A comprehensive survey of methods for overcoming the class imbalance problem in fraud detection

by

Dr Peter Brennan

Supervisor: Dr Markus Hofmann

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF M.SC IN COMPUTING AT INSTITUTE OF TECHNOLOGY BLANCHARDSTOWN DUBLIN, IRELAND

June 2012

Declaration Of Authorship

I,Peter Brennan, declare that the thesis entitled A comprehensive survey of methods for overcoming the class imbalance problem in fraud detection and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this institute;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this institute or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;

Signed: .....

Date: .....

Table of Contents

List of Figures

List of Tables

Abstract

Fraud is a hugely costly criminal activity which occurs in a number of domains. Data mining has been applied to fraud detection in both a supervised and non-supervised manner. When a supervised data mining approach is used, one of the biggest problems that is encountered, is the problem of class imbalance. The class imbalance problem is not unique to the domain of fraud, but also occurs in fields as diverse as medical diagnosis and quality control. There are two basic means of overcoming the class imbalance problem, these are data methods and algorithmic methods. Data methods generally involve an under sampling, over sampling or hybrid over/under sampling approach. Other data method investigated include SMOTE, which uses a K-NN learner to artificially synthesize minority class samples. Algorithmic methods investigated include using either a mis-classification cost in the case of the Metacost procedure or metacost thresholds. Other algorithmic methods include the use of learners which are not sensitive to the class imbalance problem. The different methods for overcoming the class imbalance problem are implemented using open-source software. 3 datasets are used to investigate the usefulness of the different methods. 2 of the datasets are from the domain of fraud, while the third is from the domain of medical diagnosis.

Acknowledgements

This thesis would not have been possible without the support of many people. I would like to express my sincere gratitude to:

- My supervisor, Dr. Markus Hofmann who was abundantly helpful and offered invaluable advice, assistance and support throughout my studies in ITB.
- My parents.
- My country.

Chapter 1

Introduction to fraud and data mining approaches to fraud detection

A number of current studies have shown white collar crime to result in year on year losses of 250 billion dollars. As opposed to approximately 17.6 billion dollars resulting from losses which occurred pertaining to crimes against people or property, (Holtfreter et al. 2008). These estimates are considered to be extremely sensitive, and one well respected writer in the field (Sparrow 2000) estimates that healthcare fraud alone to be worth anywhere between 100 to 400 billion US dollars. However white collar crime does not take precedence compared to violent criminal activity or terrorism (United States Department of Justice 2005). While white collar crime, particularly fraud may be of low importance to governments, its societal impact in terms of economic scale is huge.

Concepts or models concerning crime would assume that the deciding factor in an individual deciding to commit a crime (for example in the case of insurance fraud a person lodging a false claim) will be size of the prospective monetary benefit versus the consequences of being caught in terms of possible fines, custodial sentences and other punitive measures. This can be very simply summed up by the adage, 'Risk versus Reward'. So taking insurance fraud as our example, if the likelihood of success is sufficiently high to deliver a considerable return, the bogus claim will be filed (Becker and Gray 1969).

While consumers tend to have a negative opinion of the insurance business as a whole, the most overwhelming issue in consumers acceptance of fraud are the views of society towards fraud as well as the individual's own views on the practice of fraud (Tennyson 1997). A consumer mindset of approval (even if it's only tacit support) will cause the velocity at which fraud occurs to increase. The Insurance Research Councils (IRC) study of insurance fraud revealed a relatively strong agreement or acknowledgement of fraudulent activities amongst consumers, with eleven per cent of people interviewed finding it 'almost always' or 'usually' acceptable for a health care practitioner or a solicitor to lodge fees or 'demand of payments' for care which was not provided and five per cent thought it was perfectly agreeable for people to claim for injuries if not present in the car at the time of the accident (jump ins) (Insurance Research Council 1991).

1.1 What is Fraud?

When considering the nature of fraud, there are three major points which must be addressed, these are:

1. What do we mean when we say fraud?
2. How prevalent is fraud and what can be done to combat it?

3. And finally, how do you best disincentivise fraud through legal action?

The most useful definition of criminal fraud is that by Derrig et al. (2006) and covers four different areas of the law, these are:

1. It must be carried out with the intentional purpose of defrauding.
2. The fraudulent act undertaken must be illegal.
3. There must be a monetary gain.
4. The intentional inflation of cost damage to person or property must occur.

Wang et al. (2006), has also offered an extremely succinct definition as fraud being:

> "a deliberate act that is contrary to law, rule, or policy with intent to obtain unauthorized financial benefit"

.

1.2 What are the different types of fraud that occur in these domains?

When the four governing rules stated above are met it is possible to bring legal action against a person. If one or more are not present the case against someone would be in some difficulty, however it does not mean that fraud is not taking place. Therefore it should be remembered that fraud in its strictest sense is the improper use of a companies income which may or may not lead to a criminal prosecution. Fraud can be broken down into two distinctive domains, that is criminal fraud where a person or group of people may commit fraud and non criminal fraud. The two types of fraud are defined below:

1. Criminal Fraud, where the aim is to enter a criminal enterprise. This type of fraud is a concern for both law enforcement and insurance companies.
2. Non criminal fraud. This is also known as build up and involves exaggerating a claim. This type of fraud is harder to prove but is economically very damaging.

Criminal fraud, is of particular interest to law enforcement as not only does it cost companies and ultimately customers money, it has also been used as potential source of income for terrorists and organised crime. For example, the terrorist group Jemaah Islamiyah have used operatives to hack into business accounts of AT & T and redirect money to terrorist groups (Senigupta 2011). Online fraud is now recognized as mainly being perpetrated by organized crime. Worral (2012) estimates that about 80 per cent of online fraud being carried out by organized crime. Insurance fraud by organized crime has also been noted in both the US and UK. In 2004 a number of individuals in New York, believed to be operating as members of an organised crime ring, were prosecuted for making fraudulent claims totalling more then one million dollars (Johnson 2006).

