructing individual classifiers and then combining the outputs of the classifiers.

Several methods have been proposed for constructing ensembles by following different strategies, including:

- **Varying the initial parameters and weights of the classifiers**: The set of ensemble classifiers can be constructed such that each classifier is trained on the same training data but with different initial weights and parameters that can be obtained randomly. For example the back-propagation algorithm for neural networks randomly initializes the weights of the network. Applying this algorithm on the same training data but with different initial weights would result in different classifiers (Kolen and Pollack, 1990).
- **Manipulating the training data**: In this approach, each classifier is trained on different datasets obtained from the original training data. This alteration of training data can be achieved in several ways, including: sampling, disjoint data subsets, using different data sources, or a combination of these methods. Well known examples of this method are the bagging and boosting methods (Opitz and Maclin, 1999). Although the former approach of varying the initial parameters is the most common approach of creating ensembles, this approach of varying the training data is more effective (Dietterich, 2000).
- **Varying the classification algorithm**: The ensemble can be constructed such that each classifier uses the same training data but employs a different classification algorithm, such as: Bayesian Network, Support Vector Machine, decision tree and so on, as in the method proposed by Barbosa and Freire (2007).
- **Altering the input features**: In a similar approach to manipulating the training data, this approach trains each classifier with a different set of data, but by using different input features for each classifier. This approach is popular when training data from different sources and hence each classifier is trained with a different set of features (Barbosa and Freire, 2007) or by training with different views of the same relational data (Guo and Viktor, 2007).
- **Altering the topology:** In this approach, the topology or structure of the classifiers can be varied while using the same training data. This approach is applicable where the ensemble members are employed as modular systems. For example, creating an ensemble of multiple neural networks, where each network is created with random topology (West *et al.*, 2005).

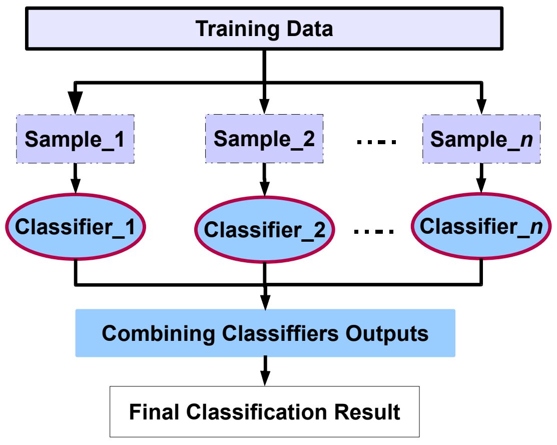


Figure 4.1: An illustration of the ensemble classification system.

- **Manipulating the output targets**: This approach is achieved by altering the class labels of the target attribute that is of interest to be classified. This approach is popular in multi-class ensemble learning by creating a set of classifiers, in which each classifier is trained for specific class labels, as in 'One-against-All' (Anand *et al.*, 1995) and 'One-against-One' approaches (Fürnkranz, 2002).

An illustration of ensemble learning based on altering the training data is shown in Fig. 4.1. In this figure each classifier is trained on a different subset of the original training data. Once the classifiers are constructed, a combining scheme is used to combine the outputs of all the base classifiers to obtain the final classification result of the ensemble.

After constructing the base classifiers of the ensemble, a combination method is required to combine the outputs of the classifiers to get the final classification result of the ensemble. The combination of the multiple classifiers depends on the type of outputs generated by the classifiers. The outputs of the ensemble classifiers can be generally classified into three groups (Duin, 2002; Tax *et al.*, 2000; Kuncheva, 2004):

- Class label: in which each classifier outputs a single label identifying the class of the test sample.
- Distance/Rank: in which each classifier outputs a list of the classes in a ranked order. The list is ordered such that the class at the top has the highest ranking. This type is practical for classification problems with a large number of class labels.
- Confidence/Posterior probability: in which each classifier outputs the confidence

of the probability of each class of the test sample. Thus, each classifier outputs a measure of the degree to which the test sample x is of each class i.

Although each of these output types provides essential information to classify the test samples, the confidence type provides the most information. Using the confidence type, the class label and the ranking of the classes can be easily derived. The class label can be obtained by selecting the class label with the highest confidence, and the rank output can be obtained by sorting the class labels in descending order according to their confidences.

To combine the outputs of the classifiers, the combining scheme depends on the output type of the classifiers. For the case of class labels outputs, non-linear approaches such as majority voting (Kuncheva, 2004) and weighted majority voting (Kuncheva, 2004; Witten and Frank, 2005) can be applied. Linear approaches such as the product, sum, average and weighted sum/average (Alkoot and Kittler, 1999; Kittler *et al.*, 1998) are applied for the case of confidence outputs. The rank outputs can be combined using more complex methods that also use nonlinear combination methods, including insertion and union methods (Ho *et al.*, 1994).

Throughout this thesis, only the methods working on the class label and confidence outputs will be required and hence considered. Consider a classification ensemble of N classifiers for classifying an attribute of K classes of a test sample x. For the case of class labels outputs, each classifier outputs a single class label of the test sample. For each class j of a test sample x, the output of a classifier C_i can be represented as follows:

$$\nabla_{ij}(x) = \begin{cases} 1 & \text{if } C_i \text{ classifies the sample } x \text{ to be of class } j \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

For the case of confidence outputs, each classifier outputs the confidence of each class of the test sample. For each class j of a test sample x, the output of a classifier C_i can be represented as follows:

$$C_{ij}(x) \triangleq \text{the confidence assigned by } C_i \text{ for the } j \text{ class of } x \tag{4.2}$$

A variety of techniques have been proposed for combining the classifiers using either the class labels or the confidence outputs. The simplest and probably the oldest approach is the majority voting technique (Kuncheva, 2004). In this approach the test sample is assigned to the class that has the largest number of votes from the independent classifiers. The popular ensemble method ‘bagging’ employs this approach for obtaining the final result of the ensemble. For the case of class label outputs, the final result is computed as:

$$F(x) = \operatorname*{argmax}_{j=1..K} \left[\sum_{i=1..N} \nabla_{ij}(x) \right] \tag{4.3}$$

In the case of confidence outputs, majority voting is often implemented as the ‘sum rule’. In this rule, the confidence outputs over all the classifiers are summed for each class and then the test sample is assigned to the class with the highest score, as follows:

$$F(x) = \operatorname*{argmax}_{j=1..K} \left[\sum_{i=1..N} C_{ij}(x) \right] \tag{4.4}$$

Another popular approach is the weighted voting strategy (Kuncheva, 2004). This approach is similar to the majority vote, but in this approach each classifier is associated with a weighting factor affecting the final classification result. The weight of each classifier is calculated in training as the accuracy performance of the classifier on the training data. Therefore, the outputs of the classifiers are weighted by using the corresponding classifier weight. For class label outputs, the weighted voting strategy is defined as follows:

$$F(x) = \operatorname*{argmax}_{j=1..K} \left[\sum_{i=1..N} W_i * \nabla_{ij}(x) \right] \tag{4.5}$$

where W_i is the weighting factor of classifier C_i.

Therefore, the test sample is assigned to the class with the highest weighted vote. This approach allows the expert individual classifiers to have a greater effect on the final classification result. This approach is used by the well-known boosting ensemble approach. Interestingly, the weighted voting could lead (in some situations) to better performance not only in comparison to the best single classifier but also compared to that obtained by majority voting (Kuncheva, 2004). In the case of confidence outputs, the weighted voting strategy is known as the ‘weighted sum rule’. In this type, the test sample is assigned to the class with the highest weighted score as follows:

$$F(x) = \operatorname*{argmax}_{j=1..K} \left[\sum_{i=1..N} W_i * C_{ij}(x) \right] \tag{4.6}$$

Various other techniques have been also developed for combining the classifiers using the confidence outputs (Duin, 2002; Alexandre *et al.*, 2001; Tax *et al.*, 2000; Kittler *et al.*, 1998):

- **The product rule:** In a similar approach to the sum rule, the product rule multiplies the confidence outputs over all the classifiers for each class instead of the summation. This rule is practical for highly reliable classifiers, as a low confidence output from a single classifier can greatly decrease the overall output regardless of the other confidence outputs. In this rule, the test sample is classified as :

$$F(x) = \underset{j=1..K}{\operatorname{argmax}} \left[\prod_{i=1..N} C_{ij}(x) \right] \tag{4.7}$$

- **The average rule:** This rule is similar to the sum rule, but it computes the average confidence output for each class over all the classifiers. This approach finds the class that has scored the best average score. Thus, the sample is classified as:

$$F(x) = \underset{j=1..K}{\operatorname{argmax}} \left[\frac{1}{N} \sum_{i=1..N} C_{ij}(x) \right] \tag{4.8}$$

- **The weighted average rule:** In a similar approach to the weighted voting strategy, the weighted approach of the average rule is employed. In this weighted approach, each classifier has a weighting factor that affects the final classification result. The test sample is classified as:

$$F(x) = \underset{j=1..K}{\operatorname{argmax}} \left[\frac{1}{N} \sum_{i=1..N} W_i * C_{ij}(x) \right] \tag{4.9}$$

  where W_i is the weighting factor of the classifier C_i.

- **The maximum rule:** In this approach, the classifier with the highest confidence is selected i.e. selecting the most confident classifier. The test sample is classified as:

$$F(x) = \underset{j=1..K}{\operatorname{argmax}} \left[\max_{i=1..N} C_{ij}(x) \right] \tag{4.10}$$

- **The max-min rule:** In this approach, the classifier with the minimum confidence is selected. The test sample is classified as:

$$F(x) = \underset{j=1..K}{\operatorname{argmax}} \left[\min_{i=1..N} C_{ij}(x) \right] \tag{4.11}$$

Several papers and experimental studies have shown that using any combination method often performs better than a single classifier (Alkoot and Kittler, 1999; Alexandre *et al.*, 2000; Kittler, 1998; Wang *et al.*, 2003). Alkoot and Kittler (1999) have experimentally evaluated some of the combination methods, such as: the minimum, maximum, average, median, majority voting and product rules, and have shown that the results of all of these approaches exceed that of a single classifier, especially the sum and median rules. The work done by Kittler *et al.* (1998) also shows that the sum approach often outperforms the other combination methods. Moreover, it has been shown that the weighted average and weighted sum are not only simple variations of the sum rule but also yield more robust results (Alexandre *et al.*, 2000; Kittler, 1998; Wang *et al.*, 2003). Kittler and Hojjatoleslami (1998) argue that the equally weighted methods perform better than a worst classifier. However, the weighted method always outperform the best classifier. Kittler and Hojjatoleslami (1998) also show that the use of weighted methods improves the overall performance as a result of allowing the expert classifiers to have the most effect on the final result and hence less effect by the poorly performing classifiers.

4.3 Methodology

This section presents the framework of PRMs-IM for learning from imbalanced relational data. In this framework, ensemble learning is employed to solve the imbalanced class problem, and the PRM learning is performed for the relational learning.

For ensemble learning, PRMs-IM constructs a set of ensemble components by adapting the scheme of varying the training data, which has shown improved performance when compared to other schemes (Dietterich, 2000). The main objective of adapting this scheme is to train each component of the ensemble on a balanced subset of minority and majority samples, and thus the performance of each component is not hindered by the imbalanced class problem.

This scheme is applied in PRMs-IM by considering the statistical distribution of the imbalanced data, in a similar approach to that reviewed by Nugroho (2000) and Barandela *et al.* (2003a), but for flat files. Considering the statistical distribution, PRMs-IM partitions the imbalanced data into a set of balanced subsets, where each balanced subset includes an equal number of the majority and minority samples.

For example, consider a relational database D that has a number of tables and attributes and includes an imbalanced two-class target attribute T, that is of interest for classification.

T consists of two classes: the majority class C_{mj} with n_{mj} samples and the minority class C_{mr} with n_{mr} samples. The size of n_{mj} is at least twice n_{mr}, which results in the imbalanced situation.

Using the statistical distribution of T, the imbalanced relational data D is partitioned into a set of balanced subsets $(D_1, D_2, D_3, ..., D_n)$. Each subset D_i is created from the original imbalanced data such that it includes relatively an equal number of samples from C_{mj} and C_{mr} classes. Each D_i includes all C_{mr} samples from D and an equal number of samples of C_{mj} selected randomly. In PRMs-IM, the selection of the samples of C_{mj} is performed randomly with replacement. The number of subsets n is determined as the difference in ratio between the number of samples of the majority and minority classes. Therefore, if the number of samples of C_{mj} is about k times those of C_{mr}, then k subsets are created.

Once the subsets are constructed, an individual PRM model M_i is learned from each subset D_i. The construction of balanced subsets of minority and majority samples is essential to ensure that the learned PRM models are not hindered by the imbalanced class problem. This is achieved as each subset includes all the minority samples and hence allows the learned model to have sufficient information about the minority class and thus not be biased towards the majority class.

However, as this learning is conducted with relational datasets, then the extracted balanced subsets must be well-defined representatives of the original data in terms of the relational structure but with a balanced data distribution. It is essential that each of the relational subsets has the same relational characteristics as that of the original data in order to learn an accurate PRM model that is qualified enough to represent the original data. Obtaining such qualified PRM representatives would result in obtaining well-learned models that are capable of providing confident classification results for new unseen instances for the final classification of the ensemble.

For this task of creating well-defined relational subsets, PRMs-IM first creates n new empty relational databases that have the same structure as the original relational data and such that the target table (the corresponding table of the target attribute T) of each database D_i is assigned with all the samples in C_{mr}. Then, an equal number of C_{mj} are allocated to the target table of each D_i. Finally, the other tables in each D_i are allocated with the records related to those in the target table. Following this procedure results in creating roughly n balanced relational datasets, in which each D_i have all the minority samples and an equal number of majority samples.

For combining PRMs-IM components, PRMs-IM adapts the popular method 'weighted voting' strategy. The weighted voting approach is used due to the characteristics of the training data of each component. In PRMs-IM, each component of the ensemble is trained on a different subset but use the same input features. Hence, a weighting factor can be calculated to measure the accuracy of the component on the training data within the same input space.

Moreover, as different data subsets are used with the same input features, it is important to allow the 'most confident' components to have the most effect on the final result and prevent the poorly performing classifiers from degrading the final result, as has been proved experimentally by Kittler and Hojjatoleslami (1998).

In PRMs-IM, the PRMs-IM components are used as classifiers by simply finding the class with the maximum likelihood. Therefore, the outputs of PRMs-IM components can be either simple class labels or probability estimations, and hence the weighted voting approach can be applied by using either the class labels or the confidence outputs, which is known as the 'weighted sum rule'. In PRMs-IM, the weighted voting strategy will be applied using the confidence outputs, as it has been shown that the weighted voting using the confidence outputs usually produces more robust results (Kittler, 1998; Wang *et al.*, 2003).

Therefore, for the weighted voting strategy in PRMs-IM, a weighting factor $W(M_i)$ is computed for each PRM model M_i. The weighting factor $W(M_i)$ is calculated as the average performance accuracy of evaluating M_i on the training subsets. PRMs-IM uses the Area Under Curve (AUC) value (Fawcett, 2006) to measure the performance of the learned models over the balanced subsets. As discussed in Chapter 2, the AUC measure is insensitive to the changes in class distribution and hence provides a robust metric for imbalanced algorithms.

The weighting factor $W(M_i)$ of each model M_i is calculated as the average AUC obtained from evaluating M_i over the training data subsets other than that of the model M_i. The weighting factor is defined as:

$$W(M_i) = \frac{1}{n-1} \sum_{s \in (D_1, D_2, D_3, ..., D_n)/D_i} AUC(M_i(s)). \tag{4.12}$$

where n is the number of balanced subsets, $AUC(M_i(s))$ is the AUC value obtained from evaluating the model M_i on the subset s, and the term $[(D_1, D_2, D_3, ..., D_n)/D_i]$ represents the set of all the training subsets except D_i that was used to build M_i.

PRMs-IM ensures that the poorly performing models have no effect on the final classification result by discarding the models with low AUC values. Therefore, only the models with AUC values greater than 0.5 are included in the final classification results. This threshold is chosen to ensure that the average performance of each model is better than that obtained by random allocation.

To classify a new test sample x, the sample is evaluated using each model M_i of PRMs-IM that have average AUC values greater than 0.5. Correspondingly, each model M_i outputs the probability scores: $M_i(x)_{mj}$ and $M_i(x)_{mr}$ of assigning the sample x to the majority and minority classes, respectively. Using the weighted voting strategy, the score of each class is computed as the summation of the weighted factors of the models. The test sample is then assigned to the class that has the highest weighted confidence, as follows:

$$F(x) = \underset{c \in (C_{mr}, C_{mj})}{\operatorname{argmax}} \left[\sum_{\forall M_i} W(M_i) * M_i(x)_c \right] \tag{4.13}$$

where $\forall M_i$ refers to the set of models with average AUC values greater than 0.5.

An illustration of the PRMs-IM approach is shown in Fig. 4.2. In summary and as shown in the figure, PRMs-IM first constructs an ensemble of PRM models. The ensemble is constructed by varying the training data such that each model is trained on balanced subsets from the original imbalanced data. The data subsets are constructed such that each subset has an equal number of minority and majority samples. An individual relational PRM model is learned from each subset. Then, the learned models with low AUC values are discarded and the remaining models are combined using the weighted voting strategy.

4.4 Experimental results

This section presents the evaluation results of applying PRMs-IM on real-world imbalanced relational datasets. In this chapter, the evaluation is performed on three relational datasets: the university dataset presented in Chapter 3, the financial database from the Discovery Challenge of PKDD99 (PKDD'99 Discovery Challenge, 2009) and the MovieLens database (GroupLens Research, 2009).

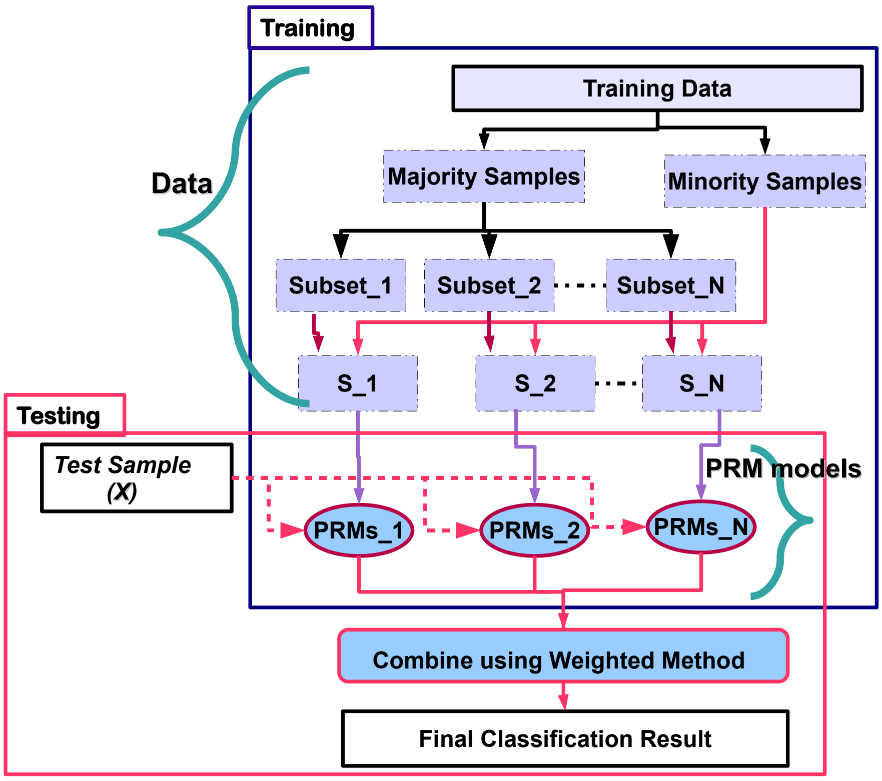


Figure 4.2: An illustration of the PRMs-IM approach.

4.4.1 Datasets

4.4.1.1 University relational database

Learning from the student relational dataset involves two main tasks: identifying the main dependencies held in the domain affecting student performance and obtaining a predictable model that can be used to predict student performance. This model should consider the imbalanced class problem that exists naturally in the student data, in which it is more important to correctly identify the students at risk ('Fail' students) which represent a small number of students as compared to the other classes. Therefore, in this chapter, PRMs-IM is evaluated on the Curtin University database for Computer Science and Commerce undergraduate students. The data was described in detail in the previous chapter, in Section 3.2.