In the UK, the British government indited a number of individuals, believed to be members of an organised crime ring, in 2010 for insurance fraud involving a substantial six figure sum (insurance-weekly.co.uk 2010).

1.3 What are the different domains where fraud occurs?

High profile financial fraud cases, including the Ponzi scheme instigated by the Madoff's which resulted in an estimated cost of US $ 50 billion worldwide or the ENRON broadband services fraud cases, receive substantial news coverage they are not atypical of the most common types of fraud that occur. The most common areas where fraud occur are in tax evasion, consumer fraud, insurance fraud and credit card fraud. According to the BDO (Binder, Dijker, Otte and Company), reported fraud was above the two billion sterling mark in 2011, rising by 50 percent over the previous year with more than a third of all fraud being tax evasion (BBC 2012). However this figure is dwarfed by the much larger estimated value of fraud, combining reported and non-reported fraud. The total cost of fraud in the UK is reported to be 30 billion sterling a year with two billion sterling a year being lost in the general insurance business alone (insurance daily 2011). This costs the average policy holder forty four pounds as additions to their annual premium (insurance fraud bureau 2012). One reason offered for the rise in fraud is the economic downturn. The (Conning and Co 1996) study estimated insurance fraud to top US $ 120 billion annually in the US. The same study found that systems designed to fight fraud show a very large return on investment, which in 1995 was estimated to be 6.88:1.

In automotive insurance, typical fraudulent behaviour would be:

1. Staged motor accidents with faked reports of injuries.
2. 'Jump in' passengers who are added to a claim after it occurs to increase claim amounts.
3. Overstated injury claims with treatment that goes on for longer than usual for that type of injury.
4. Deliberately untrue statements of cars being stolen.
5. Over-exaggerating vehicle damage after an accident has occurred.
6. Claiming for damage to a car involved in an accident, when the damage to the car already existed.
7. 'Garaging' of a car in another location to which the car is insured in.
8. The making of untrue claims before the policy period began.
9. The untrue reporting of an accident that occurred while a car was reported parked.

While in health insurance, (nhcaa.org 2012) the type of fraudulent activity that can occur would include:

1. Charging for treatments that were not provided or the stealing of patient information and using this information to fabricate claims.
2. Upcoding; this is the claiming for medical treatments that were not carried out or claiming for more expensive treatments than were actually needed.
3. Carrying out medical procedures which are not needed simply for the purpose of generating an insurance claim.
4. Carrying out false diagnosis so as to carry out erroneous treatments which are not needed.
5. Charging a patient for services that are covered in full by their health insurance plan.
6. Taking bribes for patient referrals.

In bank fraud, the most common types of fraud are:

1. Money laundering. Money laundering is probably the most high profile types of banking fraud. Money laundering is the act of obfuscating the true origin of funds raised through illicit means. In doing this, illegally gained monies are then transferred into the domain of legitimate commerce. These activities can result in criminals being harder to prosecute as they are no longer involved in day to day criminal activities, but are instead involved in the financing of criminal activity. This can lead criminals to becoming extremely powerful and difficult to prosecute. Introducing monies from illegitimate means into legitimate commerce, can lead to the tainting of financial institutions and result in criminals having unwarranted economic and political power (FBI 2007). Examples of this would be the rise in political and economic power of narco-traffickers in South America in the 1980's and in Mexico currently.

2. Mortgage fraud. Mortgage fraud can be defined as:

   > " Material misstatement, misrepresentation, or omission relating to the property or potential mortgage relied on by an underwriter or lender to fund, purchase or insure a loan"

   (FBI 2007).

3. Credit card fraud. With the rise of e-commerce over the last 12 years, the use of credit cards for purchasing on-line has risen dramatically. This has given criminals more potential opportunities for carrying out credit card fraud. By 2007 credit card fraud in the US alone had reached epidemic levels, worth an estimated value of 3.2 billion dollars in 2007 (Panigrahi 2009). In 2008 the Guardian reported credit card fraud in the UK to be worth 500 million sterling (Rogers 2009). To combat fraud, credit card companies use both prevention and detection methods. Prevention methods include; credit card authorization codes, real time authorization systems and physical address authorization systems. Detection methods include the use of computer systems and sophisticated analytics to catch credit card fraud.

However like in fraud detection in car insurance, when fraud detection measures fail, fraud prevention methods come into play. Data mining technologies have been long used for this type of work particularly Neural Networks (Ghosh and Reilly 2004). These systems have proved to be very effective for credit card fraud detection, with a 20 to 40 percent reduction in total fraud losses.

1.4 The different ways of combating fraud; prevention and detection.

There are two ways in combating fraud. These are through fraud detection and fraud prevention methods. Fraud prevention measures are a set of procedures, rules and guidelines put in place to stop fraud occurring.

Depending on the domain there are number of ways to prevent fraud, examples of this are the use of SIM cards on mobile phones and passwords on computer systems. While in e-commerce, the use of HTTPS (HyperText Transfer Protocol Secure) which is a technology based on both HTTP and SSL/TLS protocol for logging into secure payment sites. Other measures include the use of security questions for on-line banking. However these methods provide inconvenience to the customer and often a balance must be struck between security and convenience. Fraud detection is the identification of fraudulent behaviour once it has occurred. Once detection has occurred action can be taken to limit the fraudulent activity. To accomplish this, fraud detection requires constant monitoring as well as constant evolution.

Fraud detection methods are in a constant state of change or flux due to the nature of fraud. This is because as soon as detection of a type of fraudulent behaviour takes place, criminals create new plans

and schemes for fraud. Furthermore new criminals entering the fraud business may carry out fraud using existing methods causing some fraudulent activity to be cyclical. Fraud detection therefore can be seen as being part of an overall strategy which in many areas has become a business critical issue. This is because it being both common place and difficult to combat. Fraud has been found difficult to combat due to troublesome and complex nature of designing measures to prevent it.

Finally when considering fraud prevention; one must consider where the fraud occurs is it internal or external to a company. Internal fraud occurs within the normal operation of a company, taking insurance as an example this can be further broken down into underwriting fraud and procedural fraud. By enforcing operational checks and procedures it should be possible to minimize these types of internal fraud.

The second type of fraudulent behaviour is external fraud or when discussing insurance fraud, policy-holder related fraud. One must consider very carefully this difference when designing a fraud detection tool. Perpetrators of internal fraud will be knowledgeable not only about procedures and checks, but also what types of fraud are being sought after. This is opposed to external fraud ,as potential fraudsters will not have detailed knowledge of anti-fraud controls or any detection systems in place.

1.5 The different approaches to combating fraud; expert human systems versus machine based systems and hybrid machine expert based systems?

In the 1980's, Insurance companies began to realize the seriousness of fraud. They began to combat insurance fraud through the establishment of specialist teams of experienced claims adjusters with expert skills and training in claims investigation (Ghezzi 1983). While this originally happened in the 1990's in the United States, the format quickly spread though out the world (Dionne and Belhadji 1996). By the end of 1990's, insurance companies had created very comprehensive procedures to try to prevent fraud. In the late 1980's early 1990's studies were undertaken to carry out a comprehensive review of fraud and the development of metrics for categorizing fraud. These studies led to the development of a number of metrics which were indicators of fraud, these became known ubiquitously as "red flags". These red flags could be used by claims adjusters to help recognize or single out potential fraudulent claims. Due to the computerization of claim systems it became possible to capture more and more data regarding claims. Business intelligence systems and reporting techniques could be used to generate hot lists of potential fraudulent claims. This led to the development of a two stage process for dealing with fraudulent claims. During the first phase the claim is examined by a claims adjuster to analyse the amount of liability the insurance company may have for the payment of the claim. Claims which are for small amounts or appear to be non fraudulent are settled in a normal manner. Those that are irregular or are for substantial amounts of money, are referred onto a SIU (Specialist Investigation Unit, a team of vastly experienced claim handlers adept at identification fraudulent claims) hot-list for further review by the investigations team. However as always with such manual systems the bottleneck is the examination and analysis of these hot lists for fraud and also the time taken to create them by front line adjusters. Early detection of these potentially fraudulent claims and non fraudulent claims by pattern recognition software would allow larger amounts of claims to be both processed and investigated. This could save an insurance company time by having to use less claims adjusters to investigate claims. Also by looking at a larger numbers of claims, the insurance company would be able to catch more fraudulent claims. The use of a machine learning element for classification of possible fraudulent and non fraudulent claims by a binary classification system would allow the reallocation of front line claim adjusters to scanning for possible fraud, to processing the claim (Viaene et al. 2002). Insurance Service Office (ISO), a US based insurance research consultancy (ISO 2012), have found that the most common approaches to fraud detection are SIU personnel. While 80 percent of insurance companies used red flags or indicator cards to carry out fraud

checks, less than 25 percent use data mining approaches to detect fraud. This would suggest that data mining approaches to fraud detection are currently vastly under used. Also, a data mining approach to fraud detection would allow the use of less resources.Large volumes of claims could be automatically screened by classifiers to label them as potentially fraudulent or non-fraudulent.

1.6 An introduction to data mining for fraud detection, its emergence and development

Data mining or KDD (Knowledge Discovery in Databases) is the act of gaining an insight into or discovering patterns from large datasets. It is an area of growing significance due to companies being able to gain significant competitive advantage over their rivals through its use. With the emergence of data warehouses being able to provide business data in a form that can be readily accessed and analysed, data repositories from which to source datasets for data mining are available. At the same time the field of machine learning and artificial intelligence has blossomed, producing a number of techniques that can be used for data mining. This emergence of data warehouses with readily available closed source and open source data mining tools has made data mining a popular technique for discovering patterns or relations in large corporate databases by data analysts. As part of any data warehousing initiative a master data management (MDM) strategy would have to be implemented. MDM consists of a number of procedures, methodologies and instruments which aim to offer a constant, uniform and dependable definitions, organization, administration and management of Master Data. MDM at its core offers a suite of applications, operations and procedures for the collation and combining, and the guaranteeing of the quality of data and its consistency. MDM ensures that there is one single version of truth concerning business entities within an organization. Data-mapping can be considered a complimentary technology to data warehousing, but for transactional data and can be considered the practice of documenting the data element relationships connecting two disparate data models. It is generally considered the first step in data integration task. Not only does it highlight what data conversions are necessary, it will also show up the relationships between the different data elements as they pass from system to system. If any concealed relationships between the data occur they will be revealed. This information can be useful in uncovering fraudulent behaviour. As concealed relationships are sometimes evidence of fraudulent activity or people trying to hide fraudulent or criminal activity. Data-mapping can also be used in fraud detection as the initial stage that will create the framework for gathering information from data regarding fraud. Data-mapping is necessary in a fraud detection to highlight what systems, individuals and organisations that data might pass through during the processing of a claim. The mapping of data flows in a data-mapping exercise can help in the identification of possible opportunities for fraudulent behaviour (Busch 2008). As the growing collective cognizance of fraud grew, more exact and accurate measures for combating it emerged, along with the creation of databases holding information on fraudulent activities. An example of this type of database is the National Insurance Crime Bureau (NICB) database. A similar system is available in Ireland and the UK as the Insurance link database (InsuranceLink 2012).This database holds records of claims and stolen vehicles and is available to members and to law enforcement (Dauer 1993). Besides using standard reporting in the investigation of fraud, a number of statistical and machine learning methods, both supervised and unsupervised methods have been used (Brockett et al. 1998). While data mining has become popularized as a technology, it is still a complex multi-step process. The process can be described as composing of the following steps:

1. Dataset selection. As was mentioned in the previous paragraph, datasets for mining are usually selected from corporate warehouses. When the corporate data warehouse does not hold the required information, the datasets may have to be sourced from operational systems. This can sometimes cause difficulty especially if the data has to be sourced from multiple platforms and formats of data. To ease this problem, ETL (Extract transform Load) tools can be utilised.