The PRM models learned in the previous chapter pointed to the strong relationship between 'Grade', 'Status' and 'Attempt_No' attributes of the units tables, as each of these attributes refers to the student success at the unit. Therefore, in the evaluation in this

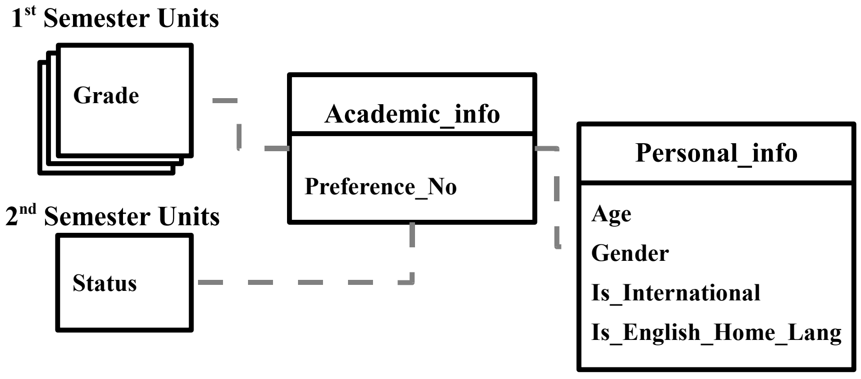


Figure 4.3: The relational schema of the students dataset used in evaluating PRMs-IM.

chapter, only one attribute of these three is used to represent student success. The 'Grade' attribute is used for semester I units, which take values of {F, 5, 6, 7, 8, 9}. The 'Grade' attribute is selected because it has more detailed information about student performance rather than the other two attributes. On the other hand, each table of semester II units includes the attribute 'Status', which represents the imbalanced two-class target attribute in this evaluation. The values of the 'Status' attribute are: 'Pass' and 'Fail'.

The attributes 'Country_of_Birth', 'Citizen_Country' from the 'Personal_Info' table and the 'Payment_Method' from the 'Academic_Info' table are not included in this evaluation, due to the insignificance of these attributes on student performance as discovered by the PRM models learned in the previous chapter. The specification of the relational structure used in this evaluation is shown in Fig. 4.3.

In this dataset, PRMs-IM is used to predict the performance of undergraduate students in semester II based on their performances in first semester. In this application, it is more important to accurately predict the 'Fail' students i.e. the students who are most likely to fail their units (deemed to be students at risk).

As each unit of semester II units represents an imbalanced situation with different imbalanced ratios, each unit must be evaluated separately. Therefore, for each unit in semester II of the Commerce and Computer Science degrees, a separate dataset is extracted from the Curtin University dataset. The dataset of a semester II unit U_i includes the records of the unit U_i and the corresponding records from the tables of 'Personal_Info', 'Academic_Info' and first semester units. Therefore, the other units of semester II rather than the U_i are not included in the database of U_i.

Table 4.1 shows the data distribution of each of the semester II units of the Bachelor of

(a)

Unit	No. Samples		No. Balanced
	'Fail'	'Pass'	Subsets
ST152	12	58	5
FCS152	11	59	5
IPE101	7	63	9

(b)

Unit	No. Samples		No. Balanced
	'Fail'	'Pass'	Subsets
MGT100	159	1556	10
MKT100	88	1627	19

Table 4.1: Summary of the data distribution of (a) the BCS (b) the BCom training datasets.

(a)

Unit	No. Samples	
	'Fail'	'Pass'
ST152	6	16
FCS152	5	16
IPE101	9	15

(b)

Unit	No. Samples	
	'Fail'	'Pass'
MGT100	30	216
MKT100	14	232

Table 4.2: Summary of the data distribution of (a) the BCS (b) the BCom test sets.

Computer Science (BCS) and Bachelor of Commerce (BCom) and also shows the number of the balanced subsets required for each unit.

The training data presented in Table 4.1 shows the data distribution for the students enrolled in the period 1999-2005. In addition, Table 4.2 shows the data distribution of an independent test set consisting of students enrolled in year 2006.

As described in Chapter 3, the BCS curriculum has a set of prerequisites for semester II units. These prerequisites resulted in a restriction of the number of students that can be used in the evaluation of this dataset. Thus, the prerequisites units (ST151 and FCS151) were restricted to include those students who passed these units and then enrolled in semester II units. This restriction resulted in lower numbers of students in the BCS dataset.

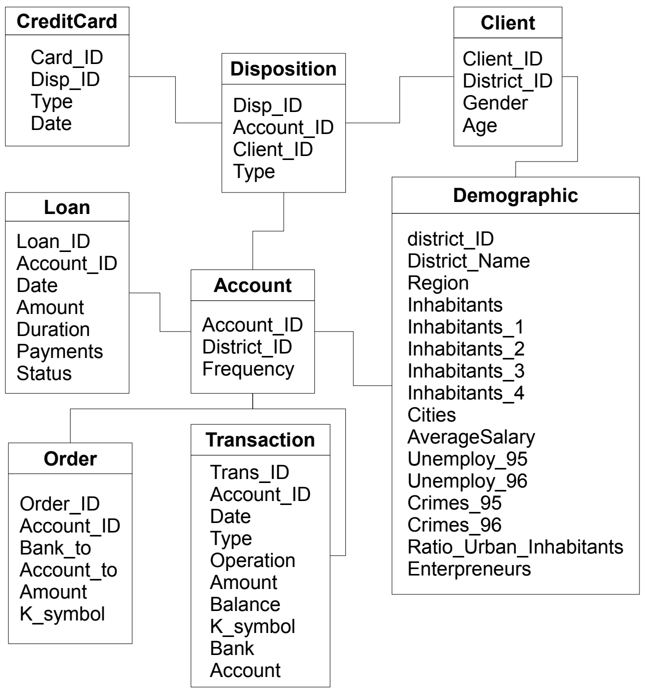


Figure 4.4: The relational schema of the Financial database.

4.4.1.2 Financial Database

This chapter also evaluates PRMs-IM on the financial database obtained from the Discovery Challenge of PKDD99 (PKDD'99 Discovery Challenge, 2009). The database is organized into a set of eight tables holding the data of 5,369 bank clients. The tables describes the characteristics of the bank clients, their transactions, accounts, cards and loans. The database schema is shown in Fig. 4.4.

This database consists of more tables and links than the Curtin University student database view used. In this database, a client can have one or more accounts and different clients can manage a single account. The 'Loan' and 'CreditCard' tables describes the bank services offered to the clients. An account can have more than one credit card but only one loan. In this chapter, the 'Status' attribute of the 'Loan' table is chosen as the target attribute in this evaluation.

The 'Status' attribute can take values of:

1. 'A' for finished loan contract with no problems.

Class	No. Samples
Class '0'	76
Class '1'	606
No. Balanced Subsets: 8	

Table 4.3: Summary of the data distribution of the Financial database.

2. 'B' for finished loan contract but the loan is not paid.
3. 'C' for running loan contract with no problems so far.
4. 'D' for running loan contract but the client is in debt.

As the objective in this chapter is to classify two-class problems, the 'Status' attribute is re-categorized into two classes: '1' for finished or unfinished loans with no problems (categories: 'A' and 'C'), and '0' for loans with problems (categories: 'B' and 'D'). The data distribution of the 'Status' attribute is shown in Table 4.3.

4.4.1.3 Movie Database

In this chapter, the MovieLens database (GroupLens Research, 2009) is also used for evaluation. MovieLens is a movie recommendation web site created by the GroupLens Reserch project about the rating of movies by people. This data is originally based on the EachMovie dataset used in several papers for building probabilistic models (Getoor *et al.*, 2000, 2001b). The main goal of using this movie dataset is to build a movie recommender system that attempts to predict how users would like movies that they have not seen yet.

The MovieLens data consists of 100,000 ratings for 1,682 movies by 943 users. In this data each user has rated at least 20 movies. This data is organized into three tables, as shown in Fig. 4.5, which is a relatively simple relational schema of similar complexity to the student database. The data consist of the following tables:

1. Rating: which includes 100,000 ratings by the users describing how much they enjoyed a specific movie on a scale from 1 to 5.
2. Movie: describing the movies, the last 20 fields of this table describes the movie genre, where a value '1' indicates that the movie is of that genre, and '0' indicates it is not. Each movie can be of several genres at the same time.
3. User: describing demographic information of the users.

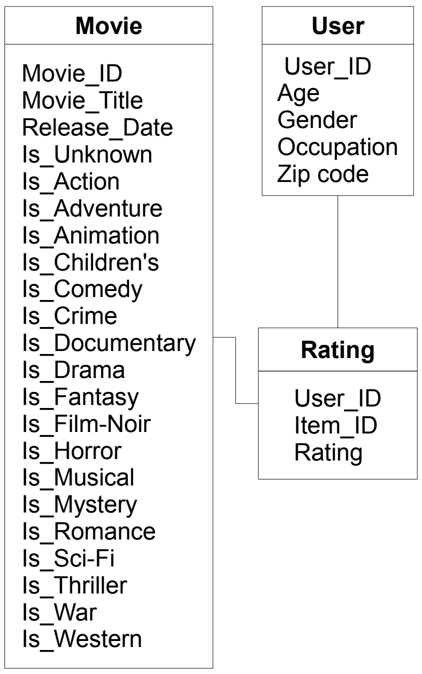


Figure 4.5: The relational schema of the MovieLense database.

In this dataset, the 'Rating' attribute is chosen to be the target attribute. Originally the rating attribute takes values 1-5 describing how much the user enjoyed the movie, where '1' represents the lowest value of enjoyment. However as PRMs-IM is proposed for binary classification, the values of the 'Rating' attribute are re-categorized as two values:

1. Class '0': which indicates that the movie is not worth watching, representing the original rating values (1-2).
2. Class '1': which indicates that the movie is enjoyable, representing the original rating values (3-5).

In this dataset, PRMs-IM is evaluated to predict the 'Rating' attribute, which has an imbalanced data distribution. The data distribution of the 'Rating' attribute is shown in Table 4.4.

Class	No. Samples
Class '0'	17,480
Class '1'	82,520
No. Balanced Subsets: 5	

Table 4.4: Summary of the data distribution of the Movie database.

4.4.2 Experimental setup

The evaluation results of PRMs-IM are presented in comparison to three types of classification techniques reviewed in Chapter 2: the traditional classifications techniques, special techniques for imbalanced data, and special techniques proposed for imbalanced relational data.

The traditional classification techniques are the Bayesian Network (BN) and Probabilistic Relational Models (PRMs) techniques. The evaluation using these techniques is performed to demonstrate the poor performance achieved by applying traditional techniques on imbalanced datasets.

The special imbalanced technique is the 'Goal Oriented K2' (Ezawa *et al.*, 1996) that is proposed for learning a goal-oriented Bayesian Network from flat data by biasing the model in favour of the imbalanced target attribute that is to be classified. In this approach, the classifier is biased to consider the relationships between certain attributes. This is achieved by firstly selecting the attributes that are more informative on the target attribute based on the conditional mutual information. Then, use the K2 algorithm (Heckerman, 1998) to construct a Bayesian network from the selected variables, with the condition that each variable should have the target attribute as a parent. This constraint is established to ensure significant impact on classification accuracy.

As the BN and 'Goal Oriented K2' techniques require flat data representations, the experimental data for these techniques are flattened from the relational data. Throughout this chapter, the term GOK2 will be used to refer to the 'Goal Oriented K2' algorithm.

The third algorithm evaluated in this chapter is a special algorithm for relational imbalanced data: MVC-IM (Guo and Viktor, 2007). MVC-IM constructs a set of classifiers, each classifier obtained from a set of features. The set of features are obtained from a flat view of the relational database. Then, an individual learner is determined from each view using either decision tree or Naïve Bayes classifier. Finally, the classifiers are combined using the majority voting approach. In this chapter, the evaluation of MVC-IM is obtained using the Naïve Bayes classification algorithm. Naïve Bayes is used in this evaluation instead of decision trees due to the probabilistic modelleing provided in Naïve Bayes, and hence the posterior probability of each class of the test sample can be obtained.

For evaluating each of these algorithms, 5-fold cross validation is used. Although, 10-fold cross validation is often considered as the standard method for cross validation to get the best performance evaluation (Witten and Frank, 2005), 5-fold is used in this chapter due to the low number of samples in some of the training datasets, especially for the BCS dataset.

The results of the evaluation are presented in terms of ROC curves and AUC values (Fawcett, 2006) discussed in Chapter 2, which are usually used for evaluating the imbalanced learning algorithms as in MVC-IM and GOK2, due to their insensitivity to the change of the data distribution. For the ROC curves and the AUC results of the 5-fold cross validation, all the test sample of the five folds are sorted according to their probability estimation. Then, the ROC curves and AUC results are calculated.

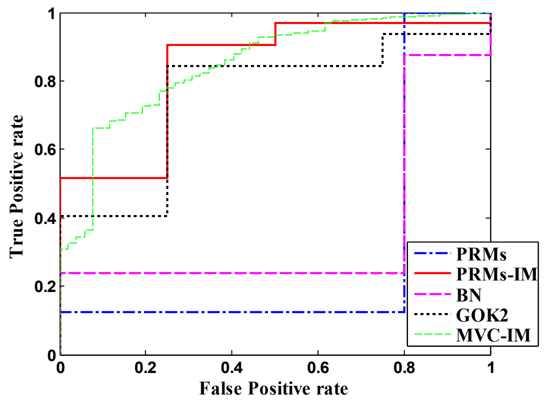
4.4.3 Results

This section presents the ROC curves and the AUC results obtained from applying each of the classification algorithms discussed in the previous section to the student, financial and movie datasets.

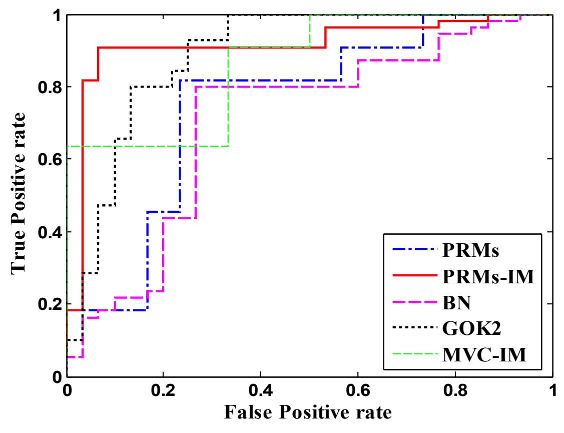
Fig. 4.6 and 4.7 show the ROC curves resulting from the 5-fold cross validation on the student, movie and financial datasets, respectively. The ROC curves of the separate test set (of 2006 students) are shown in Fig. 4.8.

Tables 4.5 and 4.6 show the AUC results obtained from each experiment for the 5-fold cross validation and the separate test data, respectively. The AUC tables also show the average AUC for each algorithm over the datasets. The best AUC result of each experiment for each dataset is shown in bold.

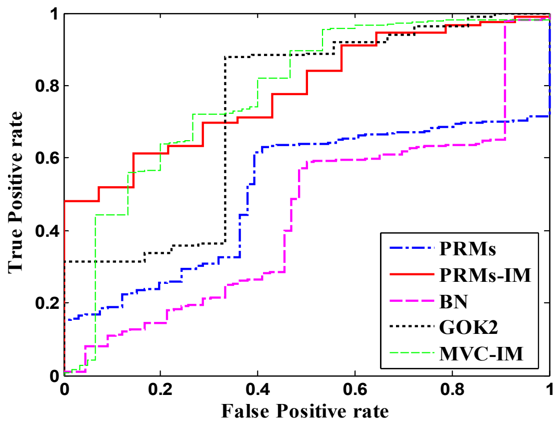
The results shown in the ROC curves and the AUC tables confirms the poor performance of the traditional learning algorithms on imbalanced datasets, as shown by the differences in performances of PRMs and BN. This confirms the argument of Ezawa *et al.* (1996) and Cowell *et al.* (1993) that even if the learned model can discover correct relationships in the domain, it could perform poorly if applied for a specific goal such as classification.



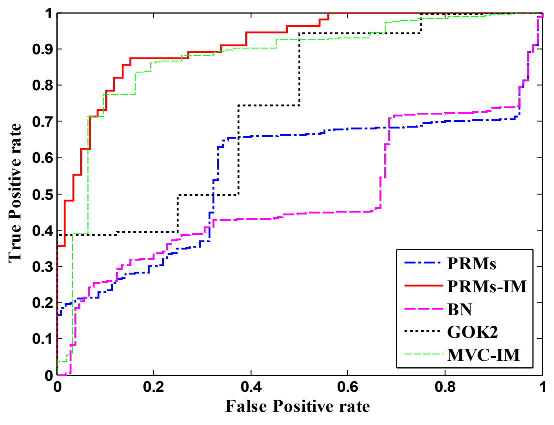
(a)



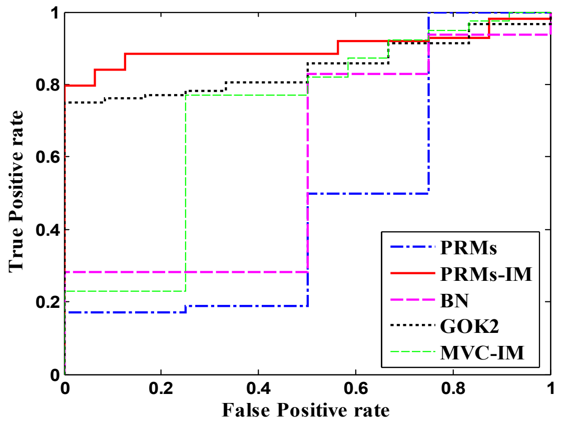
(b)



(c)

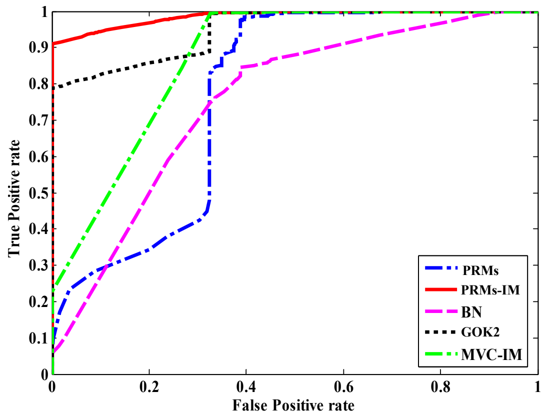


(d)

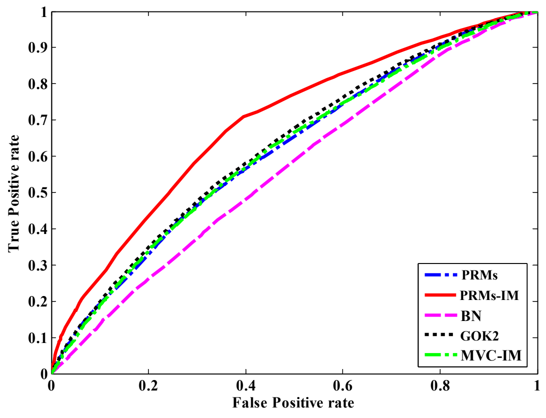


(e)

Figure 4.6: ROC curves of 5-fold cross validation of the student relational database on: (a) ST152 (b) IPE101 (c) MKT100 (d) MGT100 © 2008 IEEE (Ghanem *et al.*, 2008)
(e) FCS152 © 2008 IEEE (Ghanem *et al.*, 2008)



(a)



(b)

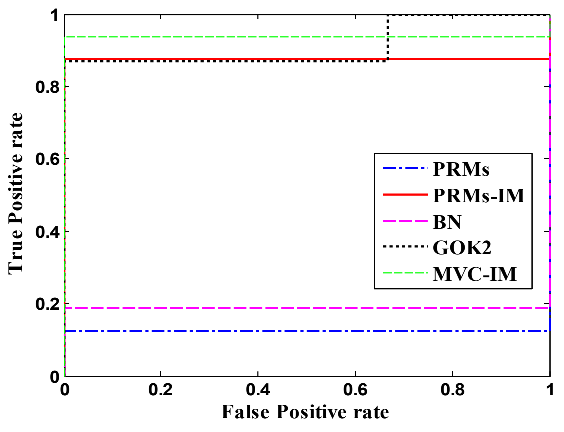
Figure 4.7: ROC curves of 5-fold cross validation of the (a) Financial (b) Movie datasets.

Dataset	Algorithm				
	BN	PRMs	GOK2	MVC-IM	PRMs-IM
MGT100	0.488	0.558	0.737	0.871	**0.914**
MKT100	0.440	0.508	0.735	**0.786**	**0.786**
ST152	0.365	0.300	0.758	**0.849**	0.839
FCS152	0.582	0.465	0.846	0.711	**0.901**
IPE101	0.704	0.751	0.889	0.863	**0.913**
Financial	0.754	0.770	0.948	0.871	**0.986**
Movie	0.565	0.617	0.628	0.617	**0.692**
Average AUC	0.557	0.567	0.792	0.795	**0.864**

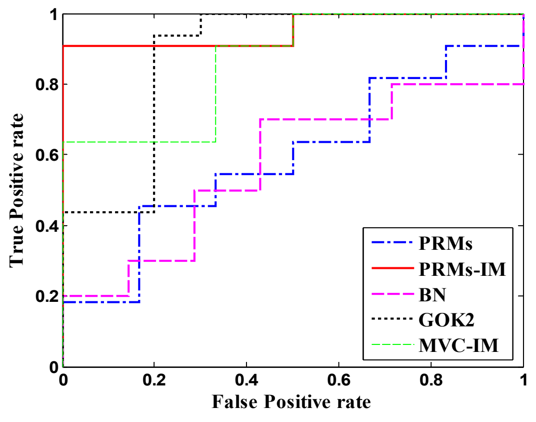
Table 4.5: The AUC results of the 5-fold Cross validation.

Dataset	Algorithm				
	BN	PRMs	GOK2	MVC-IM	PRMs-IM
MGT100	0.413	0.408	0.645	0.845	**0.921**
MKT100	0.603	0.572	0.714	0.704	**0.788**
ST152	0.187	0.125	0.913	**0.937**	0.875
FCS152	0.400	0.380	0.840	0.730	**0.927**
IPE101	0.571	0.590	0.881	0.863	**0.954**
Average AUC	0.435	0.415	0.799	0.816	**0.893**

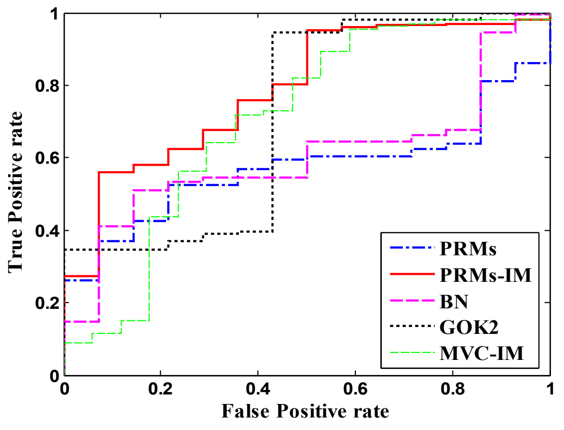
Table 4.6: The AUC results of the student dataset for the 2006-test set validation.



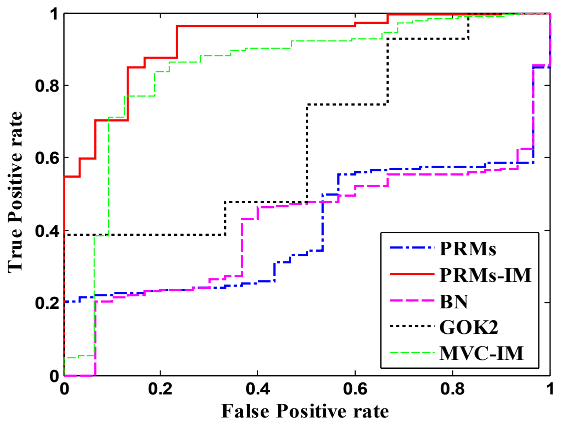
(a)



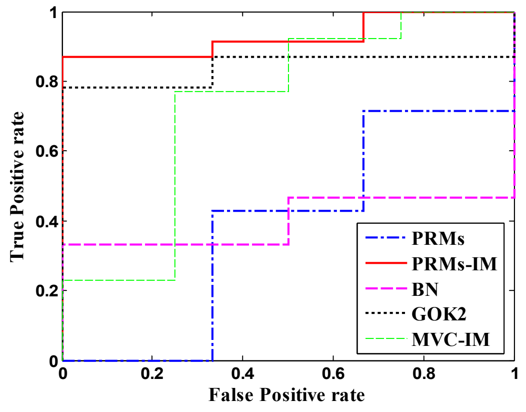
(b)



(c)



(d)



(e)

Figure 4.8: ROC curves of 2006-test set of the student relational database on: (a) ST152 (b) IPE101 (c) MKT100 (d) MGT100 © 2008 IEEE (Ghanem *et al.*, 2008) (e) FCS152 © 2008 IEEE (Ghanem *et al.*, 2008)

The results of MVC-IM and GOK2 were varied, where on some datasets they performed very well but scored less well on others. However, generally they produced better results than the traditional learning algorithms. The average AUC results show that MVC-IM was able to generally improve on GOK2, which is attributed to the use of relational data over flattened data for learning. On the other hand, out of the 12 individual experiments conducted, GOK2 outperformed MVC-IM results in seven experiments which confirms the power of BN in classification when learned for a specific goal. However, it lacks the concept of relational learning that resulted in lower performances in the other experiments and the overall average performance.

As for PRMs-IM, the results show that PRMs-IM was able to generally improve over all the other methods. PRMs-IM has shown superior results over the traditional techniques and is generally better than the specialized algorithms. In terms of the average AUC over all the datasets, PRMs-IM outperformed all the other methods of traditional and special algorithms.

Furthermore, for the individual 12 experiments conducted, PRMs-IM outperformed all the other methods in nine of the experiments. For the remaining three experiments, the result of PRMs-IM was equal to that of MVC-IM in one experiment in the case MKT100, and slightly less than the best results in the other two experiments in the case of ST152 for the cross validation and test set. These results also show that PRMs-IM was able to classify the minority class well for almost all the datasets. In addition, PRMs-IM has achieved promising results in correctly predicting students at risk in the student relational data and has scored perfect AUC results in the cases of 'MGT100', 'FC152' and 'IPE101' with values ≥ 0.900.

These results for PRMs-IM again support the arguments of Ezawa *et al.* (1996) and Cowell *et al.* (1993) that though the learned Bayesian networks can discover interesting and correct relationships in the data, they need to be carefully configured and examined when they are built for a specific task. This argument holds also for PRMs, as PRMs are basically an extension of Bayesian networks. This has been shown clearly by the poor results of PRMs when applied directly to the imbalanced relational data, and by the superior results when configuring PRMs for handling the specific classification task of the imbalanced class problem.

Interestingly, the average AUC results show that the best results are obtained by PRMs-IM followed by MVC-IM and then GOK2. Though GOK2 has scored better results than traditional learning, it was not suitable for the relational learning and hence was outperformed by the special relational learning algorithms in PRMs-IM and MVC-IM. This observation confirms the concept of constructing special algorithms for imbalanced relational data.

These results confirm the benefits gained from the PRMs-IM framework, which has resulted in better performance than for MVC-IM. The results of PRMs-IM also show the significance of applying PRMs for relational learning but with the specific purpose of handling the imbalanced problem. In this framework, the ensemble of PRM models are created on balanced subsets from the original relational data. Therefore, each subset maintains the same relational structure of the tables, attributes and relationships as those of the original data but with a balanced data distribution. This relational structure allows PRMs to learn from a relational subset that is identical to the original data and hence can obtain good results in terms of discovering the relationships in the domain, and at the same time not being hindered by the imbalanced problem.

Following this concept, each learned PRM model is a well-learned PRM candidate obtained from a single subset, however, it has enough knowledge about the original data and hence can propose confident classification results when it is applied to new test cases. This is achieved as a result of learning PRM models from the same input space as that of the original, and also due to the confirmed effectiveness of PRMs in learning from relational domains. In addition, using the weighted voting strategy in PRMs-IM had a significant benefit on the classification results by allowing the most confident expert models to have more effect on the final result.

In contrast, in MVC-IM, each classifier is built on a flat view of the relational database with a set of features of the domain, and hence each classifier is an expert classifier in a specific region of the domain. Furthermore, MVC-IM employs majority voting and hence allows non-expert classifiers to vote towards the overall classification.

4.5 Conclusion

This chapter has outlined a framework (PRMs-IM) for learning from imbalanced relational data, which occurs in several real-world domains. PRMs-IM extends Probabilistic Relational Models (PRMs) to handle the imbalanced class problem via ensemble learning. The ensemble learning employed in PRMs-IM has focused on obtaining a high overall classification accuracy, as well as accurately classifying the rare cases that are usually of high importance. The proposed framework attains this goal by constructing an ensemble of PRM models learned from well-balanced subsets extracted from the original imbalanced dataset. Each of the subsets maintains the same relational structure of the original dataset but with balanced distribution of the minority and majority samples. PRMs-IM has obtained superior results when evaluated as a two-class algorithm for learning from a number of highly imbalanced real-world datasets, and has achieved promising results for predicting students at risk on the student relational dataset. The results show that PRMs-IM has successfully outperformed the traditional learning techniques and obtained improved performances when compared with the specialized imbalanced algorithms. The main complexity of this framework is that it allows only one imbalanced target attribute to be modelled at a time. Thus, for a classification problem with several imbalanced attributes, each attribute must be modelled independently. In real-world domains, there are usually a number of imbalanced target attributes that are of interest to be classified. Moreover, building an independent model for each imbalanced attribute prohibits the attributes from interacting with each other. Nevertheless, it is commonly the case that the information obtained from one attribute can help in achieving better conclusions about the others, and hence building one model of all the attributes can effectively enhance the performance results. The problem of handling the classification of multiple imbalanced attributes in relational data forms the topic of the next chapter.

Chapter 5

Classifying Multiple Imbalanced Attributes in Relational Data

The previous chapter addressed the problem of learning from imbalanced relational data and proposed a new model: PRMs-IM. This model has shown promising results in learning from a number of imbalanced real-world relational datasets, and particularly for classifying the rare cases that are usually of high importance. However, PRMs-IM has been proposed with the assumption of modelling a single imbalanced attribute at a time. Therefore, for classifying multiple imbalanced attributes of a given problem, PRMs-IM constructs an independent model for each attribute. In these models, it is assumed that these attributes are independent, which is not the case in many domains. This chapter investigates the problem of classifying multiple imbalanced attributes in relational data and proposes an extension of PRMs-IM. The proposed approach utilizes the balancing concept of PRMs-IM and the popular bagging ensemble approach to obtain a single model for classifying multiple attributes. The performance of the proposed method is shown on real-world relational imbalanced datasets obtained from the Curtin University student database.

5.1 Introduction

A wide range of classification techniques have been developed for classifying different types of objects and for various types of application domains, including: handwritten digit recognition, text classification and speech recognition. However, most of these techniques are proposed for typical situations that do not represent the majority of real-world domains. Examples of such assumptions include: learning from flat data files, classifying binary attributes and assuming balanced data distribution.

The previous chapter reviewed the problem of the typical assumption of learning from a single flat data file with balanced data distribution, and introduced PRMs-IM to handle the classification of imbalanced attributes in relational domains. However, PRMs-IM, as for many other classification algorithms, is based on the typical assumption of classifying a single attribute at a time. Therefore, for classifying N imbalanced attributes of the same problem domain, N independent PRMs-IM models must be constructed, one for each imbalanced attribute.

The construction of independent models not only increases the number of models required for training and inference, but also loses the different relations between the imbalanced attributes that could help in achieving better performance results. In many real-world problems, there is often more than one attribute to be classified, which are related in different ways. Therefore, constructing a single model showing the different interactions between these attributes would significantly allow the information of one attribute to aid in reaching a better understanding about the other related attributes. For example, constructing a single model for classifying the performance of the second semester units, as discussed in the previous chapter, can help in obtaining better prediction results, as the performance of a unit can often indicate the performance of the other units.

Therefore, this chapter proposes an extension of PRMs-IM, named PRMs-IM2, for classifying multiple imbalanced attributes in relational data. The main idea behind the proposed approach is combining the concepts of PRMs-IM and the 'Bagging' ensemble approach (Breiman, 1996) to classify multiple imbalanced attributes in a single model. The basic idea behind PRMs-IM2 is the construction of an ensemble by varying the training data, such that each classifier is trained on a subset that includes a balanced distribution of each imbalanced attribute. Thus, each classifier is an expert classifier for each of the imbalanced attributes, and can vote for any attribute for new test instances.

PRMs-IM2 constructs the balanced training sets by first applying the balancing concepts of PRMs-IM to generate independent balanced subsets of each imbalanced attribute. Then, the independent subsets are integrated using the bagging approach to generate new training sets that consist of all the imbalanced attributes. Consequently, an independent PRM model is learned from each training set, and hence the PRM model captures the different relationships between the attributes. The PRM models are then combined to obtain the final classification results.

Therefore, PRMs-IM2 models all the significant imbalanced attributes in a single model showing the different interactions between the attributes. In comparison to the methods reviewed in Chapter 2 for classifying multiple attributes, this interaction between the attributes achieved in PRMs-IM2 cannot be reached by the typical independent, hierarchical or combined approaches, or even by the special bilinear model (Tenenbaum and Freeman,

2000) that employs linear modelling. Moreover, although the mutual suggestions approach proposed by Hiraoka and Mishima (2002) relatively achieve such interactions by allowing the predictions of one classifier to update the predictions of others, yet this approach requires an independent classifier for each attribute and is mainly proposed for the special case of modelling two attributes.

In this chapter, PRMs-IM2 is evaluated on a number of real-world imbalanced data extracted from the Curtin University student dataset presented in Chapter 3. The results are shown in terms of Receiver Operating Characteristics (ROC) curves and the Area Under ROC (AUC) values that are usually used for measuring imbalanced learning algorithms (Fawcett, 2006). In addition, the results also show the number of models required for PRMs-IM2 in comparison to the number of models required by the typical methods of classifying multiple attributes.

The remainder of this chapter is structured as follows: Section 5.2 introduces the bagging ensemble approach, followed by the description of PRMs-IM2 in Section 5.3. The datasets evaluated in this chapter and the evaluation results are then presented in Section 5.4. Finally, Section 5.5 summarizes the chapter.

5.2 Bagging approach

Several studies have investigated the ensemble technique of combining multiple classifiers to obtain a single classification system. The ensemble learning has generally demonstrated its improved performance not only in terms of the training time, but also in terms of the robustness and performance of the classification, as discussed in Chapter 4. A popular approach of creating ensembles is the "Bagging" approach (Breiman, 1996; Freund and Schapire, 1996), which relies on sampling techniques to obtain different training sets for each classifier.

In the bagging approach, a set of different training sets are sampled from the original data. Each training set is generated at random by sampling N samples, with duplicates, from the original data, where N is the size of the original data. Then, a classifier is learned from each training set using a particular learning algorithm, such as a Decision tree or Naïve Bayes classifier. For classifying new test instances, the outputs of the classifiers are combined. The most popular method for combining the classifier outputs is majority voting, by allowing each classifier to have a vote on the test instance, and hence the class that receives the more votes is the one considered to be the correct class.

It is important to note that in the bagging approach, the classifiers have equal weights, and thus have equal votes for the test instance. The bagging approach has been shown to be quite effective, especially, for 'unstable' learning algorithms, in which a small change in the training data can result in a large difference in the predictions (Breiman, 1996). Moreover, Opitz and Maclin (1999) have shown that bagging approach is often appropriate for most problems as it nearly always outperform a single classifier.

5.3 Methodology

PRMs-IM2 is an ensemble approach that combines the concepts of PRMs-IM and bagging approaches to classify multiple imbalanced attributes in relational data. The main tasks addressed in this framework are: handling the imbalanced class problem in relational data by employing PRMs-IM approach, and constructing a single model for classifying multiple attributes by using the concept of the bagging approach .

To illustrate the PRMs-IM2 approach, consider a relational dataset $\mathcal{S}$. $\mathcal{S}$ consists of a set of attributes $(X_1, X_2, ..., X_M, Y_1, Y_2, .., Y_N)$ organized into a set of tables $(T_1, T_2, ..., T_y)$ that are related to each other via different relationships. $(Y_1, Y_2, .., Y_N)$ are the domain imbalanced attributes that are of interest to be classified. A target table T_i is a table that includes an imbalanced attribute Y_i. Each imbalanced attribute Y_i is a two-class attribute, where the majority of samples belong to the majority class $Y_{i_{mj}}$ and only a few samples belong to the minority class $Y_{i_{mr}}$.

In order to classify the imbalanced attributes $(Y_1, Y_2, .., Y_N)$ in $\mathcal{S}$, PRMs-IM2 applies the following stages:

1. **The Balancing Stage:** in this stage, PRMs-IM2 follows the balancing concept of PRMs-IM to construct balanced subsets of each imbalanced attribute Y_i. This is achieved as follows:

   - For each Y_i, PRMs-IM2 constructs n balanced relational subsets ($Y_i s = \{Y_i$-$s_1, Y_i s_2, ..., Y_i s_n\}$), by following the same approach of PRMs-IM. The value n is determined as the ratio difference between the number of samples of the majority class and those of the minority class of Y_i.
   - Each balanced subset $Y_i s_i$ consists of the attributes $(X_1, X_2, ..., X_M, Y_i)$.
   - For each $Y_i s_i$, PRMs-IM2 assigns a balanced data distribution of Y_i, such that

the subset includes all the minority samples and an equal random selection of the majority samples.

- For each $Y_i s_i$, PRMs-IM2 allocates the data for $(X_1, X_2, ..., X_M)$ related the records selected of Y_i in $Y_i s_i$.

2. **Aggregation Stage:** In this stage, PRMs-IM2 aggregates the balanced subsets constructed in the previous stage to build new training datasets. Thus, using the bagging concept, the subsets are sampled randomly to construct new training datasets, such that each dataset will include the data of all the imbalanced attributes. The sampling procedure is performed as follows:

   - PRMs-IM2 creates L empty datasets $(S_1, S_2, ..., S_L)$ that have the same relational structure as $\mathcal{S}$. As each Y_i has a different number (n_i) of balanced subsets based on the statistical distribution of Y_i, L is determined as the largest number of subsets of $(Y_1, Y_2, .., Y_N)$, as follows:

$$L = \max_{i=1..N} n_i \tag{5.1}$$

   - For each S_i, PRMs-IM2 allocates a random subset from each imbalanced attribute subsets $Y_i s$. Thus, each S_i will include a balanced distribution for each imbalanced attribute.

Fig. 5.1 shows an example of the balancing and aggregation stages of PRMs-IM2 for two imbalanced attributes (Y_1, Y_2). Each of the imbalanced attributes has two classes $(-1, 1)$, where class '$1'$ is the majority class. The majority class of both attributes is twice the size of the minority class. Therefore, in the balancing stage, two balanced subsets are created for each attribute, where each balanced subset includes all the samples of the minority class and an equal number of random samples of the majority class.

At the end of the balancing stage, two subsets (Y_{1a}, Y_{1b}) are created for Y_1, and two subsets (Y_{2a}, Y_{2b}) are created for Y_2. These balanced subsets are then randomly sampled to generate new training sets, such that each new training set includes the data of both Y_1 and Y_2. For Y_1 and Y_2 attributes, the largest number of the balanced subsets is two, thus, in the aggregation stage, two datasets are created (S_1, S_2). Each S_i is assigned to a random subset from each Y_i. In this example, S_1 consists of (Y_{1a}, Y_{2a}), and S_2 consists of (Y_{1b}, Y_{2b}).