2. Cleansing and preprocessing. During this stage a number of procedures are applied to the data to overcome data quality issues. These can range from the mundane tasks of validation of dates to more exotic techniques, such as geo-coding of address fields to geographic locations and imputation of missing data. Two of the most common problems to do with cleansing and pre-processing are overcoming the problems of high dimensionality in some datasets, often referred to as the 'curse of dimensionality' and skewed datasets.

3. Data Analysis. In this stage the dataset is analysed. A data-mining approach must be decided upon. A supervised data mining approach or unsupervised data mining approach is chosen. In supervised data mining, a dataset with a label or a known value is used to train a model, typically this is a classification or prediction task. In unsupervised data mining such techniques as clustering or association rule mining are used to learn more about a dataset and with the aid of domain experts are used to gain an insight into the problem area.

4. Interpretation and Evaluation. Here the models performance is evaluated against an independent dataset to the one that was used in training the model. Where possible, the insights learned are formulated into an actionable business plan with the aim of aiding the organization who commissioned the exercise. For example a decision tree, rule set or association rules can be coded as SQL (Structured Query Language) statements and deployed to a database to make classifications or predictions.

5. Deployment and model lifecycle management. During this stage the models developed and evaluated in the previous stage are deployed. At this stage it is important to create a life cycle plan and time-lines to re-evaluate the models developed previously. This is necessary so as to avoid concept drift. Concept drift is the process or re-evaluating a model at regular intervals with amended datasets with the purpose of (a) evaluating the model to see if it still accurately predicts behaviour or the model it developed is still valid and (b) to retrain the model were necessary so as to maintain performance of the model or to develop new insights that evolve over time.

There a number of different data mining application areas supported by algorithmic methods to distil the pertinent connections or associations held within the data. The data mining application areas are:

1. Classification. Classification involves developing a model to predict qualified labels for unclassified items so as to mark them as different from other items belonging to a different class. Common classification techniques are SVM's (Support Vector Machines), neural networks, decision trees and (K-NN) K-Nearest Neighbour.

2. Prediction. Prediction extrapolates numeric values based on patterns held within the data. Typical techniques used to deliver this functionality include logistic regression.

3. Clustering. Clustering is the separation of objects into groups, with the objects belonging to each group being alike but not like objects belonging to other groups. Common clustering techniques include hierarchical clustering and K-means clustering.

4. Outlier detection. Outlier detection is used to detect objects that show extreme differences between objects in a dataset with the overall population.

5. Visualization. This is the act of making data understandable through the use of diagrams or pictorial methods to display complicated information or relationships held in data. One of the best examples of this is Charles Minards (Jacobs 2007) flow map of Napoleon's March on Moscow and subsequent retreat. Minard's map is famous as it displays a number of factors on a 2-D representation. Edward Tufte (Tufte 2001), the father of data visualization has called it "the best statistical graphic ever drawn".

6. Social Network Analysis. Social network analysis considers social connections or associations through the utilization of network theory which uses nodes and ties. Nodes being the actual people in the network with the ties being the association or links between these people.

1.7 A review of current software available for fraud data mining

A number of closed and open source tools available for data mining, allow a analyst to build data mining applications. These include closed source tools like SAS Enterprise Miner and SPSS Modeler. Open source tools include; R, RapidMiner, Weka (Drazin and Montag 2012) and KNIME. Some of the closed source tools also offer pre-built frameworks for detecting fraud. Besides these tools a number of pre-built applications exist, these are reviewed in the following section.

1.8 Financial fraud data mining applications

One of the earliest tools used in financial fraud detection is Mineset (Becker 1997). Mineset utilizes various data-visualization methods with supervised and unsupervised learning data mining methods which can be used, for example, to detect credit card fraud. Another well known data mining credit card fraud detection tool is FALCON (Brachman et al. 1996), which utilizes a neural network to establish credit card fraud. Kokkinaki (1997) has also shown how similarity trees can be used in a similar vein to FALCON. FAIS (Senator et al. 1995) was developed to discover greater than normal cash exchanges. Other tools include Wirevis, a tool developed in collaboration with the Bank of America (Chang et al. 2007), who used a series of coordinated chain of techniques for visualizing wire transactions. The visualization technique combined the use of network views and heatmaps to deliver a clear view of the highly complex temporal, monetary and usage statistics of the transfer relationships amongst different account holders. Other tools used to detect financial fraud have been developed based on Benford's law, one such tool is DATAS (Lanza 2000). This tool is based on the fact that there are expected rate of occurrences of specific digits in numbers, the tool delivers a visual representation of a grouping of numbers to be audited in easy to understand graphs. Fraud detection utilizing data mining has also been used in the arena of health insurance. An electronic fraud detection system was developed by Major and Reidinger (1992) who trained a model based on a number of historical claims to help in the identification of healthcare practitioners which may have presented possible fraudulent claims. Another system developed for the healthcare industry is KEFIR. Which allows for critical examination based on a number of critical factors including cost, quality and particular common practices for treatment so as to discover anomalies. These anomalies are often signs of fraud (Matheus and Piatetsky-Shapiro 1996). Another tool used both in insurance and financial fraud investigations is analysts notebook (I2 2012), which is used to carry out investigation of criminal networks through both reporting and visualization techniques. The software uses a mix of both social network analysis and a series of pre-created reports to allow the reduction of very sizeable datasets to intelligence which may lead to evidence which can be used in a criminal prosecution (I2 2012).