Moreover, in the first stage, PRMs-IM2 also follows the same procedure of PRMs-IM to create well-defined subsets in terms of the relational structure but with a balanced data

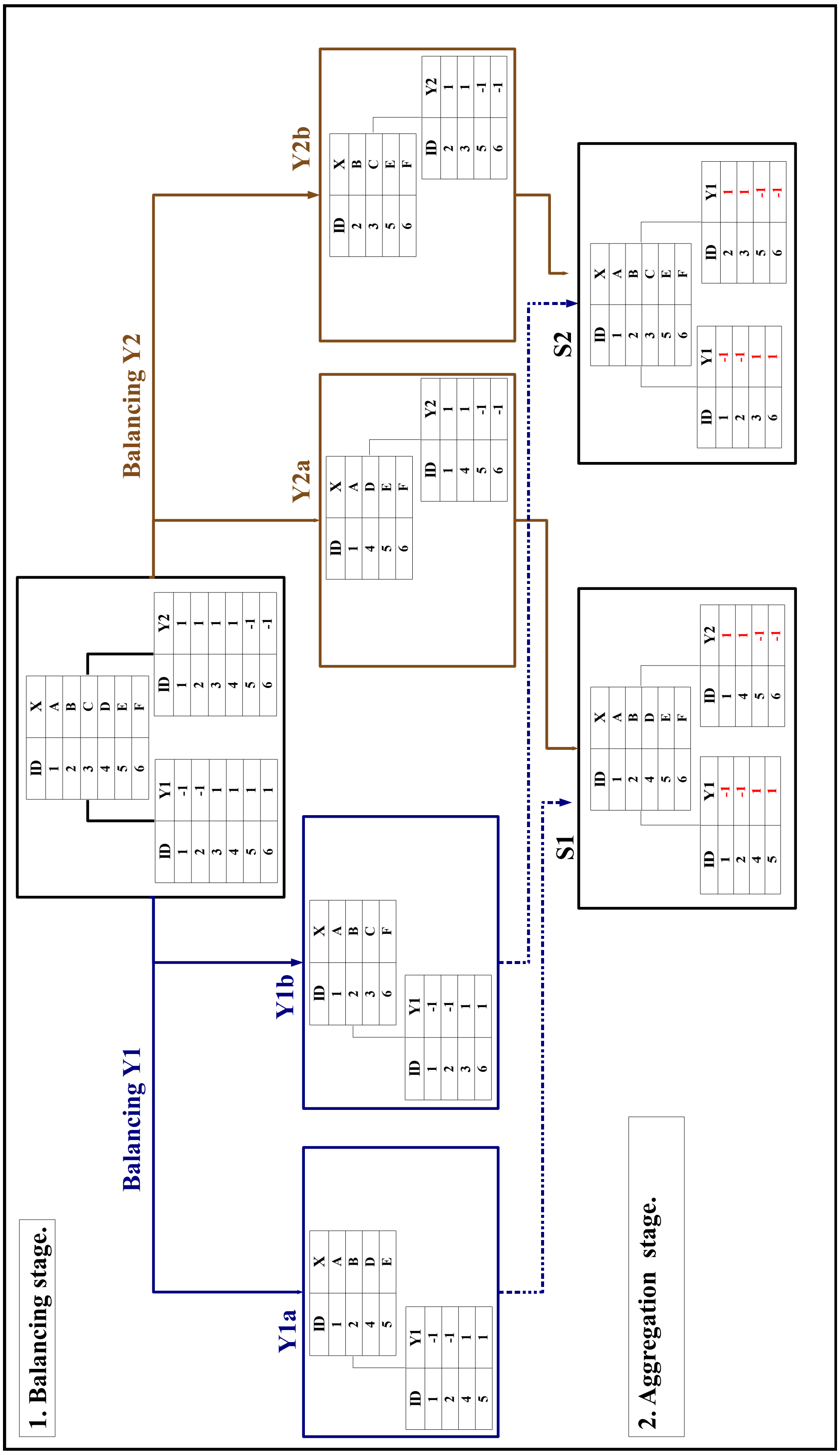


Figure 5.1: An illustration of the balancing and aggregation stages of PRMS-IM2.

distribution. Therefore, for n balanced relational subsets of Y_i, PRMs-IM2 first creates n new empty databases that have the same relational structure as that of $\mathcal{S}$. Then, each subset is assigned all the $Y_{i_{mr}}$ samples and an equal number of random samples from $Y_{i_{mj}}$. Finally, PRMs-IM2 allocates the records of $(X_1, X_2, ..., X_M)$ that are linked to those selected records of Y_i. Following this procedure, roughly n well-defined balanced relational subsets of Y_i are created.

After constructing the L relational datasets in the second stage, an independent PRM model M_i is learned from each relational subset L_i using the learning techniques described by Getoor (2002). Therefore, a PRM model will be learned from each dataset that has the same relational structure as the original data with all attributes and tables. Then, PRMs-IM2 employs the weighted voting strategy to combine the PRM models following the same procedure of PRMs-IM described in the previous chapter. Thus, each M_i has a different weighting factor $M_{i_w}Y_i$ for each attribute Y_i. $M_{i_w}Y_i$ is calculated as the average AUC results of using M_i for classifying Y_i over the training datasets other than that used for training M_i.

For classifying new test samples, the samples are evaluated by the PRM models. Then, PRMs-IM2 combines the outputs of the PRM models using the weighted voting method. For example, to classify a new test sample x, x is evaluated by each M_i. Consequently, given the values of the attributes $(X_1, X_2, ..., X_M)$ of x, M_i outputs the probabilities of each of the imbalanced attributes $(Y_1, Y_2, .., Y_N)$. Therefore, for each Y_i, each M_i outputs the probabilities $(M_i(x)Y_{i_{mj}}, M_i(x)Y_{i_{mr}})$ for assigning x to the majority and minority classes, respectively. Then, the score of each of the majority and minority classes of Y_i is calculated as the summation of the weighted outputs of the PRM models. As a result, Y_i of x is assigned to the class with the highest weighted score. For example, classifying the attribute Y_i of the test sample x is calculated as follows:

$$F(x) = \underset{m \in (Y_{i_{mj}}, Y_{i_{mr}})}{\operatorname{argmax}} \left(\sum_{\forall M_i} M_i(x)Y_{i_m} * M_{i_w}Y_i \right) \tag{5.2}$$

where $M_i(x)Y_{i_m}$ is the probability assigned by model M_i for class m of attribute Y_i, and $M_{i_w}Y_i$ is the weighting factor of model M_i for attribute Y_i.

The PRMs-IM2 algorithm is summarized in Fig. 5.2, and an illustration of the approach is shown in Fig. 5.3. In summary and as shown in the figures, PRMs-IM2 first constructs balanced subsets of each imbalanced attribute by following the concept of the PRMs-IM approach. Then, using the bagging concept, the subsets are sampled randomly to construct new training datasets, such that each dataset includes the data of all the imbalanced attributes and the data of each of these attributes has a balanced data distribution. An individual relational PRM model is learned from each dataset. Then, the learned models are combined using the weighted voting strategy.

In this approach, it is important in the first stage to construct the balanced subsets of Y_i, such that each subset includes only the data of $(X_1, X_2, ..., X_M)$ and that of Y_i, i.e. excluding the data of the other imbalanced attributes other than Y_i. This is essential for creating L balanced datasets of all the attributes in the second stage. For example, if subsets of Y_i include the related data of Y_j, then Y_i samples will be balanced but those of Y_j will not. This situation gets more complicated in the aggregation stage. For instance, if a subset S_k is constructed by including the random subsets: $Y_i s_l$ and $Y_j s_h$ from Y_i and Y_j, respectively. Then, the data records of Y_j in S_k will include data from $Y_j s_h$, which are balanced, and the Y_j records from $Y_i s_l$, which are not balanced.

An example of this case is shown in Fig. 5.4. In this example, the same data of Fig. 5.1 is used, where there are two imbalanced attributes: Y_1 and Y_2, in which class '$1'$ represents the majority class that is as twice as the minority class. Therefore, in the balancing stage, two balanced subsets are created for each attribute. However, in this example, the subsets: Y_{1a} and Y_{1b}, include the related records of the other imbalanced attribute Y_2. Therefore, each subset will include a balanced distribution of the corresponding attribute but imbalanced distribution of the other attribute. For example, the subset (Y_{1a}) includes a balanced distribution of Y_1, but the distribution of Y_2 is not.

This imbalanced distribution is carried out to the aggregation stage, when new training sets are created from the previous stage. For example, the training set S_1 is constructed from Y_{1a} and Y_{2a}. Thus, the data in S_1 for Y_1 consists of those balanced records from Y_{1a} and the imbalanced records from Y_{2a}, and similarly for Y_2. This results in imbalanced distribution in S_1 for both Y_1 and Y_2.

On the other hand, in the proposed approach, as shown earlier in Fig. 5.1, this imbalanced situation is solved by constructing the balanced subsets of Y_i to include only the data of Y_i and exclude that of the other imbalanced attributes. For example, the training set S_1 in Fig. 5.1 is constructed from Y_{1a} and Y_{2a}. Therefore, the data of Y_1 and Y_2 in S_1 consist of the balanced data from Y_{1a}, and Y_{2a}, respectively. Consequently, S_1 will include balanced representation of both Y_1 and Y_2.

Furthermore, in this task of classifying multiple imbalanced attributes in relational domains, the benefits gained from the relational structure are greater when compared to the flat representation. In a relational domain that consists of a set of imbalanced attributes,

```
Model Generation:
  • Let $N$ be the number of imbalanced attributes to be classified in relational
    database $\mathcal{S}$.
  • For each imbalanced attribute $Y_i$ of the $N$ imbalanced attributes:
      – Let $Y_{i_{mj}}$ and $Y_{i_{mr}}$ be the majority and minority classes of $Y_i$,
        respectively.
      – Let $n_i(mj)$ and $n_i(mr)$ be the number of samples of the majority and
        minority classes of $Y_i$, respectively.
      – Compute $n_i = n_i(mj)/n_i(mr)$, where $n_i$ is the number of balanced subsets
        required for $Y_i$.
      – For each of $n_i$ iterations:
          * Create a subset $Y_is_i$.
          * Assign all the $n_i(mr)$ samples from $Y_{i_{mr}}$ to $Y_is_i$.
          * Sample $n_i(mr)$ samples with replacements from $Y_{i_{mj}}$ to $Y_is_i$.
          * Allocate other data $(X_1, X_2, ..., X_M)$ from $\mathcal{S}$ according to those
            selected of $Y_i$ in $Y_is_i$.
  • Compute $L = \max_{i=1..N}(n_i)$.
  • For each of $L$ iterations:
      – Create $S_i$ database of same structure as $\mathcal{S}$.
      – For each $Y_j$, allocate a random subset $Y_js_k$ from $Y_j$ subsets to $S_i$.
      – Learn PRM model $M_i$ from $S_i$.
  • For each PRM model $M_i$, and for each $Y_j$:
      – Use $M_i$ to classify $Y_j$ of instances from each $S_k$, such that $i \neq k$.
      – Compute $M_{i_w}Y_j =$ average AUC of using $M_i$ to classify $Y_j$ over datasets.
Classification:
  • For each $Y_j$:
      – For each of the $M_i$ models:
          * Predict $Y_j$ class of instance using $M_i$.
          * For each class $C_i$ of $Y_j$, add ($M_{i_w}Y_j$ multiplied by the probability
            predicted of $C_i$ by $M_i$) to the score of $C_i$.
      – Return $Y_j$ class with the highest score.
```

Figure 5.2: The algorithm of PRMs-IM2.

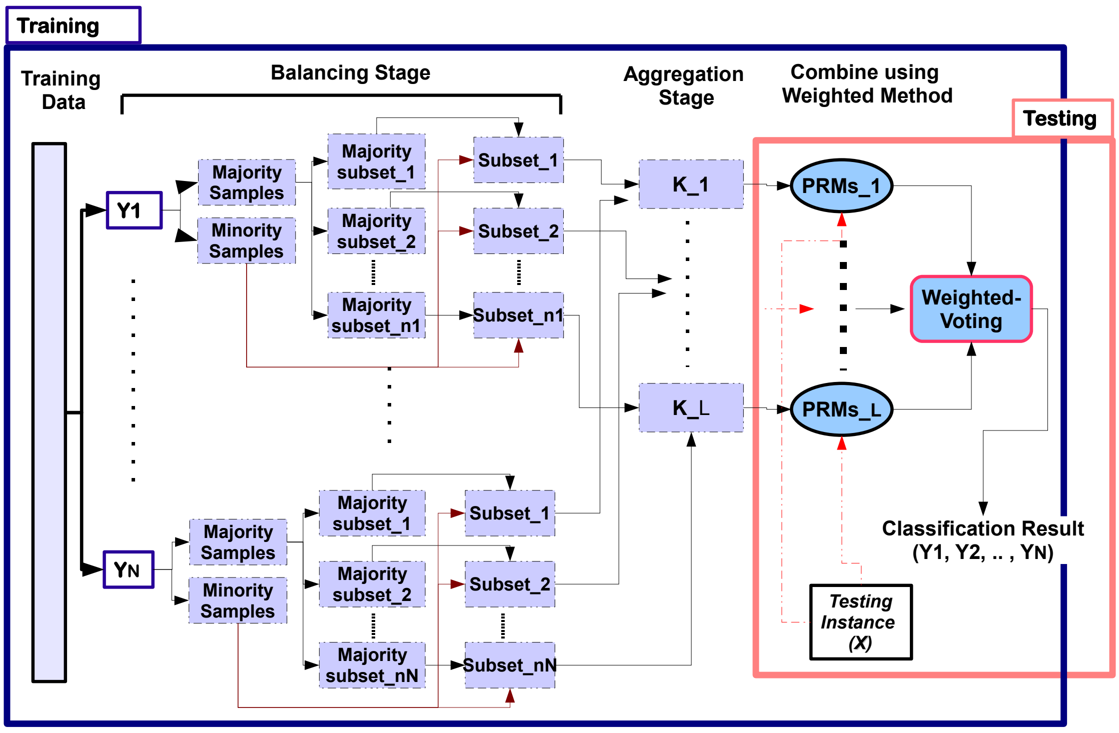


Figure 5.3: An illustration of PRMs-IM2 framework for classifying multiple imbalanced attributes © 2009 Springer (Ghanem *et al.*, 2009).

where there is one imbalanced attribute Y_i per table, the balanced subsets of Y_i can be easily constructed to include only the data of Y_i and those of $(X_1, X_2, ..., X_M)$. Consequently, the balanced subsets can be simply aggregated to create the L relational datasets, such that each training set includes the data of all the imbalanced attributes without any missing values. In Fig. 5.1, shown earlier, the final training sets include complete data for each imbalanced attribute as a result of maintaining the data of each attribute in the corresponding table.

In contrast, in the case of the flat data representation, following the balancing and aggregation concepts of PRMs-IM2 can result in missing values in the final training sets. An example of this situation is shown in Fig. 5.5. In this figure, the flat data file consists of two imbalanced attributes (Y_1, Y_2), where class '1′ is the majority class. In the balancing stage, two balanced subsets (Y_{1a}, Y_{1b}) are created for Y_1, and similarly two balanced subsets (Y_{2a}, Y_{2b}) are created for Y_2. Therefore, each balanced subset consists of a balanced distribution of the corresponding attribute.

Consequently, two flat sets (S_1, S_2) are constructed in the aggregation stage. Each set is constructed by randomly sampling a subset from the balanced subsets of each imbalanced

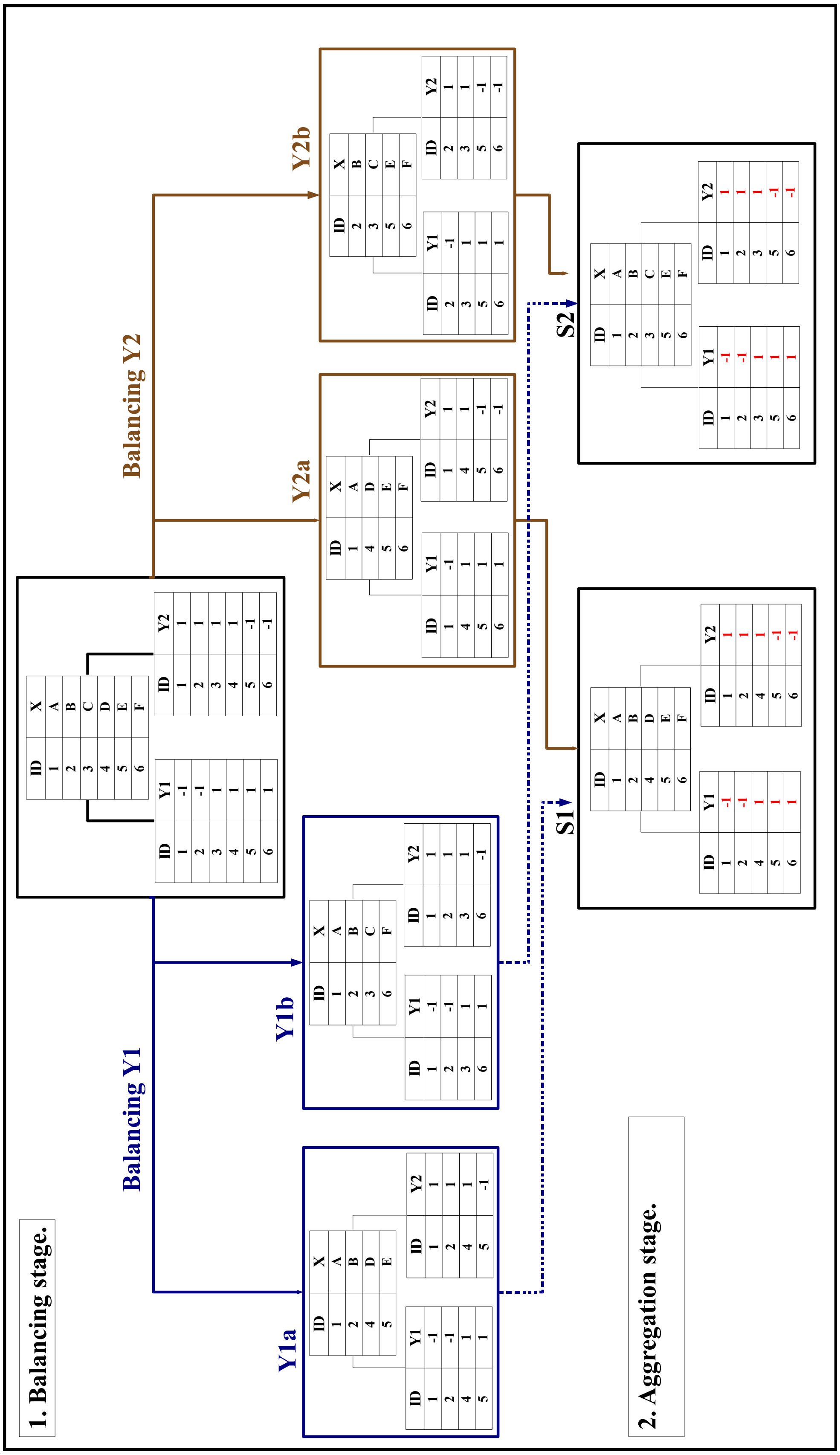
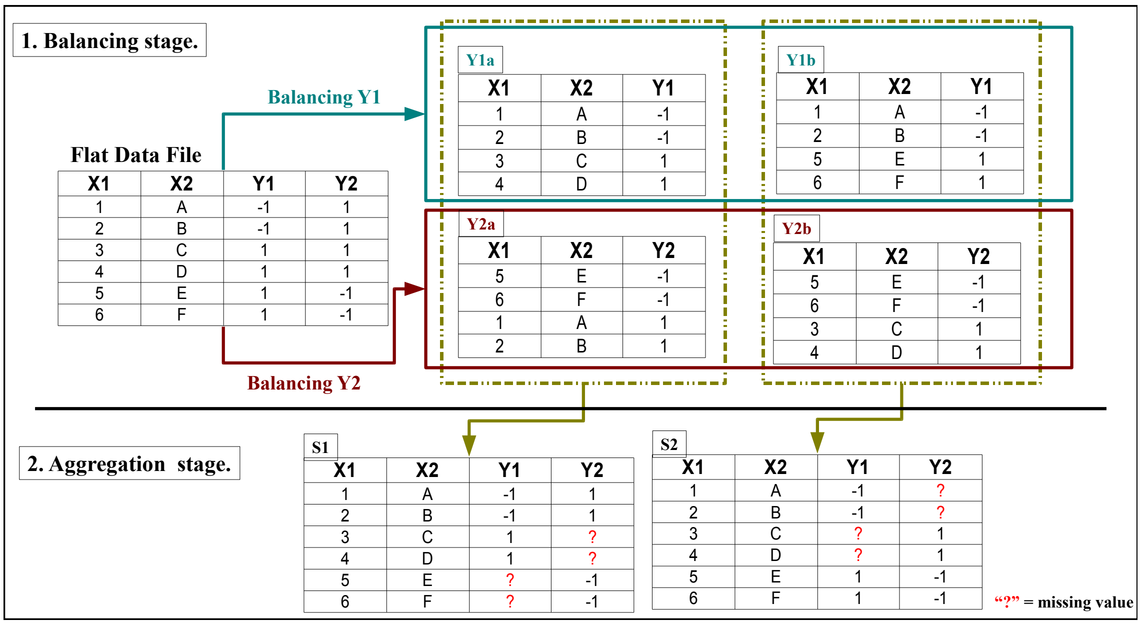


Figure 5.4: An illustration of the imbalanced situation caused by including the data of all the imbalanced attributes in the balancing stage.