1.9 Problems with taking a data mining approach to fraud detection.

There are a number of technical problems when trying to discover fraud (Jensen 1997). As was discussed previously, a relatively minor proportion of claims made are fraudulent. This results in an imbalanced dataset and measures must be taken to overcome this. Phua et al. (2005) definitive and all inclusive study of data mining fraud detection systems has shown a large number of studies using credit transactional and insurance data showing a large class imbalance. The second major problem when considering fraud

detection based data mining solutions is that there are very few open datasets available for data mining. They found that 80 per cent of surveyed papers utilized class imbalanced datasets. Phua et al. (2004) have also claimed that datasets comprising data taken from real world commercial entities are difficult to get as they may offer their rivals a competitive advantage by informing them of what variable / attributes to construct a dataset of for mining. Also due to the political and economic sensitivity of the topic there is very few openly available datasets. Finally as fraudsters are adept at developing new methods for perpetrating fraud, any model used to counter fraud must be highly adaptable. To overcome this problem, (Barse et al. 2003) the creation of artificial data which approximates very closely real data has been proposed. However authentic fraudulent and non-fraudulent data is needed, this acts as seed data to create the artificial data. Fawcett et al. (2003) has suggested the use of a email dataset as a substitute dataset for data mining fraud detection studies, as a number of issues encountered when mining email data are also present when mining fraud detection data. When creating a model to detect fraud, it must ideally be easy and quick to retrain, not sensitive to outliers, noise in the data or missing data.

1.10 The aims and objectives of the study

The aims of the study are three fold:

1. To investigate the use of open source data mining software to combat fraud, with particular attention to fraud/non-fraudulent cases classification system.

2. To identify the type of problems that arise when taking a data mining approach to fraud detection, particularly the class imbalance problem.

3. To identify solutions to the class imbalance problem and show how they can be implemented through the use of open source software.

1.11 The structure of the thesis

In chapter 2, the class imbalance problem, one of the major problems in fraud detection, is discussed. The problem is clearly described and how it pertains to fraud detection and other domains. The different solutions from an algorithmic and data method approach to the problem are then discussed at length. In chapter 3 the different ways of overcoming the class imbalance problem through the implementation of the different algorithmic and data methods in RapidMiner and R are discussed. A review of performance metrics for assessing learning in a class skewed environment is also carried out. The experimental methodologies are either displayed in code or through a RapidMiner workflow. Chapter 4 describes the approaches used to overcome the class imbalance problem in the car insurance fraud dataset. Both data methods and algorithmic methods were utilized to overcome the class imbalance problem in the dataset. The data methods encompassed under sampling, oversampling and hybrid approaches of the two. The SMOTE algorithm was also used to synthesize artificial replicants of the minority classes in an attempt to rebalance the dataset. The algorithmic methods involved the use of the Metacost procedure to try and overcome the class imbalance in the dataset by introducing a cost of misclassification. The data methods proved to be vastly superior to the algorithmic methods surveyed. They proved to be easier to implement and delivered better performance. Chapter 5 describes the application of the data methods and algorithmic methods to overcome the class imbalance problem in the consumer fraud dataset. This chapter looks at the application of the different data methods and algorithmic methods to overcome the class imbalance problem. Besides the previous methods, new methodologies are introduced. Additional data methods introduced are a progressive under sampling approach. Additional data methods surveyed include the use of metacost thresholds and the use of learners which are less susceptible to the class imbalance problem.

Again data methods prove superior to algorithmic methods. The Metacost procedure and the use of metacost thresholds were not found to be as useful as the data methods. However the use of learners which are not susceptible to the class imbalance problem such as the RIPPER algorithm proved to deliver a level of performance approaching that of the data methods. Experiments to improve the performance of these algorithms by combining the classifiers through voting or stacking was also investigated. Chapter 5 is similar in structure to the previous two chapters. However there are significant differences. The thyroid dataset is not from the the domain of fraud, but instead medical diagnosis. The choice of this dataset was for two reasons. Firstly, there were no other readily and freely available datasets from the fraud domain. Secondly the thyroid dataset is not only class imbalanced it is also a multi class dataset. This caused further complications to the data mining approach to overcoming the class imbalance problem. Both data methods and algorithmic methods were investigated. Again the data methods proved superior to the algorithmic methods. However the use of algorithmic methods such as RIPPER when combined with other classifiers were found to deliver performance approaching that of the best of the data methods. Chapter 7 reviews all methods for overcoming the class imbalance problem across all the datasets. The different approaches to overcoming the class imbalance problem using both data methods and algorithmic methods are reviewed and discussed. The methods that delivered the highest level of performance gain are highlighted. The problem of implementing the different methods are also discussed. Finally a recommended implementation strategy of the optimal methods are given through open software.

Chapter 2

The class imbalance problem

2.1 An introduction to the class imbalance problem

When utilising orthodox machine learning algorithms the mining of skewed datasets can result in models that are strongly predictive for the larger class, while delivering performance which is poorly predictive for the minority class (Xu and Chow 2006, Zhou and Liu 2006). This is due to the fact that orthodox classifiers will attempt to return the most correct predictions based upon the entire dataset, this results in them categorizing all data as belonging to the larger class. This class is usually the class which is of least interest to the data-mining problem. In the case of fraud data-mining, the majority class is the class where no fraud has occurred and the minority class being fraud. The problem of skewed datasets in data-mining are illustrated in figure 2.1, if the minority class is very small a learner can deliver very high predictive accuracy even though it has classified none of the minority class correctly. Taking the area of interest of this thesis, fraud as an example, the minority class would be 'possibly fraudulent claims' while the majority class would be 'non fraudulent'. Other real world problem areas where this type of problem occurs are:

1. Medical Diagnosis. The use of a homogeneity index to judge the appropriate level of information granularity to overcome the problem of skewed datasets in medical diagnosis has also been demonstrated (Su et al. 2006).

2. Quality assurance of goods. In the quality assurance of cellular phones (Su et al. 2006 b), specifically the radio frequency test. Developing a model that can accurately predict the result of the test can lower test time and the accompanying cost associated with the test. By coupling information granulation with a neural network to overcome the class imbalance in the dataset (the class imbalance is due to the number of defective models being far less than the number of correctly functioning models) a highly efficient model can be created.

3. Credit scoring. The class imbalance problem to be major problem in credit scoring (Yen and Lee 2009).

4. Electricity Distribution. Xu et al. (2007) who has developed a AIRS (Artificial Immune Recognition System) for identifying power outages due to the three major incidents resulting in outage.

2.2 Methods of overcoming the class imbalance problem

Weiss (2004) has created a list of 5 types of problem that occur when learning from skewed datasets.

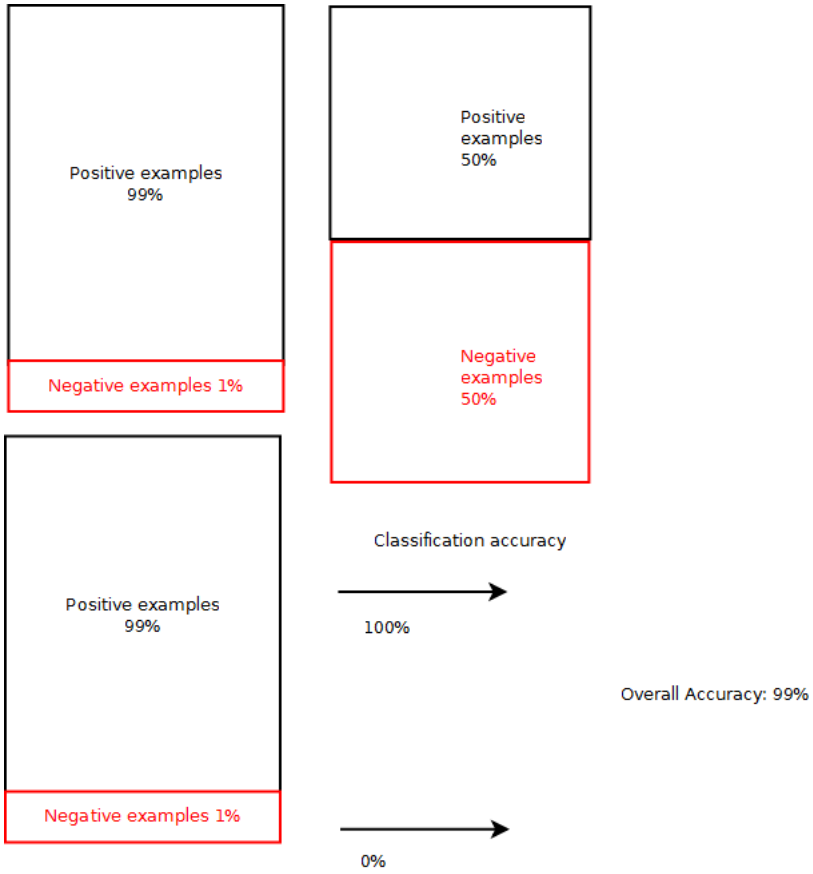

Figure 2.1: The class imbalance problem.

1. Incorrect evaluation measures. One must be very careful when selecting the correct performance measure to evaluate a learner.

2. Absolute Scarcity of data. The number of items that belong to the minority class are small when compared to the total number of samples, this can make it difficult to observe patterns within the minority class.

3. Data fragmentation. This can be a problem as most classifiers utilize a divide and conquer approach. This causes a continual division of the learning space. This results in patterns that are found in the data as a whole cannot be found in the resulting partitions formed by this divide and conquer strategy.

4. Inappropriate inductive bias.

5. Noise. While effecting the performance of a data mining system as a whole, noise has been found to have a greater effect on the minority classes.

To overcome these problems Weiss et al. (2007) also collated a number of approaches to overcome these problems. These are listed below, showing the data mining problem and the method to address the problem:

1. Data mining problem - A particular data mining method to address the problem (methods that are resistant to the class imbalance problem), data mining algorithms that can cope with the class imbalance. Learners of this type would include RIPPER (Gaines and Compton 1995) or Ridor (Cohen 1995).

2. Improper evaluation metrics - The use of more appropriate evaluation metrics (F-measure, G-mean or AUC values), cost sensitive learning (Metacost procedure).

3. Total and absolute rarity of the minority class - Over sampling, SMOTE.

4. Relative rarity or sparsity of the minority class - needles in haystack. Methods to do this include, learning only the rare class, segmenting the data, sampling (over and under or hybrid approach), two phase rule induction, accounting for rare items, cost sensitive learning, boosting.

5. Data fragmentation rare classes/cases split apart - Partition or divide the rare cases into separate classes, Sampling (over and under or hybrid sampling approach), Learn only the rare class.

6. Inappropriate bias - Cost sensitive learning, Appropriate evaluation metrics (such as F-measure, G-mean or AUC value).

7. Noise - Advanced sampling procedures and methods, removal or treatment of outliers.

These methods can be divided into two general approaches. These are data methods and algorithmic methods. At the algorithmic level, cost sensitive learning is probably the most well known method of dealing with the class imbalance. The data centric solutions generally involve re-sampling. The use of over/under sampling methods along with methods for creating artificial (SMOTE) samples are discussed in the following sections. The methods based on some form of re-sampling methods are listed below:

1. Oversampling the minority class, most commonly implemented as random oversampling, (ROS).

2. Under sampling the majority class, most commonly implemented as random under sampling, (RUS).

3. Combined over and under sampling, most commonly implemented using a combination of random under sampling and over sampling, (ROS/RUS).

4. Use of SMOTE algorithm to artificially synthesize items belonging to the minority class.

5. The implementation of cost sensitive learning.

6. Cluster based over-sampling.

7. Boosting.

8. Use of Learners which can cope with class imbalance in the data, i.e weighted random forests.

Of the above methods over and under sampling, or methods based on these are the most common data methods for overcoming the class imbalance problem.