Flat Data File

X1	X2	Y1	Y2
1	A	-1	1
2	B	-1	1
3	C	1	1
4	D	1	1
5	E	1	-1
6	F	1	-1

Y1a

X1	X2	Y1
1	A	-1
2	B	-1
3	C	1
4	D	1

Y1b

X1	X2	Y1
1	A	-1
2	B	-1
5	E	1
6	F	1

Y2a

X1	X2	Y2
5	E	-1
6	F	-1
1	A	1
2	B	1

Y2b

X1	X2	Y2
5	E	-1
6	F	-1
3	C	1
4	D	1

S1

X1	X2	Y1	Y2
1	A	-1	1
2	B	-1	1
3	C	1	?
4	D	1	?
5	E	?	-1
6	F	?	-1

S2

X1	X2	Y1	Y2
1	A	-1	?
2	B	-1	?
3	C	?	1
4	D	?	1
5	E	1	-1
6	F	1	-1

Figure 5.5: An example of the flat representation problem that could result when balancing multiple attributes.

attribute. For example, the set S_1 is constructed from Y_{1a} and Y_{2a}. However, as a result of using the flat file representation, each of S_1 and S_2 could include missing values for (Y_1, Y_2). For example, the third record of the flat set S_1 consists of the following data $(‘3', ‘C', ‘1', ‘?')$ of the attributes $(X1, X2, Y1, Y2)$, thus, the value of Y_2 is missing or unknown. The value of Y_2 in this record is missing because this record is obtained from the balanced subset Y_{1a}, which is related to Y_1, and hence the data of Y_2 is not available. Likewise, the values of attribute Y_2 are missing in the fifth and sixth records of S_1.

5.4 Experimental results

In this section, the evaluation results of PRMs-IM2 are presented on real-world imbalanced relational datasets. In this chapter, the evaluation is performed on the Curtin University relational dataset presented in Chapter 3 for Computing and Commerce students. In addition, PRMs-IM2 is also evaluated on another dataset extracted from Curtin University database for first year Engineering students.

Degree	Unit	$\|Fail\|$	$\|Pass\|$
BCS	ST152	12	58
	FCS152	11	59
	IPE101	7	63
BCom	MGT100	159	1556
	MKT100	88	1627

Table 5.1: Summary of the data distribution of the BCS and BCom training datasets.

Degree	Unit	$\|Fail\|$	$\|Pass\|$
BCS	ST152	6	16
	FCS152	5	16
	IPE101	9	15
BCom	MGT100	30	216
	MKT100	14	232

Table 5.2: Summary of the data distribution of the BCS and BCom test sets.

5.4.1 Datasets

5.4.1.1 Computing and Commerce datasets

In this chapter, PRMs-IM2 is evaluated on the same datasets used in the previous chapter for undergraduate students of the Bachelor of Computer Science (BCS) and the Bachelor of Commerce (BCom), described in Section 4.4.1.

As in the previous chapter, for each of the BCom and BCS datasets, it is of interest to predict student performance in second semester units given student personal information and performance in the first semester. In these datasets, the 'Status' attribute of a semester II unit is a two-class imbalanced attribute with two classes: {'Pass', 'Fail'}, representing the majority and minority classes, respectively.

The data distribution of the training data is shown in Table 5.1, representing students enrolled in the period 1999-2005. In addition, Table 5.2 shows the data distribution of the test set, representing data of students enrolled in year 2006.

Semester	Unit Code	Unit Name	Prerequisite
I	Math1	Engineering Mathematics 1	-
	Mechanics1	Engineering Mechanics 1	-
II	Math2	Engineering Mathematics 2	Math1
	Systems1	Electrical Systems 1	-
	Materials1	Engineering Materials 1	-

Table 5.3: The curriculum of first year of BEng.

5.4.1.2 Engineering dataset

In this chapter, PRMs-IM2 is also evaluated on another dataset extracted from the Curtin University database representing the data of first year Engineering students. This dataset represents first undergraduate students of the Bachelor of Engineering Pre-Major (BEng). The Bachelor of Engineering Pre-Major is designed to prepare students for the second year of any of the engineering disciplines.

The units of BEng are shown in Table 5.3. Although the current BEng curriculum consists of eights units, only five units are considered in this dataset, as these are the only units that have remained unchanged for the period 2004-2007.

The structure of the database for the BEng is identical to that of the BCom and the BCS, which includes a set of tables and relationships representing students information and their performances in first year. To recap, the database is organized as follows:

- The *Personal_Info* table that consists of:
  - Age: {16-19, 20-29, 30-40},
  - Gender: {Male, Female},
  - Is_international: {Yes, No},
  - Is_English_home_language: {Yes, No}.
- The *Academic_Info* table, which includes:
  - Preference_no: takes values of: {1, 2, 3}, indicating the student's preferred course.
- Semester I units tables, each includes:
  - Grade: of values: {F, 5, 6, 7, 8, 9} representing the categories: {0-49, 50-59, 60-69, 70-79, 80-89, 90-100}.

Unit	$\|Pass\|$	$\|Fail\|$
Math2	1333	35
Material1	1350	18
Systems1	1336	32

Table 5.4: Data distribution of the BEng training dataset.

- Semester II units tables, including:
  - Status: {Pass, Fail}.

As for the BCom and BCS datasets, it is of interest to classify students performance in second semester units, which forms the problem of classifying multiple imbalanced attributes, as the majority of data belongs to the majority 'Pass' class compared to few samples belonging to the minority 'Fail' class. Table 5.4 shows the data distribution of the BEng training data for students enrolled in BEng in the period 2004-2007.

5.4.2 Experimental setup

Chapter 2 presented three naïve methods for classifying multiple attributes: independent, hierarchical and combined methods. In this section, PRMs-IM2 is evaluated in comparison to the independent and hierarchical approaches. The combined approach is not evaluated in this chapter as it represents a multi-class problem by considering the set of multiple attributes as one complex attribute and hence requires special multi-class algorithms. The multi-class problem will be discussed in the next chapter.

For evaluating each of these algorithms, 5-fold cross validation is used. The evaluation results are presented in terms of Receiver Operating Characteristics (ROC) curves and the Area Under ROC (AUC). For the ROC curves and the AUC results of the 5-fold cross validation, all the test samples of the five folds are sorted according to their probability estimation. Then, the ROC curves and AUC results are calculated.

For a set of multiple imbalanced attributes $\mathcal{Y} = (Y_1, Y_2, .., Y_n)$, the independent approach is evaluated using the PRMs-IM, as discussed in Chapter 4, as each attribute is classified independently. The approaches employed for constructing each PRMs-IM ensemble and for classifying new samples are carried out as explained in Section 4.3.

For evaluating the hierarchical approach, a hierarchy of classifiers is constructed such that a classifier is constructed for a given attribute Y_i and then a specialized classifier is constructed for Y_j for each class of Y_i. However, in order to minimize propagating the misclassification errors to lower levels of the hierarchy, the attributes classifiers are first arranged such that the most confident classifiers are at the top of the hierarchical structure. This is performed by ordering the classifiers based on their AUC values obtained for each attribute from the PRMs-IM evaluation. Thus, classifiers with higher AUCs are placed at the top of the hierarchical structure.

To recap from Chapter 2, consider the example presented earlier in Section 2.5.1 but for a relational data $\mathcal{S}$ that includes a set of attributes $\mathcal{X} = (X_1, X_2, ..., X_N)$ organized into a set of tables and relations. In addition, $\mathcal{S}$ includes three multiple imbalanced attributes $\mathcal{Y} = (Y_1, Y_2, Y_3)$ that are of interest to be classified, and each imbalanced attribute Y_i has two classes: A, B. The imbalanced attributes are ordered based on the AUC results obtained from the PRMs-IM evaluation as follows: (Y_1, Y_2, Y_3).

In the training phase, the top classifier $C(Y_1)$ for classifying Y_1 is trained on a subset extracted from $\mathcal{S}$ that includes the data of Y_1 and the corresponding records of $\mathcal{X}$. Consequently, two classifiers are constructed for classifying Y_2, the first classifier $C(Y_2)a$ is trained on a data subset that includes the data of Y_2 related to the records of $(Y_1 = A)$, and the other classifier $C(Y_2)b$ is trained on a data subset that includes the data of Y_2 related to the records of $(Y_1 = B)$. For classifying the third attribute Y_3, four classifiers are constructed, two classifiers $C(Y_3)a_1$ and $C(Y_3)b_1$ for the case of data of Y_3 related to data of $(Y_1 = A \ \& \ Y_2 = A)$ and $(Y_1 = A \ \& \ Y_2 = B)$, respectively. Similarly, two classifiers $C(Y_3)a_2$ and $C(Y_3)b_2$ are created for the case of data of Y_3 related to data of $(Y_1 = B \ \& \ Y_2 = A)$ and $(Y_1 = B \ \& \ Y_2 = B)$, respectively. An illustration of this process is shown in Fig. 5.6.

Considering the example shown in Fig. 5.6, new samples are classified in the hierarchical approach by first evaluating the test sample by the top classifier of the hierarchy $C(Y_1)$ for classifying the attribute Y_1. Accordingly, $C(Y_1)$ outputs the predicted class of Y_1. Based on the predicted class of Y_1, the second classifier for classifying the attribute Y_2 is evaluated. For example, if the predicted class of Y_1 is A, then the second classifier selected is that corresponding to the classifier trained on data related to class A of Y_1, i.e. classifier: $C(Y_2)a$. Consequently, the second selected classifier outputs the predicted class of attribute Y_2, and the process continues until the last attribute is classified.

Furthermore, to resolve the imbalanced class problem, each classifier in the hierarchical approach is constructed as a PRMs-IM. Therefore, the classifier of an attribute $C(Y_1)$ is constructed as a PRMs-IM trained on balanced subsets extracted from the training data

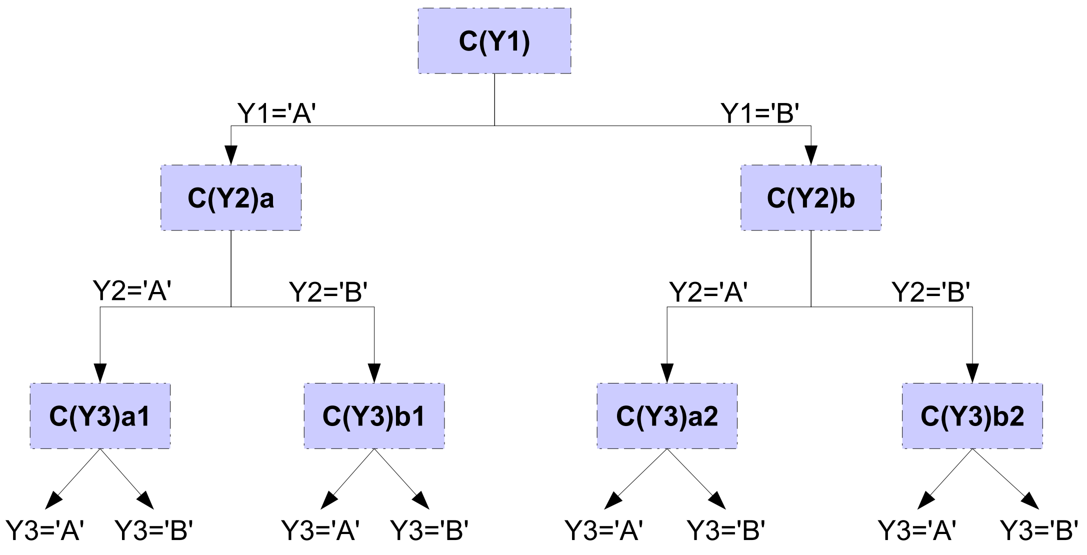


Figure 5.6: An illustration of the hierarchical process for classifying multiple attributes.

corresponding to Y_1.

5.4.3 Results

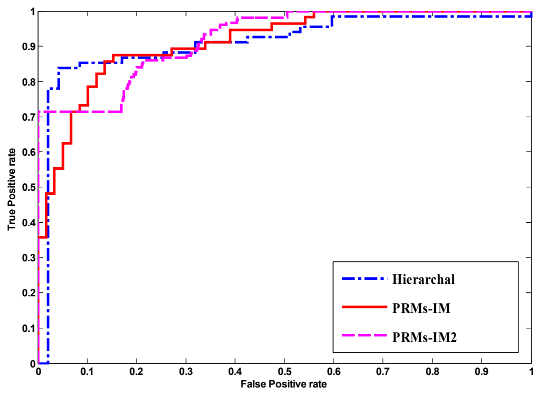
The AUC results of each algorithm are shown in Table 5.5, in which the best AUC result of each dataset is shown in bold. The AUC table also shows the average AUC for each algorithm over the datasets. The corresponding ROC curves are presented in Fig. 5.7 - Fig. 5.11.

The overall average AUC shows that the best performance is obtained by PRMs-IM2, then PRMs-IM followed by the hierarchical approach. The hierarchical approach obtained the least overall average AUC, though it was structured by placing the most accurate classifiers at the top of the structure. In a comparison between the independent and hierarchical approaches, the results show that among the thirteen experiments evaluated, the independent approach has outperformed the hierarchical approach in nine experiments and obtained equal AUC values in the remaining four experiments.

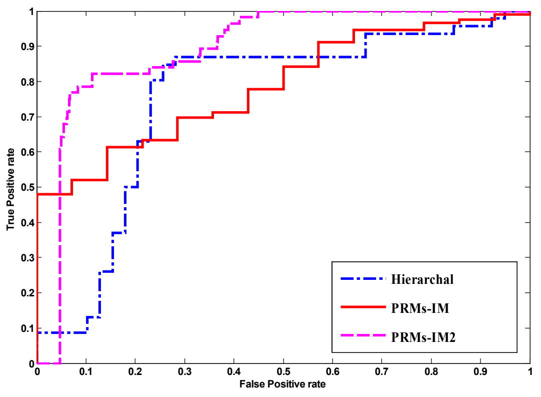
The lower performance of the hierarchical approach is caused by allowing the top classifiers to have a significant role in the classification of the lower attributes. Thus, the decision of the lower classifiers are controlled by those at the top, and hence if the top classifier misclassifies a sample, the lower classifier cannot correct the decision.

Validation	Dataset		Algorithm		
	Degree	Unit	Independent	Hierarchical	PRMs-IM2
Cross validation	BCom	MGT100	0.914	0.914	**0.922**
		MKT100	0.786	0.756	**0.893**
	BCS	ST152	0.839	0.811	**0.950**
		FCS152	0.901	0.897	**0.923**
		IPE101	**0.913**	**0.913**	0.892
	BEng	Math2	0.883	0.821	**0.965**
		Systems1	**0.945**	0.833	0.871
		Materials1	0.854	0.717	**0.941**
2006-Test set	BCom	MGT100	**0.921**	**0.921**	**0.921**
		MKT100	0.788	0.787	**0.840**
	BCS	ST152	0.875	0.785	**0.984**
		FCS152	0.927	0.887	**0.968**
		IPE101	0.954	0.954	**0.993**
	Average AUC		0.886	0.850	**0.931**

Table 5.5: The AUC results of the 5-fold Cross validation and 2006 testing set.

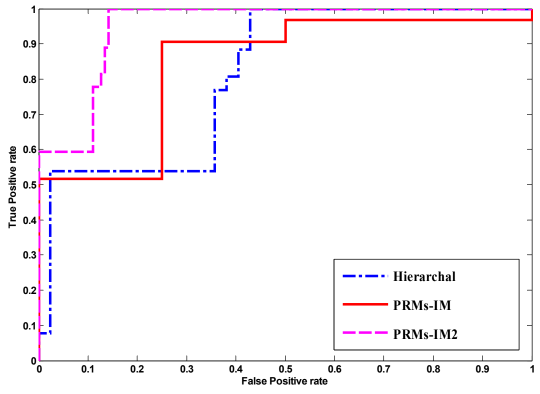


(a)

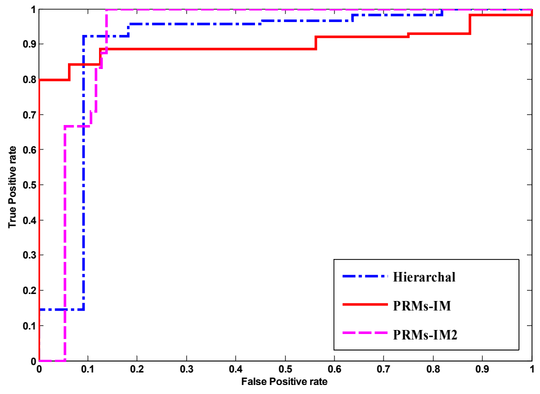


(b)

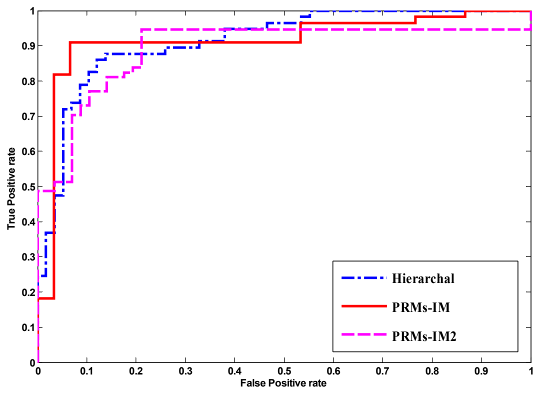
Figure 5.7: ROC curves of the BCom student relational database for (a) MGT100 (b) MKT100 units (5-fold cross validation).



(a)

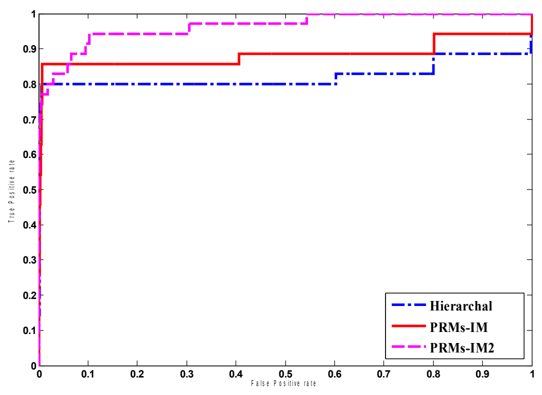


(b)

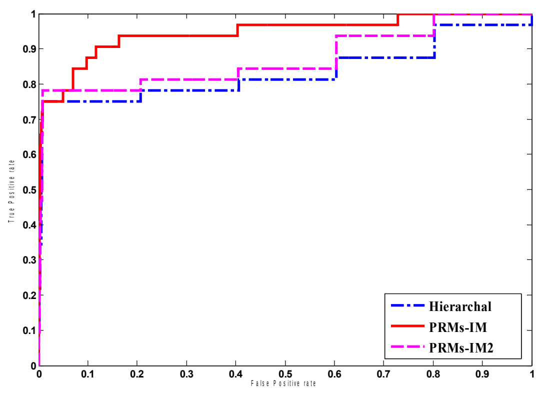


(c)

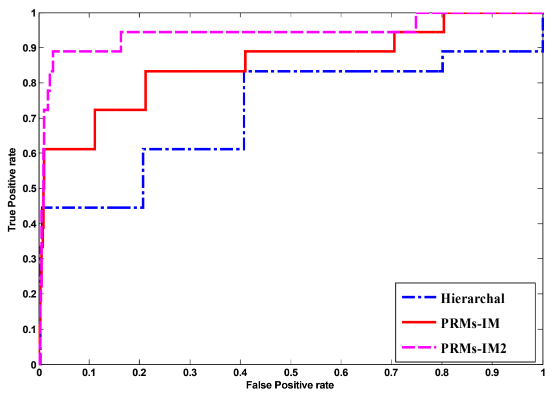
Figure 5.8: ROC curves of the BCS student relational database for (a) ST152 (b) FCS152 (c) IPE101 units (5-fold cross validation).