2.3 Oversampling, under sampling and hybrid approaches to overcoming the class imbalance problem

The class imbalance problem can also lead to other problems due to the small size of the minority class. These are often caused by within class differences. When considering the class imbalance problem were the minority class is very small, the ability of a learner to discover patterns in the minority class data can not be relied upon. However as the number of classes increases the learner becomes better at being able to recognize minority class samples from the majority (Japkowicz and Stephen 2002). Sampling is the most popular means for overcoming this problem. Sampling is used as a means of altering the distribution of the minority class so that it is not under represented when training the learner. There are three basic approaches to overcome the class imbalance problem. These are oversampling of the minority class, under sampling of the majority class or the use of a hybrid approach based on both. Ling and Li (1998) have suggested the utilization of a combined approach which utilizes both the oversampling of the class with a smaller number of instances and under sampling of the class with the larger number of instances. These different sampling methods can be explained using the following example. In the case of under sampling, take for example a case were there are 1100 samples in the training set, distributed as follows; the minority class consists of 100 samples and the majority class consists of a 1000 samples. When carrying out under sampling the majority class would be reduced in number through the use of a sampling procedure to 100. In the case of minority oversampling a sampling procedure would be used to adjust the sample size to a sample size of 1000. When a hybrid approach is used, the majority class would be under sampled to a sample size of 550 and the minority class would be oversampled to a sample size of 550. When under sampling is used to overcome the problem of class imbalance, the number of majority class examples is reduced until the number of majority samples equals the number of minority samples. When using this solution to the class imbalance problem certain problems may arise from removing such a large number of the majority class. Due to a number of potentially useful samples from the majority class may be discarded. When this is the case, the majority class is made up of a number of smaller groups, which are often referred to as sub-concepts. These sub-concepts may themselves be skewed, this is termed 'within class imbalance'. This further complicates the data mining exercise (Zhou and Liu 2006) and a more complicated under sampling approach must be used which carries out under sampling a number of times so as to synthesize a number of classifiers. This approach is similar to ensemble methods such as boosting and is discussed further in the chapter. Alternatively a focussed under sampling method may be used which provides an educated selection of the majority class. Under sampling does offer a number of benefits to oversampling. The main one being that it results in a smaller training set as compared to oversampling, thus resulting in shorter training times. This smaller training set will also allow the use

of more complex learners, like neural networks (Zhiyong et al. 2009). Oversampling the minority class creates a number of replicants of the minority class until the number of the minority class examples equals the number of examples in the majority class. This redistribution functions similarly to under sampling as it rebalances the distribution of the minority class when compared to the majority class. Thus allowing the learner sufficient samples of the minority class to train the model to recognize the minority class. Oversampling can result in a number of problems, these include over-fitting of a model especially in the cases of noisy data. Also oversampling does not result in more information being included in the training set, which can cause the production of overly complex models. Take for example the case of decision trees which when developed with over sampling, can result in overly complex decision trees (Steinbech et al. 2006). This is due to the larger number of the minority examples causing the learner to not prune a number of sections of the model, which encompass a very small number of training examples.

2.3.1 Alterations and adaptations of the use of oversampling the minority class

Oversampling is amongst the most popular methods for overcoming the class imbalance problem. This is due its ease of implementation. For example in RapidMiner, it can be implemented using the bootstrapping operator. In Weka using the re-sample operator, in R using the sample function with replacement. Oversampling is most commonly implemented using random sampling of the minority class with replacement of the minority class. This is known as random over sampling of the minority class with replacement, or more simply it is known by the acronym ROS. In ROS, random oversampling with replacement is carried out. The samples must be replaced otherwise the pot of minority class samples would be exhausted before rebalancing of the minority class with respect to the majority class had taken place. Japkowicz (2000) has carried out studies on methods suitable for over sampling the minority class. Two methods of sampling were considered these were:

1. Random sampling were the minority class was randomly re-sampled with replacement until the number of objects in the minority class equalled the number in the majority class.

2. Focussed re-sampling was also considered, this involved selective re-sampling of minority class objects which exist close to a border between the minority and majority class.

Japkowicz (2000) found that either methods were useful in overcoming the class imbalance problem however the use of the more advanced focussed sampling method resulted in any significant increase in performance.

2.3.2 Alterations and adaptations of the use of under sampling the majority class

Japkowicz (2000) also considered random under sampling of the majority class and focussed under sampling in her study. While under sampling the data has the advantage of not using replicants of pre-existing (or in the case of SMOTE, artificial data) data, the amount of data available to create a model may be very small unless the dataset is very large. However it does have the advantage of creating a model on data which has been observed. When carrying out a comparison of oversampling and under sampling along with recognition based methods, Japkowicz (2000 b) found that recognition based methods were inferior to both under sampling and oversampling. More exotic forms of under sampling have also been investigated. In these methods the under sampling is focussed on the members of the majority class which are most useful in training the model to recognize the majority class. Kubat, Holte and Matwin (1998) investigated this focussed under sampling approach. In their study they used a discriminatory under sampling technique. This is applied to the majority class and keeps the number of samples in the majority class the same as the number of samples in the minority class. The minority samples are then divided into our categories, these are:

1. Noisy minority samples overlapping with the positive class decision space.
2. Borderline samples.
3. Examples which are not necessary.
4. Useful examples.

The borderline samples were found by the utilization of Tomek's links (Tomek 1976). Tomek's links is an adaptation of the K-NN learner. Tomek's links works by removing samples which are next to each other but have different labels. Tomek link can be described as taking two examples "x and y" which are members of different classes. Let d(x,y) be the distance between examples x and y. For (x,y) pair of examples is classed as a Tomek link if case z does not exist where the distance between x and z is less than the distance between x and y $(d(x, z) < d(x, y))$. Or if the distance between y and z is not less then the distance between y and x $((d(y, z) < d(y, x))$. If two examples are part of a Tomek link then one of the pair can be considered to be noise or both members of the pair are borderline samples. Similarly to oversampling, under sampling can be be easily implemented through a number of applications and is available in RapidMiner, Weka and R.

2.4 Use of SMOTE algorithm to artificially synthesize items belonging to the minority class

The use of the SMOTE algorithm to artificially synthesize items belonging to the minority class has also been postulated as a means of overcoming the class imbalance problem. SMOTE can be considered to be a particular case of over-sampling. SMOTE draws heavily from work carried out by Kubat,Holte and Matwin (1998) who used a type of discriminatory under sampling of the larger class while maintaining the number of examples of the minority class. The geometric mean was employed to quantify the level of performance of the classifier, this being representative of a lone point on the receiver operator characteristic curve. The minority instances were split into four different types, these being:

- Noise samples, having common characteristics with the positive class.
- Borderline examples detected through Tomek's links (Tomek 1976) concept of nearest neighbour.
- Superfluous or unnecessary examples.
- Safe samples.