(a)

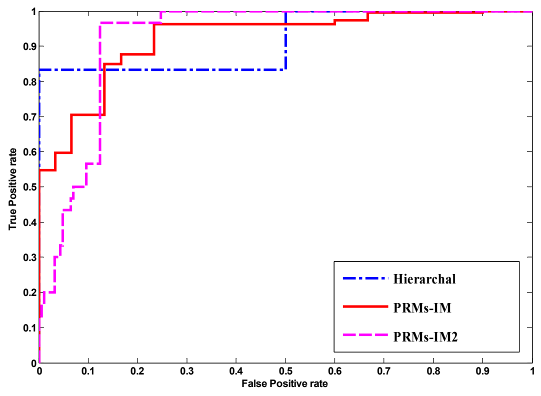


(b)

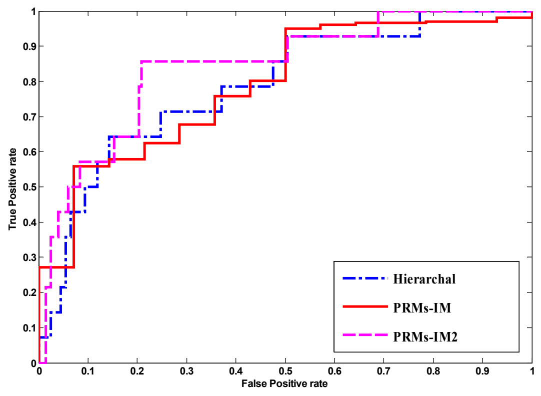


(c)

Figure 5.9: ROC curves of the Engineering student relational database for (a) Math2 (b) Systems1 (c) Materials1 units (5-fold cross validation).

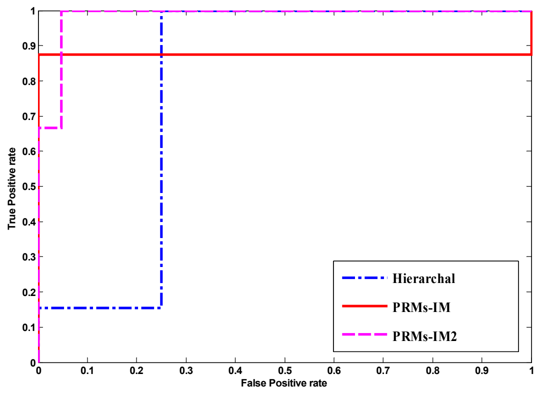


(a)

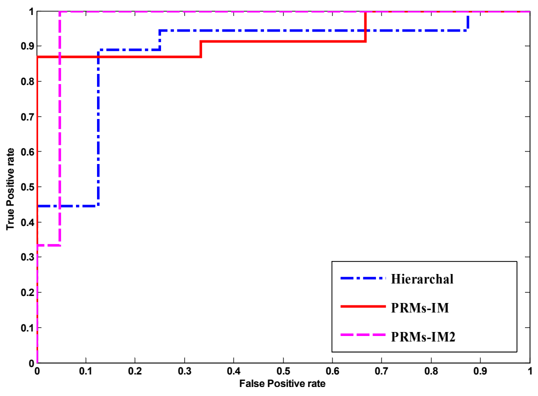


(b)

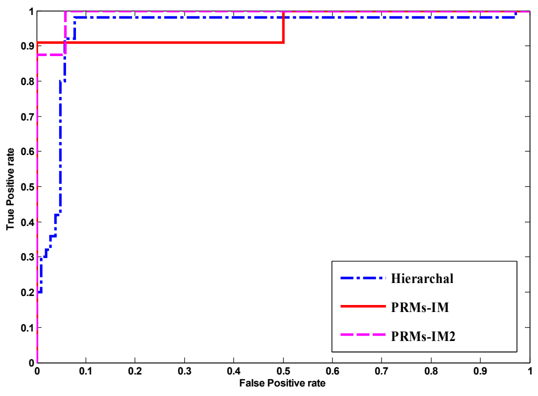
Figure 5.10: ROC curves of the BCom student relational database for (a) MGT100 (b) MKT100 units (2006-Testing Set).



(a)



(b)



(c)

Figure 5.11: ROC curves of the BCS student relational database for (a) ST152 (b) FCS152 (c) IPE101 units (2006-Testing Set).

As for the proposed approach (PRMs-IM2), the results show that PRMs-IM2 has generally achieved promising results in classifying the multiple imbalanced attributes. Among the thirteen experiments evaluated, PRMs-IM2 has outperformed the other classifiers in ten experiments and obtained equal AUC values to these for the independent and hierarchical approaches in the case of the BCom data for 'MGT100' unit in the test set. However, PRMs-IM2 was outperformed in two experiments, first by the independent method in the case of the BEng data for 'Systems1', and the second case by the independent and hierarchical approaches for the BCS data for 'IPE101'.

These AUC results confirm the benefits gained from the PRMs-IM2 framework for classifying multiple imbalanced attributes, which has resulted in the best overall average AUC result. The results of PRMs-IM2 also show the significance of learning from balanced subsets that includes all the imbalanced attributes. Thus, PRM models are learned from complete training sets, and hence can capture the different relationships between the attributes. In addition, the relational learning technique (PRM) offers an effective tool for capturing such interactions between the relational attributes, as discussed in Chapter 2.

This modelling of all attributes plays a critical role in prediction, as any prediction obtained for one of the imbalanced attributes can greatly improve the predictions of those related attributes. Modelling this interaction between the attributes in PRMs-IM2 could explain the improved performance over the other methods. In view of the fact that this interaction could not be achieved using the independent model, as each attribute is modelled independently and hence the value of one attribute cannot be used to reach any conclusion about the others. Furthermore, this interaction could not be achieved in the hierarchical method as well, as only the top attributes can participate in the classification of the lower ones but not vice versa. Moreover, though the combined approach has not been evaluated in this chapter, the combined approach considers the imbalanced attributes as one single complex attribute and thus cannot model the different interactions between the attributes to reach better conclusions.

In the classification task and, in particular, for classifying multiple attributes, it is important to take into consideration the number of classifiers required for training and inference. Therefore, this section also presents the number of classifiers required for training and inference for each algorithm, as shown in Table 5.6. The number of classifiers are presented in comparison to that obtained by PRMs-IM2. For each algorithm, the number of classifiers for a particular dataset is divided by the corresponding number of classifiers of PRMs-IM2. Algorithm with a normalized value greater than one indicates that the given algorithm requires more classifiers than PRMs-IM2.

(a)

Algorithm	Dataset (DS)			Average
	BCom	BCS	BEng	over DS
PRMs-IM	1.53	2.11	1.95	1.86
Hierarchical	2.32	5.56	1.28	3.05
PRMs-IM2	1.00	1.00	1.00	1.00

(b)

Algorithm	Dataset (DS)			Average
	BCom	BCS	BEng	over DS
PRMs-IM	1.53	2.11	1.95	1.86
Hierarchical	2.11	3.67	1.16	2.31
PRMs-IM2	1.00	1.00	1.00	1.00

Table 5.6: Normalized number of classifiers used for (a) Training (b) Inference.

The results in Table 5.6 show that PRMs-IM2 requires the least number of classifiers for both training and inference. For example, in the case of training, the number of classifiers for PRMs-IM and the hierarchical approaches are about twice and triple, respectively, that of PRMs-IM2. In the case of inference, the number of classifiers for PRMs-IM and the hierarchical approaches are about twice those of PRMs-IM2.

PRMs-IM2 requires the least number of classifiers because it constructs the ensemble subsets such that each subset includes all the imbalanced attributes and hence one classifier is required for each subset. Consider the case of two imbalanced attributes (Y_1, Y_2) that have balanced subsets (N_1, N_2), respectively, where $[N_1 \geq N_2]$. The number of classifiers required of PRMs-IM for this case is $(N_1 + N_2)$ for training and classifying new samples. On the otherhand, consider the case of the hierarchical approach of the order (Y_1, Y_2), and (n_1, n_2) balanced subsets of Y_2 corresponding to the two classes of Y_1. The number of classifiers required for training is $(N_1 + n_1 + n_2)$ and the number of classifiers required for inference is $(N_1 + \max(n_1, n_2))$. In contrast, PRMs-IM2 only requires N_1 classifiers for both training and inference.

5.5 Conclusion

This chapter has outlined a framework (PRMs-IM2) for classifying multiple imbalanced attributes in relational domains. PRMs-IM2 is proposed as an extension of PRMs-IM to model all the imbalanced attributes in a single model rather than building an independent model for each attribute as in PRMs-IM. The construction of the single model in PRMs-IM2 is achieved by employing the popular bagging ensemble approach. PRMs-IM2 was evaluated to classify a number of multiple imbalanced attributes in student relational data, particularly for undergraduate students of Computing, Commerce and Engineering. The evaluation results show that PRMs-IM2 not only generally perform better than the other approaches, but also requires the least number of models for training and inference. Although PRMs-IM and its extension PRMs-IM2 have effectively addressed the imbalanced class problem in relational domains, both frameworks are proposed for the special case of two-class classification. However, in several domains, many more different pattern classes are required to be classified rather than just the two-class case. The multi-class pattern classification in imbalanced data is discussed in the next chapter.

Chapter 6

Multi-Class Classification in Imbalanced Data

The preceding chapters discussed and presented different techniques for handling the imbalanced class problem in relational domains for the special two-class case. In particular, the previous chapter outlined a framework to handle the problem of modelling multiple imbalanced attributes in such domains. However, in several applications, it is of interest to assign to one class from many classes. Therefore, this chapter discusses learning from imbalanced multi-class data and proposes a new approach, named Multi-IM, to deal with the imbalanced multi-class problem. Multi-IM derives its fundamentals from the probabilistic relational technique (PRMs-IM), designed for learning from imbalanced relational data for the two-class problem. Multi-IM extends PRMs-IM to a generalized framework for multi-class imbalanced learning for both relational and non-relational domains. In this chapter, Multi-IM is evaluated on a number of imbalanced datasets of different domains. The evaluation results show that Multi-IM generally outperforms other algorithms and achieves high performance rates in predicting all the classes, including the minority classes.

6.1 Introduction

There is a rich literature of pattern classification techniques proposed for the two-class classification problem, such as support vector machines (SVM) (Hsu and Lin, 2002) that are designed specifically for binary functions. Along with binary classification, several algorithms have been also proposed for multi-class pattern classification. In multi-class pattern classification, it is often of interest to choose one class from more than two pattern classes, as in: text document classification (Stamatatos, 2008), bio-informatics (Al-Shahib *et al.*, 2005), and speech recognition (Even-Zohar and Roth, 2001).

Multi-class pattern classification can be simply defined as building a system that correctly maps the input of a given problem to an output of one of more than two classes. Many of the multi-class algorithms are based on decomposing the multi-class problem into a set of two-class classification problems, including the popular multi-class methods: One-Against-All (OAA) (Anand *et al.*, 1995) and One-Against-One (OAO) (Fürnkranz, 2002; Hastie and Tibshirani, 1998), discussed in Chapter 2.

Although multi-class methods have been shown to perform extremely well in classifying multiple pattern classes in different domains, these methods were mainly proposed for learning from assumed balanced datasets. However, many real-world domains have an imbalanced data distribution, where some classes of data have very few training examples when compared with the other classes.

This chapter discusses the imbalanced class problem in multi-class classification and presents a new framework to handle this problem. In a review of the imbalanced class problem, Chapter 4 introduced PRMs-IM method as a relational technique that can effectively handle the imbalanced problem in relational domains. However, PRMs-IM is designed specifically for the special two-class problem.

In terms of multi-class classification reviewed in Chapter 2, though the popular multi-class methods: One-Against-All (OAA) (Anand *et al.*, 1995) and One-Against-One (OAO) (Fürnkranz, 2002; Hastie and Tibshirani, 1998), have their strengths in multi-class classification as pointed out by several papers (Rifkin and Klautau, 2004; Lézoray and Cardot, 2008; Garcia-Pedrajas and Ortiz-Boyer, 2006), they have major limitations when dealing with the imbalanced problem (Ou and Murphey, 2007). Furthermore, the new method All-and-One (A&O) (Garcia-Pedrajas and Ortiz-Boyer, 2006) that combines the strengths of both OAA and OAO approaches and avoids the problems of each, has not been designed to address the imbalanced class problem.

The One-Against-Higher-Order (OAHO) approach (Murphey *et al.*, 2007), reviewed in Chapter 2, is proposed to handle the imbalanced problem for multi-class classification by building a hierarchy of classifiers based on the data distribution. The experiments conducted by Ou and Murphey (2007) show that OAHO tends to outperform the other multi-class methods in imbalanced multi-class learning. However, OAHO includes the challenge of arranging the classifiers in a good order to minimize propagating the classification errors to the next levels of the hierarchy, as mistakes made at the top level cannot be corrected by the following levels.

Based on this review, this chapter introduces a new approach (Multi-IM) to handle the challenges of learning from imbalanced multi-class domains. The proposed approach is based on extending PRMs-IM to handle multi-class classification by applying the concepts of A&O. Multi-IM applies the framework of A&O system by constructing both the OAA and OAO approaches, but constructs a classifier for each of these approaches as a PRMs-IM ensemble. Moreover, Multi-IM is proposed as a general framework to handle the imbalanced multi-class problem for relational and flat (non-relational) domains.

Multi-IM is evaluated on a number of imbalanced datasets obtained from the UCI machine learning database (Asuncion and Newman, 2007) and Curtin University. The experimental results show that the proposed approach achieves high performance rates in learning from imbalanced multi-class problems. Moreover, the results demonstrate the need for enriching multi-class pattern classification techniques with balancing tools to improve classification results.

The remainder of this chapter is organized as follows: in Section 6.2, the proposed approach is introduced. The dataset used for evaluations and the evaluation results of the approach are presented in Section 6.3. Finally, Section 6.4 concludes the chapter.

6.2 Methodology

This chapter introduces a new framework, named: Multi-IM, for multi-class classification in imbalanced domains. Multi-IM can be viewed as an extension of PRMs-IM to multi-class classification for relational and non relational domains. In Multi-IM, the balancing concept of PRMs-IM is employed to handle the imbalanced class problem and the All-and-One (A&O) approach is used for multi-class classification.

PRMs-IM was proposed in Chapter 4 as an ensemble approach for classifying two-class imbalanced attributes in relational domains. In the PRMs-IM approach, a set of independent PRM models are trained on balanced subsets extracted from the original imbalanced dataset. Then, the models are combined to obtain the final classification results.

The All-and-One (A&O) approach, reviewed in Chapter 2, was introduced to handle multi-class classification by training both OAO and OAA approches. The top two candidates of the OAA are then selected, and the final answer is obtained by using the corresponding OAO classifier of the two candidates.

In the proposed approach, the concepts of PRMs-IM and A&O are combined to build an ensemble of classifiers. The construction and combinations of the classifiers in Multi-IM

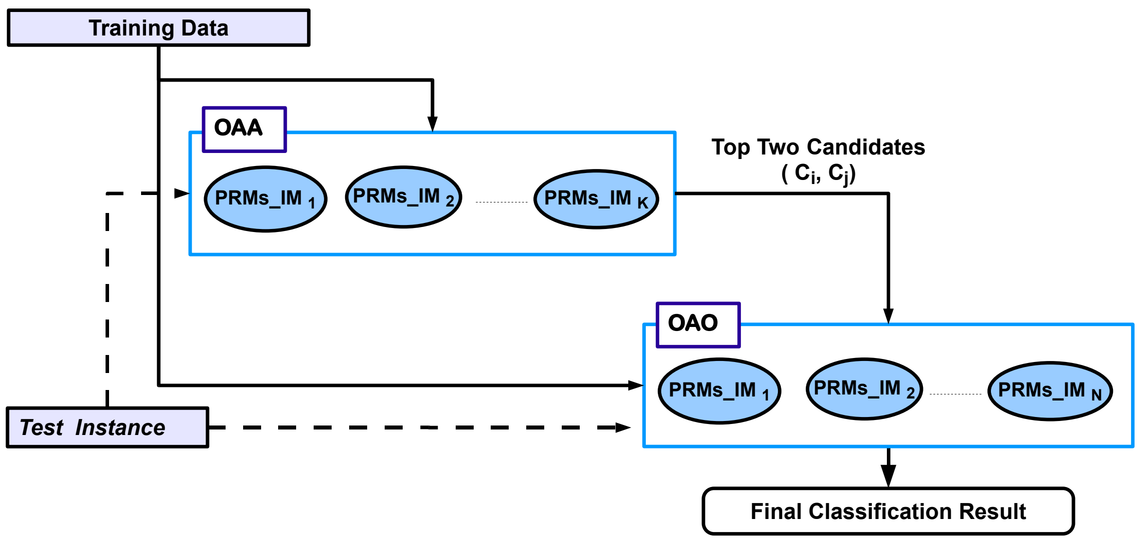


Figure 6.1: The training and inference procedures of Multi-IM approach.

uses the All-and-One (A&O) approach, with each classifier in the framework is trained as a PRMs-IM model.

In Multi-IM, OAA and OAO systems are first constructed by creating the required classifiers. Thus, creating a classifier for each class in the OAA system, and creating a classifier for each pair of classes in the OAO system. Each classifier in the OAA and OAO systems is built as a PRMs-IM ensemble.

To construct each of the OAA and OAO classifiers as a PRMs-IM ensemble, the training data of each classifier is partitioned into balanced subsets, in which each subset includes all the minority samples of the training data and a random selection of the majority samples. Then, an independent classifier is trained on each balanced subset. The components are then combined using the weighted voting strategy, following the same approach for PRMs-IM, discussed in Chapter 4.

For classifying new instances, the instance is first predicted by each of the classifiers of the OAA system. The outputs of the OAA classifiers are then combined using majority voting to find the top two candidates. Based on these top two candidates of the OAA system, the final classification result is obtained by using the corresponding binary OAO classifier of the two candidates. Fig. 6.1 shows the training and inference procedures of Multi-IM approach.

It is proposed to use Multi-IM as a general framework for relational and non-relational

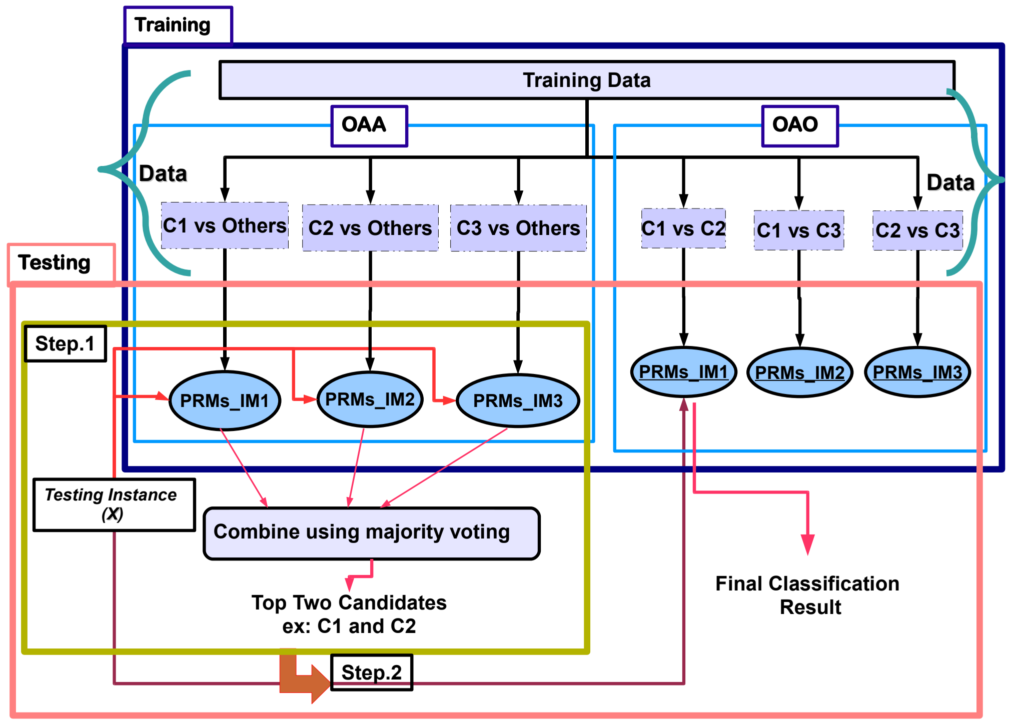


Figure 6.2: An illustration of Multi-IM approach for learning from imbalanced multi-class data.

domains. The idea behind this generalized framework is based on the belief that the balancing concept of PRMs-IM of creating balanced subsets can be employed effectively in both relational and flat domains. The major difference between the implementation on these domains would be the selection of the classifier that best fits the domain characteristics. For example, in relational domains, PRMs can be applied as the relational learning technique to learn from the balanced subsets, whereas Bayesian Networks (BNs), for example, can be applied for flat (non-relational) datasets.

To illustrate the Multi-IM approach, consider the example of a three class problem (C_1, C_2, C_3), which has an imbalanced data distribution. The Multi-IM approach for this problem is illustrated in Fig. 6.2. Multi-IM first builds OAA and OAO systems by constructing the required set of classifiers. For the OAA system, Multi-IM constructs three classifiers (OAA_1, OAA_2, OAA_3), one classifier for each class *vs* the other two. For classifier OAA_i, the corresponding training data S_i consists of all the examples of class (i) as positives and all the other examples as negatives. For OAO, Multi-IM builds three classifiers $(OAO_{(1,2)}, OAO_{(1,3)}, OAO_{(2,3)})$ for each pair of classes. The training data $S_{(i,j)}$ of classifier $OAO_{(i,j)}$, consists of the examples of class i as positives and the examples of class j as negatives.

To incorporate the imbalanced solution, the balancing concept of PRMs-IM is employed to create balanced subsets for each classifier in OAA and OAO. Therefore, the corresponding data D_i of a classifier F_i is further partitioned into balanced subsets based on the statistical distribution of D_i. Each subset consists of all the D_i minority samples and an equal random number of samples of the D_i majority class sampled randomly with replacement. Then, an independent classifier is trained on each balanced subset. The components are then combined using the weighted voting strategy. Thus, each classifier outputs the class with the highest weighted score as the predicted class.

By obtaining the classifiers of the OAA and OAO systems, a new test sample can be classified. For example, a test sample (x) is predicted by submitting (x) to each of the classifiers of the OAA system. Then, the results of the OAA system are combined to find the top two candidates (C_i, C_j). The test sample is then evaluated using the corresponding binary OAO classifier $OAO_{(i,j)}$ to find the final predicted result.

As in A&O approach, the main concern with Multi-IM is the large number of classifiers required in this approach. For example, for the K-class problem, $K(K-1)/2$ classifiers are required for OAA, K classifiers for OAA, in addition to the set of PRMs-IM ensemble classifiers, in which each depends on the statistical distribution of the corresponding data.

However, this number of classifiers can be reduced as suggested for A&O approach (Garcia-Pedrajas and Ortiz-Boyer, 2006), by constructing the OAO binary classifiers once they are required. Therefore, for a training data D, the OAA system is first constructed by training the required classifiers. Then, for a new test sample x, the OAA system is used to obtain the top two candidates (C_i, C_j). The corresponding OAO classifier $OAO(C_i, C_j)$ is then trained and used to predict the class of x. This could reduce the number of classifiers, especially for problem domains with a large number of classes.

6.3 Experimental results

In order to get a clear idea about the performance of the proposed model, three datasets from different domains are used, which are: Students, Glass, and Shuttle datasets. The first dataset (Students) is obtained from Curtin University and the other two (Glass and Shuttle) from the Machine Learning Data Repository of the University of California at Irvine (UCI) (Asuncion and Newman, 2007).

This chapter uses the imbalance ratio described by Ou and Murphey (2007), for the OAO

Phase	Degree	Dataset	No. Samples			Imbalance Ratio
			Fail	Average	Excellent	
Training	BCom	MGT100	159	1470	86	0.058
		MKT100	88	1559	68	0.044
	BCS	ST152	12	50	8	0.160
		FCS152	11	53	6	0.113
		IPE101	7	61	2	0.033
Testing	BCom	MGT100	30	185	31	
		MKT100	14	200	32	
	BCS	ST152	6	12	4	
		FCS152	5	11	5	
		IPE101	9	13	2	

Table 6.1: Training and testing data distribution of students datasets.

method, to measure the class imbalance of each dataset. The imbalance ratio is defined as the ratio of the size of the biggest class to the size of the smallest class. The imbalance ratio has the range [0-1], with '1' indicating well balanced data, and '0' indicating that the data is most imbalanced. The following sections describes the characteristics of each dataset.

6.3.1 Datasets

6.3.1.1 Students Database

In this chapter, Multi-IM is evaluated on the same datasets used in PRMs-IM for undergraduate students of the Bachelor of Computer Science (BCS) and the Bachelor of Commerce (BCom), described in Section 4.4.1, but for multi-class pattern classification.

To evaluate the performance of Multi-IM on this relational imbalanced multi-class data, the proposed model is used to predict the performances of the BCom and BCS students in semester II units given their personal and academic information, and the grades achieved in semester I. Table 6.1 shows the distribution of the training datasets for students enrolled in the period 1999-2005. The imbalance ratios of the training datasets shown in Table 6.1 indicate that the datasets are highly imbalanced. The data of year 2006 is kept separately as independent testing set to be used for evaluating the model trained on the data from 1999-2005.

Dataset	No. Samples							Imbalance
	C1	C2	C3	C4	C5	C6	C7	Ratio
Training:								
Glass	70	76	17	13	9	29	-	0.044
Shuttle	34,108	37	132	6748	2548	6	11	0.00018
Testing:								
Shuttle	11,478	13	39	2155	809	4	2	

Table 6.2: Training and Testing data distribution of Glass and Shuttle datasets.

6.3.1.2 UCI Datasets

The proposed approach is also evaluated using the Glass and Shuttle datasets obtained from the UCI machine learning database. These datasets have been selected because they represent highly imbalanced datasets with different numbers of pattern classes.

The Glass and Shuttle datasets have been widely used in different experiments by several papers discussing the problem of multi-class pattern classification (Ou and Murphey, 2007; Murphey *et al.*, 2007; Garcia-Pedrajas and Ortiz-Boyer, 2006). These experiments have shown that the popular multi-class algorithms, such as OAO and OAA, generally obtain high prediction rates for the majority classes but perform poorly on predicting the minority ones. Table 6.2 shows the distribution of the Glass and Shuttle datasets.

6.3.2 Experimental setup

The results of Multi-IM are presented in comparison to the multi-class methods: OAA, OAO, OAHO, Stacking, and A&O, discussed in Chapter 2. For each dataset, 5-fold cross validation is used to obtain the performance of each algorithm. For the Students and Shuttle datasets, the algorithms are also evaluated by using the independent testing sets shown in Table 6.1 and 6.2.

To build the Multi-IM ensembles as a set of components (classifiers) trained on balanced subsets, a classifier must be selected based on the dataset characteristics. Therefore, the PRM is used as the classifier model for the relational dataset (Students Database), because of the relational format of the experimental data and the effectiveness of the PRM on learning from relational data. For the other datasets (Glass, Shuttle), the Naïve Bayes classifier is used, due to the flat characteristics of the datasets.

In evaluating the OAO approach, the majority voting scheme is used to combine the outputs of the binary classifiers. Therefore, each classifier in OAO outputs the predicted class of the test sample. Then, the test sample is assigned to the class with the highest number of votes.

In the evaluation of the OAA approach, the outputs of the binary classifiers are combined using probability outputs of the classifiers. Therefore, for each class i, the corresponding classifier f_i, that is trained on classifying class i against others, outputs the probability score of class i. Then, the test sample is assigned to the class with the highest probability score. The decision function of the OAA approach is defined as:

$$F(x) = \underset{i=1,\ldots,K}{\operatorname{argmax}} f_i(x) \tag{6.1}$$

where K is the number of classes, $f_i(x)$ is the probability score assigned to class i by the OAA classifier for classifying class i against all the other classes.

In OAHO modelling, the classes are ordered in descending order based on the size of the data in each class. Therefore, the class with the largest number of samples is at the top of the list. This order is applied in order to minimize the effect of the imbalanced class problem as suggested by Ou and Murphey (2007). By this order, the minority classes are used together as one class against the majority class, and hence, the effect of the imbalanced data distribution is reduced.

In stacking modelling, there are two approaches to generate the new training set for stacking, either by considering the probability estimations of the different classifiers or by using the predicted output values of the classifiers to be fed to the meta classifier. The experiments conducted by Lézoray and Cardot (2008) have shown that the latter approach produces better results than the former.

Therefore, in the evaluation of the stacking approach in this chapter, the predicted outputs of the OAO approach are used to generate the new training set for the stacking approach. The new training set is defined using the outputs of the binary classifiers as: $D^{stacking} = (\widehat{x_i}, y_i)$, in which y_i is the original class of the testing sample and $\widehat{x_i}$ is the set of the predictions of the binary OAO classifiers for the corresponding testing sample. $D^{stacking}$ is then used to train a Meta Classifier to predict the final class of the test sample. In this evaluation, a decision tree (C4.5) (Quinlan, 1993) is employed as the Meta Classifier.

In the A&O approach, the OAA classifiers are combined using the same approach outlined above for the OAA approach using the probability outputs and Equation 6.1. However, in this approach the decision function is used to get the first and second output classes of the OAA model rather than only the first output class. Then, the corresponding OAO model of the two outputs is used to get the final prediction of the A&O approach.

In this chapter, the evaluation of each algorithm is performed twice. The first evaluation is performed by applying the multi-class classification algorithms to the imbalanced dataset. The second evaluation is performed by embedding the balancing concept of PRMs-IM into multi-class classification algorithms. Thus, training the algorithms on balanced subsets of the imbalanced data. This balancing evaluation is performed to get a clear idea as to which degree the classification performance is improved/worsened by enriching the classification method with a balancing tool.

For example, in evaluating the OAA algorithm for the three-class problem, for the first evaluation, three binary classifiers (C_1, C_2, C_3) are constructed, in which classifier C_i is trained using all the examples of class i as the positive examples and all the other examples of the other classes as the negative examples, and then combining the outputs of the classifiers.

The second evaluation of the OAA approach is more concerned with balancing the data using the concept of PRMs-IM. Therefore, each classifier C_i is built as an ensemble of classifiers trained on balanced subsets. Thus, the data of a classifier C_i is divided into balanced subsets, where each subset includes an equal number of positive and negative samples. An independent classifier is then trained on each balanced subset and the classifiers are combined using the weighted voting strategy to get the output of C_i. The classifier C_i is then combined with the other binary classifiers of the OAA to solve the original classification problem.

6.3.3 Results

In order to evaluate the performance of the proposed model, the Receiver Operating Characteristics (ROC) curve and the Area Under Curve (AUC) (Fawcett, 2006) are used. These measures have been used by others to analyse the performance of imbalanced classification algorithms.

However, as the objective is to evaluate multi-class classification algorithms in this chapter, the multi-class AUC method proposed by Provost and Domingos (2000) is employed. In this approach, a separate AUC is calculated for each class, where the AUC for class i is calculated by considering all the examples belonging to class i as positives and the examples of all other classes as negatives. Then, the total AUC is calculated as the summation of the AUCs weighted by the class prior probability, as follows:

$$AUC_{total} = \sum_{c_i \in C} AUC(c_i) * p(c_i) \tag{6.2}$$

where, $AUC(c_i)$ is the AUC for class i, and $p(c_i)$ is the prior probability of class i.

Table 6.3 shows the total AUC results obtained on the datasets. The AUC table shows two sets of experimental results: Experiment-A: the results obtained by applying the classification methods as described in Chapter 2, Experiment-B: the results on the same datasets after introducing the balancing concept into each of the classification methods. A classification method M in Experiment-A is denoted as $\overline{M}$ in the balancing Experiment-B. The best result for each dataset is shown in bold. The table also shows the average performance of each algorithm over the datasets.

The overall average AUC shows that the best performance is obtained by Multi-IM, and the least is achieved by the stacking algorithm. Table 6.4 shows the order of the algorithms based on their average AUC performance, in which the best performed algorithm is listed at the top of the list.

The results presented in Table 6.4 show that majority of algorithms have shown improved performances when combined with the balancing concept of PRMs-IM. For the five algorithms evaluated, four algorithms have shown improved performance when trained on balanced subsets. These results confirm the necessity of enriching multi-class techniques with balancing tools to improve the classification results in imbalanced domains.

Interestingly, both the Stacking and $\overline{Stacking}$ approaches have achieved the least average AUC performance over the datasets. In terms of the individual experiments, Stacking has obtained relatively similar results to those of OAO and OAA and did not show significant improvement. However, $\overline{OAA}$ and $\overline{OAO}$ obtained improved performance over that of $\overline{Stacking}$.

The poor performance of the Stacking approach is caused by the imbalanced data distribution of the new Stacking training set trained by the Meta Classifier. In the new training set, the minority class still has a low number of samples in comparison to the majority class, which results in poor prediction over the minority samples. However, the Stacking approach gained slightly a better performance in the $\overline{Stacking}$ after using the

Dataset	Classification Algorithm									
	Experiment-A					Experiment-B				
	OAA	OAO	Stacking	OAHO	A&O	$\overline{OAA}$	$\overline{OAO}$	$\overline{Stacking}$	$\overline{OAHO}$	Multi-IM
Glass *	0.807	0.779	0.748	0.801	0.844	0.828	0.805	0.759	0.816	**0.860**
Shuttle *	0.974	0.879	0.992	0.959	**0.997**	0.977	0.878	0.989	0.980	0.965
Shuttle **	0.947	0.941	0.956	0.985	0.981	0.983	0.986	0.987	0.897	**0.993**
MGT100 *	0.733	0.718	0.608	0.727	0.740	0.840	0.764	0.638	0.840	**0.898**
MGT100 **	0.755	0.707	0.612	0.750	0.772	0.882	0.795	0.699	0.879	**0.895**
MKT100 *	0.770	0.778	0.687	0.756	0.770	0.771	**0.787**	0.664	0.765	0.786
MKT100 **	0.780	0.781	0.671	0.766	0.781	0.780	0.778	0.682	0.780	**0.789**
ST152 *	0.793	0.757	0.738	0.786	0.801	0.783	0.824	0.791	0.790	**0.843**
ST152 **	0.786	0.757	0.739	0.790	0.802	0.784	0.826	0.807	0.786	**0.845**
FCS152 *	0.733	0.665	0.645	0.725	0.766	0.746	0.612	0.615	0.730	**0.904**
FCS152 **	0.733	0.665	0.645	0.725	0.766	0.746	0.612	0.615	0.730	**0.904**
IPE101 *	0.805	0.749	0.762	0.774	0.805	**0.886**	0.826	0.787	0.781	0.883
IPE101 **	0.810	0.729	0.741	0.786	0.811	0.874	0.839	0.756	0.768	**0.897**
Average over datasets	0.802	0.762	0.734	0.795	0.818	0.837	0.795	0.753	0.811	**0.882**

Table 6.3: Summary of the total AUC results obtained on datasets. Experiment-A shows the results of the classification methods. Experiment-B shows the results after introducing the balancing concept into each of the classification methods. * : denotes the results obtained from the 5-fold cross validation on the training set, ** : denotes the results from the independent testing set.

Order	Algorithm	Average AUC performance
1	Multi-IM	0.882
2	$\overline{OAA}$	0.837
3	A&O	0.818
4	$\overline{OAHO}$	0.811
5	OAA	0.802
6	$\overline{OAO}$ and OAHO	0.795
7	OAO	0.762
8	$\overline{Stacking}$	0.753
9	Stacking	0.734

Table 6.4: The order of the algorithms based on their average AUC performance.

balancing concept.

The results show that A&O has obtained the third best average performance, after Multi-IM and $\overline{OAA}$. This shows that though the multi-class approach A&O is not designed for imbalanced problems, the A&O approach outperforms most of the evaluated algorithms, including those using the balanced approach. In a comparison between the results of the A&O approach and those of A&O components (OAA and OAO), the A&O approach has outperformed both the OAO and OAA in all of the thirteen experiments shown in Table 6.3. These findings demonstrate the benefits gained from the A&O framework that combines the strengths of both OAA and OAO and avoids the problems of each.

The evaluation results of OAHO shows that $\overline{OAHO}$ performs better than OAHO. Nevertheless, the experiments show that the performance of $\overline{OAHO}$ is still hindered by the misclassification rates of the classifiers in the hierarchical strategy. For example, consider the case of a hierarchy of three classifiers (A, B, C) with the corresponding misclassification rates (20%, 10%, 20%), respectively. Each classifer has a low error rate by itself, but when it is used in the hierarchy with the order (A, B, C), this means that the misclassified examples (20%) of the first classifier $(A, \text{not } A)$ are propagated to the next level and hence again misclassified by the classifiers (B, C), which adds extra error to the examples missclassified into classes B and C.

An interesting observation in the average AUC performance is that both $\overline{OAO}$ and OAHO obtained the same performance. This outcome could be a result of using the OAHO framework that trains the small classes together against the majority class, and then trains the small classes against each other. Therefore, in OAHO, the classes are by some means trained against each other but by maintaining as much balanced distribution as possible. Similarly, in $\overline{OAO}$, classes are trained against each other but on balanced distributions for each classifier.

Another interesting observation for the average AUC performance is that $\overline{OAA}$ obtained the second best performance. This outcome again confirms the effectiveness of the OAA approach in multi-class classification, as pointed out by Rifkin and Klautau (2004). Furthermore, this outcome also reinforces the concept of introducing balancing into the multi-class algorithms to handle the problem of multi-class classification in imbalanced domains.

As for the proposed approach (Multi-IM), it can be viewed as a framework that first evaluates OAA on balanced subsets, i.e. using $\overline{OAA}$ to obtain the top two candidates (i, j), then, it evaluates the corresponding binary classifier of the top candidates $(\overline{OAO(i, j)})$.

The results presented in Table 6.4 show that Multi-IM has obtained the best average AUC performance. Moreover, among the thirteen experiments evaluated, shown in Table 6.3, Multi-IM achieved the best results in ten experiments, and obtained relatively less performance than the best performance in the remaining three experiments.

The results for Multi-IM confirms the effectiveness of using the $\overline{OAA}$ approach, which has obtained superior results as the second best performing algorithm after Multi-IM. The improved performance of Multi-IM over its component $\overline{OAA}$ refers to using the independent binary classifiers of $\overline{OAO}$ to classify between the two top candidates, which are usually highly accurate by their own, as pointed out by Garcia-Pedrajas and Ortiz-Boyer (2006).

The improved performance of Multi-IM over its basic A&O framework has been achieved as a result of introducing the balancing concept of PRMs-IM into Multi-IM. Therefore, each classifier can, in the Multi-IM framework, effectively classify new samples without being hindered by the imbalanced class problem. The experiments conducted on the relational and non-relational domains also show the effectiveness of the proposed approach for both domains. The major difference between the two domains is in selecting the appropriate classifier for each domain.

6.4 Conclusion

In this chapter, the research focused on two main challenges in pattern recognition: the imbalanced class problem and multi-class classification. The research outlined and evaluated the different strategies developed to solve each of these two challenges on its own, and those strategies proposed to handle both challenges. Based on this evaluation, a framework is introduced to handle these challenges simultaneously. The proposed approach (Multi-IM) is based on a relational technique designed for the binary imbalanced problem (PRMs-IM). However, Multi-IM extends PRMs-IM to a generalized framework for relational and non-relational domains. The proposed approach was applied to a number of imbalanced datasets from different domains. The results of Multi-IM improved generally over the other strategies evaluated. Furthermore, the evaluation results confirm the concept of Multi-IM for the design of special multi-class algorithms for imbalanced domains.

Chapter 7

Conclusion

This thesis has investigated the problem of mining imbalanced relational domains because of its potential in a wide range of real-world applications. In this investigation, the need for additional research for learning from such domains became evident and the need was identified to improve Probabilistic Relational Models (PRMs) for this task.

Therefore, the aim of this thesis has been to explore this need, along with the construction of a number of models for mining imbalanced relational domains. This has been achieved with extended versions of PRMs to handle the learning from imbalanced relational domains.

In Chapter 3, an application of PRMs for mining relational domains was presented. The applications demonstrated the ability of PRMs for mining a real-world university database, and particularly, the data of first year undergraduate students. The learned PRM models revealed the important dependencies in this domain and have significantly pointed to a number of recommendations to further improve student performance in first year. Furthermore, a data mining visual tool was presented to demonstrate the mining outcomes of the learned PRM models. The tool spatially illustrated the dependencies found by the PRM and aids the interpretation of the outcomes of the PRM learned models for non-PRMs expert.

In Chapter 4, an investigation into the efficiency of PRMs in mining imbalanced relational domains was presented. The investigation showed that PRMs performance is hindered by the imbalanced class problem in a similar way to the attribute-based method it is based on: the Bayesian Network (BN). An extension of PRMs for handling the imbalanced class problem in relational domains was introduced. The new model, PRMs-IM, utilizes the effectiveness of PRMs in relational learning and enriches it with ensemble learning. The ensemble learning employed in PRMs-IM focused on accurately classifying the rare cases that are usually of high importance. Experimentation revealed that the performance of PRMs-IM improved significantly over the standard techniques: PRMs and BN, and obtained comparable results with the special algorithms of imbalanced relational learning.

Chapter 5 continued the work into the analysis of mining multiple imbalanced attributes in relational domains, where several attributes are of interest to be classified. The work showed the significance of this problem in different domains and proposed a model, PRMs-IM2. The new model extends the work presented in Chapter 4 by employing the bagging approach. This approach models all the imbalanced attributes in a single model rather than building an independent model for each attribute as in PRMs-IM. Experimentation of PRMs-IM2 on a student database was presented to infer students' performance in second semester units of the first year. The evaluation results demonstrated the success of PRMs-IM2 in inferring student performances, and showed that the proposed approach requires the least number of models for training and inference than the other classical techniques.

Finally, Chapter 6 investigated the problem of multi-class imbalanced classification. The investigation compared and evaluated several well-known methods in this field. Based on this study, an extension of PRMs-IM is introduced for multi-class imbalanced classification. The new model (Multi-IM) utilizes the benefits of the well-known methods in this field to target the imbalanced class problem in multi-class classification. Furthermore, the model was proposed as a generalized model that can be employed for flat and relational domains. Experimentation on imbalanced datasets from both domains demonstrated the efficiency of the proposed model and supported the concept of enriching the multi-class algorithms with tools to handle the imbalanced class problem.

7.1 Future directions

Several interesting problems related to this research could be further investigated for future work. First of all, in this thesis, three models were presented for mining imbalanced relational domains. The first model targeted the problem of classifying a single binary imbalanced attribute in relational domains, while the other two models extended the first one to handle the classification of several binary imbalanced attributes, and the classification a single multi-class imbalanced attribute. These models outline another line for future work to build a model that targets all the problems handled by each model separately. This denotes constructing a model that can be employed for classifying several imbalanced attributes, along with handling the multi-class classification. This goal could be achieved by exploring the different techniques employed in the three proposed approaches and investigating their strengths and weaknesses. Such a model would certainly play a vital role in mining different domains without being constrained by some contexts.

Another interesting direction for future work is exploring the Probabilistic Relational

Models with Class Hierarchies (PRMs-CH) (Getoor *et al.*, 2000) with regards to modelling imbalanced domains. PRMs-CH extend PRMs by introducing the theory of class hierarchies via inheritance and specialization concepts. Therefore, constructing subclasses that can have a specialized probabilistic model for some instances of the parent class. This approach mainly targets the problem in standard PRMs of having all elements of the same class to use the same model, and thus can not have different dependencies for different types of instances. For example, for a university database holding the data of different courses, the set of parents affecting the performance of a Computing student are the same as those affecting an Engineering student, because the performance attribute is not specific for each course. Therefore, PRMs-CH offers a general mechanism to define the dependency model of the parent class as well as the dependency model for the subclasses. Such hierarchical modelling would certainly have an impact on the imbalanced class problem, and hence a further exploration in this field would be a interesting challenge.

Finally, in this thesis, the proposed approaches considered learning from static databases, where the entire data record represents a single time frame. On the other hand, data can be often organized as ordered data streams with temporal indices. For example, the university datasets presented in this thesis could be organized into a set of databases, where each database represents the students data for an academic year. This temporal modelling over years could enhance the predictability of the learned models, and thus can infer about a given student in a specific year, given the performances in the other years.

This indicates an interesting opportunity for exploring temporal mining with regards to imbalanced relational domains. A possible approach for achieving this goal would be through extending the learning technique of PRMs to accommodate temporal learning, and then investigating the imbalanced class problem in such models. Extending PRMs for temporal learning could be investigated in a similar way to how Dynamic Bayesian Networks extend Bayesian Networks. Although such work may lead to complex models, the resulting models would play a critical role in mining different real-world applications, where temporal interdependencies need to be considered.

Bibliography

Al-Shahib, A., Breitling, R., and Gilbert, D. (2005). Feature selection and the class imbalance problem in predicting protein function from sequence. *Applied Bioinformatics*, **4**(3), 195–203.

Alexandre, L. A., Campilho, A. C., and Kamel, M. S. (2000). Combining independent and unbiased classifiers using weighted average. In *Proc. International Conference on Pattern Recognition (ICPR)*, volume 2, pages 495–498, Los Alamitos, CA, USA. IEEE Computer Society.

Alexandre, L. A., Campilho, A. C., and Kamel, M. (2001). On combining classifiers using sum and product rules. *Pattern Recognition Letters*, **22**(12), 1283–1289.

Alkoot, F. M. and Kittler, J. (1999). Experimental evaluation of expert fusion strategies. *Pattern Recognition Letters*, **20**(11-13), 1361–1369.

Allwein, E. L., Schapire, R. E., and Singer, Y. (2001). Reducing multiclass to binary: a unifying approach for margin classifiers. *J. Mach. Learn. Res.*, **1**, 113–141.

Anand, R., Mehrotra, K., Mohan, C., and Ranka, S. (1995). Efficient classification for multiclass problems using modular neural networks. *IEEE Transactions on Neural Networks*, **6**(1), 117–124.

Astin, A. (1993). *What Matters in College: Four Critical Years Revisited.* San Francisco: Jossey-Bass.

Asuncion, A. and Newman, D. (2007). UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. [Online]http://www.ics.uci.edu/ mlearn/MLRepository.html.

Barandela, R., Valdovinos, R. M., and Sánchez, J. S. (2003a). New applications of ensembles of classifiers. *Pattern Analysis & Applications*, **6**(3), 245–256.

Barandela, R., Sánchez, J. S., García, V., and Rangel, E. (2003b). Strategies for learning in class imbalance problems. *Pattern Recognition*, **36**(3), 849–851.

Barbosa, L. and Freire, J. (2007). Combining classifiers to identify online databases. In *Proc. international conference on World Wide Web*, pages 431–440, New York, NY, USA. ACM.

Batista, G. E., Prati, R. C., and Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, **6**(1), 20–29.

Bennett, R. (2003). Determinants of undergraduate student drop out rates in a university business studies department. *Journal of Further and Higher Education*, **27**(2), 123–141.

Blockeel, H. and Raedt, L. D. (1998). Top-down induction of first-order logical decision trees. *Artificial Intelligence*, **101**(1-2), 285–297.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, **24**(2), 123–140.

Budny, D., Bjedov, G., and LeBold, W. (1997). Assessment of the impact of the freshman engineering courses. In *Proc. Frontiers in Education Conference (FIE)*, pages 1100–1106.

Burez, J. and Van den Poel, D. (2009). Handling class imbalance in customer churn prediction. *Expert Systems with Applications*, **36**(3), 4626–4636.

Byrne, P. and Lyons, G. (2001). The effect of student attributes on success in programming. *ACM SIGCSE Bulletin*, **33**(3), 49–52.

Chawla, N. V., Japkowicz, N., and Kotcz, A. (2004). Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, **6**(1), 1–6.

Chickering, D. M. (1996). Learning Bayesian networks is NP-Complete. In D. Fisher and H. J. Lenz, editors, *Learning from Data: Artificial Intelligence and Statistics*, pages 121–130. Springer-Verlag.

Cortes, C. and Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, **20**(3), 273–297.

Cowell, R., Dawid, A., and Spiegelhalter, D. (1993). Sequential model criticism in probabilistic expert systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**(3), 209–219.

Crammer, K. and Singer, Y. (2000). On the learnability and design of output codes for multiclass problems. In *Proc. of the Thirteenth Annual Conference on Computational Learning Theory*, pages 35–46, San Francisco, CA, USA.

Czyz, J., Kittler, J., and Vandendorpe, L. (2004). Multiple classifier combination for face-based identity verification. *Pattern Recognition*, **37**(7), 1459–1469.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Proc. First International Workshop on Multiple Classifier Systems*, pages 1–15, London, UK. Springer-Verlag.

Dietterich, T. G. and Bakiri, G. (1995). Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research*, **2**, 263– 286.

Druzdze, M. J. and Glymour, C. (1994). Application of the TETRAD II program to the study of student retention in U.S. colleges. In *KDD Workshop*, pages 419–430.

Duin, R. P. W. (2002). The combining classifier: To train or not to train? In *Proc. International Conference on Pattern Recognition (ICPR)*, pages 765–770, Los Alamitos, CA, USA. IEEE Computer Society.

Even-Zohar, Y. and Roth, D. (2001). A sequential model for multi-class classification. In *Proc. Conference on Empirical Methods in Natural Language Processing*, pages 10–19.

Ezawa, K. J., Singh, M., and Norton, S. W. (1996). Learning goal oriented Bayesian networks for telecommunications risk management. In *Proc. Thirteenth International Conference on Machine Learning*, pages 139–147. Morgan Kaufmann.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, **27**(8), 861–874.

Fawcett, T. and Provost, F. (1997). Adaptive Fraud Detection. *Data Mining and Knowledge Discovery*, **1**(3), 291–316.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Proc. International Conference on Machine Learning (ICML)*, pages 148–156, San Francisco. Morgan Kaufman.

Fürnkranz, J. (2002). Pairwise Classification as an Ensemble Technique. In *Proc. of the 13th European Conference on Machine Learning*, pages 97–110, London, UK. Springer-Verlag.

García, P., Amandi, A., Schiaffino, S., and Campo, M. (2007). Evaluating Bayesian networks' precision for detecting students' learning styles. *Computers & Education*, **49**(3), 794–808.

Garcia-Pedrajas, N. and Ortiz-Boyer, D. (2006). Improving multiclass pattern recognition by the combination of two strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **28**(6), 1001–1006.

Getoor, L. (2002). *Learning Statistical Models from Relational Data*. Ph.D. thesis, Stanford University.

Getoor, L., Koller, D., and Friedman, N. (2000). From instances to classes in probabilistic relational models. In *Proc. ICML-E000 Workshop on Attribute- Value and Relational Learning: Crossing the Boundaries*, pages 25–34.

Getoor, L., Friedman, N., Koller, D., and Taskar, B. (2001a). Learning probabilistic models of relational structure. In *Proc. International Conference on Machine Learning*, pages 170–177, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Getoor, L., Friedman, N., Koller, D., and Taskar, B. (2001b). Learning probabilistic models of relational structure. In *Proc. the Eighteenth International Conference on Machine Learning*, pages 170–177, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Getoor, L., Segal, E., Taskar, B., and Koller, D. (2001c). Probabilistic models of text and link structure for hypertext classification. In *Proc. International Joint Conference on Artificial intelligence Workshop on Text Learning: Beyond Supervision*, pages 24–29.

Getoor, L., Taskar, B., and Koller, D. (2001d). Using probabilistic models for selectivity estimation. In *Proc. ACM SIGMOD International Conference on Management of Data*, pages 461–472. ACM Press.

Getoor, L., Rhee, J., Koller, D., and Small, P. (2004). Understanding tuberculosis epidemiology using probabilistic relational models. *Journal of Artificial Intelligence in Medicine*, **30**(3), 233–256.

Ghanem, A. S., Venkatesh, S., and West, G. A. W. (2008). Learning in imbalanced relational data. In *Proc. International Conference on Pattern Recognition (ICPR)*. IEEE Computer Society.

Ghanem, A. S., Venkatesh, S., and West, G. (2009). Classifying multiple imbalanced attributes in relational data. In *Proc. 22nd Australasian Joint Conference on Advances in Artificial Intelligence*, pages 220–229, Berlin, Heidelberg. Springer-Verlag.

Google SketchUp ([Last Accessed May. 20, 2009]). http://sketchup.google.com/index.html.

Goold, A. and Rimmer, R. (2000). Factors affecting performance in first-year computing. *ACM SIGCSE Bulletin*, **32**(2), 39–43.

Grimes, D. B., Shon, A. P., and Rao, R. P. N. (2003). Probabilistic bilinear models for appearance-based vision. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, page 1478, Washington, DC, USA. IEEE Computer Society.

GroupLens Research ([Last Accessed June. 3, 2009]). MovieLens Data Sets. http://www.grouplens.org/node/73.

Guo, H. and Viktor, H. L. (2007). Mining imbalanced classes in multirelational classification. In *Proc. the 6th Multi-Relational Data Mining Workshop (PKDD/MRDM'07), in conjunction with 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD07)*, Warsaw, Poland.

Hastie, T. and Tibshirani, R. (1998). Classification by pairwise coupling. *The Annals of Statistics*, **26**(2), 451–471.

Heckerman, D. (1998). A tutorial on learning with bayesian networks. In *Proc. NATO Advanced Study Institute on Learning in graphical models*, pages 301–354, Norwell, MA, USA. Kluwer Academic Publishers.

Hiraoka, K. and Mishima, T. (2002). Classification of double attributes via mutual suggestion between a pair of classifiers. In *Proc. International Conference on Neural Information Processing (ICONIP)*, volume 4, pages 1852–1856.

Ho, T. K., Hull, J., and Srihari, S. (1994). Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16**(1), 66–75.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, **13**(2), 415–425.

Japkowicz, N. and Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis*, **6**(5), 429–449.

Jordan, M. I., editor (1999). *Learning in graphical models.* MIT Press, Cambridge, MA, USA.

Jordan, M. I. (2004). Graphical models. *Statistical Science*, **19**(1), 140–155.

Kamel, M. S. and Wanas, N. M. (2003). Data dependence in combining classifiers. In *Proc. 4th International Workshop on Multiple Classifier Systems*, pages 1–14.

Kedarisetti, K. D., Kurgan, L., and Dick, S. (2006). Classifier ensembles for protein structural class prediction with varying homology. *Biochemical and Biophysical Research Communications*, **348**(3), 981 – 988.

Keyhole Markup Language (KML) ([Last Accessed May. 20, 2009]). http://code.google.com/apis/kml/.

Kittler, J. (1998). Combining classifiers: A theoretical framework. *Pattern Analysis & Applications*, **1**(1), 18–27.

Kittler, J. and Hojjatoleslami, S. A. (1998). A weighted combination of classifiers employing shared and distinct representations. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, page 924, Washington, DC, USA. IEEE Computer Society.

Kittler, J., Hatef, M., Duin, R. P. W., and Matas, J. (1998). On combining classifiers. *IIEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(3), 226–239.

Kolen, J. F. and Pollack, J. B. (1990). Back propagation is sensitive to initial conditions. In *Proc. the 1990 conference on Advances in neural information processing systems 3(NIPS-3)*, pages 860–867, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Koller, D. and Pfeffer, A. (1998). Probabilistic frame-based systems. In *Proc. Artificial intelligence/Innovative applications of artificial intelligence (AAAI/IAAI)*, pages 580–587, Menlo Park, CA, USA. American Association for Artificial Intelligence.

Kubat, M., Holte, R., and Matwin, S. (1997). Learning when negative examples abound. In *Proc. European Conference on Machine Learning (ECML)*, pages 146–153, London, UK. Springer-Verlag.

Kubat, M., Holte, R. C., and Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, **30**(2-3), 195–215.

Kuncheva, L. I. (2004). *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience.

Lee, C.-I., Tsai, C.-J., Wu, T.-Q., and Yang, W.-P. (2008). An approach to mining the multi-relational imbalanced database. *Expert Systems with Applications*, **34**(4), 3021–3032.

Lézoray, O. and Cardot, H. (2008). Comparing Combination Rules of Pairwise Neural Networks Classifiers. *Neural Processing Letters*, **27**(1), 43–56.

Li, X.-L. and Zhou, Z.-H. (2007). Structure learning of probabilistic relational models from incomplete relational data. In *Proc. European conference on Machine Learning (ECML)*, pages 214–225, Berlin, Heidelberg. Springer-Verlag.

Ma, H., Zhou, W., Dong, X., and Xu, H. (2008). An empirical research of multi-classifier fusion methods and diversity measure in remote sensing classification. In *Proc. the 1st international conference on Forensic applications and techniques in telecommunications, information, and multimedia and workshop*, pages 1–4, ICST, Brussels, Belgium, Belgium. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Mashao, D. J. and Skosan, M. (2006). Combining classifier decisions for robust speaker identification. *Pattern Recognition*, **39**(1), 147–155.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, New York.

Murphey, Y., Wang, H., Ou, G., and Feldkamp, L. (2007). OAHO: an effective algorithm for multi-class learning from imbalanced data. In *International Joint Conference on Neural Networks (IJCNN)*, pages 406–411.

Noguez, J., Huesca, G., and Sucar, L. (2007). Shared learning experiences in a contest environment within a mobile robotics virtual laboratory. In *Proc. Frontiers In Education Conference (FIE)*, pages F3G–15–F3G–20.

Nugroho, A. S. (2000). Fog Forecasting Using Self Growing Neural Network 'CombNET-II:' A Solution for Imbalanced Training Sets Problem. In *Proc. of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)*, page 4429, Washington, DC, USA. IEEE Computer Society.

Opitz, D. W. and Maclin, R. (1999). Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research (JAIR)*, **11**, 169–198.

Ou, G. and Murphey, Y. L. (2007). Multi-class pattern classification using neural networks. *Pattern Recognition*, **40**(1), 4–18.

Pazzani, M. J., Merz, C. J., Murphy, P. M., Ali, K., Hume, T., and Brunk, C. (1994). Reducing misclassification costs. In *Proc. 11th International Conference on Machine Learning (ICML)*, pages 217–225.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA.

PKDD'99 Discovery Challenge ([Last Accessed June. 3, 2009]). Guide to the Financial Data Set. http://lisp.vse.cz/pkdd99/Challenge/berka.htm.

Provost, F. and Domingos, P. (2000). Well-trained PETs: Improving probability estimation trees. CDER Working Paper 2000-04-1S, Stern School of Business, New York University, NYU, NY 10012.

Quinlan, J. R. (1993). *C4.5: programs for machine*. Morgan Kaufmann, San Francisco.

Quinlan, J. R. and Cameron-Jones, R. M. (1993). FOIL: A Midterm Report. In *Proc. of the European Conference on Machine Learning*, pages 3–20, London, UK. Springer-Verlag.

Rifkin, R. and Klautau, A. (2004). In Defense of One-Vs-All Classification. *Machine Learning Research*, **5**, 101–141.

Savický, P. and Fürnkranz, J. (2003). Combining Pairwise Classifiers with Stacking. In *Intelligent Data Analysis (IDA).*, pages 219–229.

Sen, P. and Getoor, L. (2008). Cost-sensitive learning with conditional markov networks. *Data Mining and Knowledge Discovery, Special Issue on Utility Based Data Mining*, **17**(2), 136–163.

Stamatatos, E. (2008). Author identification: Using text sampling to handle the class imbalance problem. *Information Processing and Management*, **44**(2), 790–799.

Sun, Y., Kamel, M. S., Wong, A. K. C., and Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, **40**(12), 3358–3378.

Taskar, B., Abbeel, P., and Koller, D. (2002). Discriminative probabilistic models for relational data. In *Proc. Uncertainty in Artificial Intelligence (UAI)*, pages 485–492, Edmonton, Canada. Morgan Kaufmann.

Tax, D. and Duin, R. (2002). Using two-class classifiers for multiclass classification. In *Proc. 16th International Conference on Pattern Recognition (ICPR)*, volume 2, pages 124–127.

Tax, D. M. J., van Breukelen, M., Duin, R. P. W., and Kittler, J. (2000). Combining multiple classifiers by averaging or by multiplying? *Pattern Recognition*, **33**(9), 1475–1485.

Tenenbaum, J. B. and Freeman, W. T. (2000). Separating style and content with bilinear models. *Neural Comput.*, **12**(6), 1247–1283.

Vasilescu, M. and Terzopoulos, D. (2002). Multilinear image analysis for facial recognition. In *Proc. International Conference on Pattern Recognition*, volume 2, pages 511–514.

Wang, Y., Tan, T., and Jain, A. K. (2003). Combining face and Iris biometrics for identity verification. In *Proc. Audio- and Video-Based Biometric Person Authentication (AVBPA)*, pages 805–813.

West, D., Dellana, S., and Qian, J. (2005). Neural network ensemble strategies for financial decision applications. *Computers and Operations Research*, **32**(10), 2543–2559.

Weston, J. and Watkins, C. (1999). Support vector machines for multiclass pattern recognition. In *Proc. European Symposium On Artificial Neural Networks*.

Wetzel, J. N., OToole, D., and Peterson, S. (1999). Factors affecting student retention probabilities: A case study. *Journal of Economics and Finance*, **23**(1), 45–55.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2nd edition.

Xu, G., Bao, H., and Meng, X. (2008). Multi-relational classification in imbalanced domains. In *Proc. International Symposium on Advances in Computation and Intelligence (ISICA)*, pages 562–570, Berlin, Heidelberg. Springer-Verlag.

Yin, X., Han, J., Yang, J., and Yu, P. S. (2004). Crossmine: Efficient classification across multiple database relations. In *Proc. of the 20th International Conference on Data Engineering*, page 399, Washington, DC, USA. IEEE Computer Society.

Zhang, G., Thorndyke, B., Ohland, M. W., and Anderson, T. J. (2004). How science course performance influences student retention - a statistical investigation. In *Proc. American Society for Engineering Education (ASEE) Annual Conference & Exposition.*