When 'SMOTING' a dataset, the class having the smaller number of examples is oversampled through the synthesis of artificial instances, as opposed to oversampling existing samples with replacement. The class having the smaller number of examples is over-sampled by the use of a K-NN to add artificial instances along the line segments connecting some or the entire population of nearest neighbours. The number of nearest neighbours to add is dependent on the level of over-sampling required, for example if a level of over-sampling of 200 per cent is required. Two nearest neighbours are selected and a single example is created in the direction of the nearest neighbour. The (Chawla et al. 2000) original implementation used a total of five nearest neighbours. Artificial samples are created in the following manner, the level of dissimilarity between the sample item feature and its nearest neighbour are calculated. This is then multiplied by a random number chosen between 0 and 1 and then this is added to the specific sample feature. This results in the selection of a random point along a line segment between the two features under consideration (the sample feature and its nearest neighbour). This ensures the generalization of the decision area of the minority class having smaller number of examples. SMOTE has been criticized in

some quarters as it creates artificial minority samples without taking into account majority class samples, which can lead to an over generalization of the data. The algorithm is given as follows (Chawla et al. 2000):

```
Inputs

>D:the amount of the duplication(D%).

>K: the amount of of nearest neighbours to be used to synthesize the

>balanced minority class.

> Step 1:
> Randomize the members of the under-represented class.

> Step 2:
> For each member of the minority class:
> Locate k  the number of nearest neighbours.

>  For j=1:d/100

> Choose one of these neighbours.

> Create a new instance which is positioned on the line segment
> connecting the group of
> minority samples  and the chosen neighbour.

>   End

> End
```

In code SMOTE is implemented in R using code similar to below:

```
goitresmote <- SMOTE(thyrprob ~ .,th12,k=5,perc.over = 700)
```

Here we are applying the SMOTE algorithm to the thyroid dataset(th12) and specifying five nearest neighbours and the percent over we want to oversample the minority class (thyrprob). The code is adopted from (inside-R 2012). All R code to apply the SMOTE algorithm to the three datasets surveyed are included in appendix 1.

2.5 The implementation of cost sensitive learning

In the case of classification data mining projects the cost of misclassification of a particular class can be extremely expensive. In financial fraud this can be calculated in terms of monetary losses or in the domain of medical diagnosis, the non diagnosis of dangerous diseases which may lead to death. Normal machine learning algorithms are built to deliver maximum accuracy. When these classifiers are used in a binary class imbalanced classification problem, they can result in building a model which classifies all samples as members of the majority class. One way of addressing this mis-classification problem is to train a cost sensitive mis-classification learner where the cost mis-classifying the minority class is far greater than that of the majority class. There are two general methods for doing this:

1. The use of learners that are cost-sensitive.

2. The design of general approaches for turning normal classifiers into cost sensitive ones.

An example of the latter is Metacost. Metacost (Domingos 1999) can be seen as a wrapper which is used in conjunction with a classifier. A cost matrices can be seen as a method for capturing the cost of misclassification of examples of one class as belonging to another. C(i,j) can be described as the cost of predicting a record that is actually from class i as belonging to class j. Therefore using this notation the cost of generating a false negative error is denoted by C(+,-), while C(-,+) is the cost of generating a false positive. A negative value in the cost matrix represents a reward for making correct classifications. The overall cost of a model is thereby given by:

$$Ct(M) = TPXC(+,+) + FPXC(-,+) + FNXC(+,-) + TNXC(-,-) \tag{2.1}$$

When a cost sensitive approach is used, the above cost matrix is utilized to generate the a model with the lowest cost. This is the approach the Metacost learner takes. In explaining the MetaCost (Domingos 1999) algorithm it is important to note that the conditional risk R(i|x) is equal to the anticipated price of predicting that example x belongs to class i. As the Bayes optimal prediction for example x is the class I that reduces as much as possible the conditional risk. Therefore for C(i,j) which is the cost of predicting that an example is of class i when actually it belongs to class j and the probability of x belonging to class j. The Bayes best prediction for example x is the class that keeps to a minimum the conditional risk equation (Duda and Hart 1973), this is shown below .

$$R(i|x) = \sum_{J} P(j|x)C(i,j) \tag{2.2}$$

This results in the cause of a division of the example space x into j number of divisions in space so that for class j , class j is the lowest cost prediction in the j division in space. The aim then for cost-sensitive learning is to locate the borders between these divisions. If the samples used to train the model were now named according to their most accurate class as provided by the cost matrix, a error based learner could be utilized to find the location of these idealized borders of the divisions of the sample space. This is the approach the Metacost procedure is built on. However to rename the samples in the training set with their optimal classes, a means of deriving their class probabilities must be found. Metacost accomplishes this by training multiple models and then utilizing each class's percentage of the total vote as an approximate of its probability. This is somewhat similar to bagging (Breiman 1996) which is itself derived from the concept of model averaging. In bagging, multiple samples (with replacement) are taken from the dataset following a consistent probability distribution. With each bootstrap being the same number of samples as those in the original dataset. As the sampling is carried out with replacement, a number of the samples will appear multiple times within the training set of data, while some others may not appear at all in the the training set. As each sample has a probability of $1-(1-1/N)^N$ (where N is the number of samples) of appearing in each bootstrap sample, for sufficiently large values of N this will converge to a value of 0.63. The bagging algorithm is given as follows:

```
> Let k be the number of bootstrap samples

> For  1 to k do

> Take a bootstrap sample of Size N, $D_i$
> (this will hold approximately sixty three
> per cent of the original data)
```

```
> Learn a model based on the the bootstrap sample provided

> End For

> In the case of linear models average the output
> or in the case of classification vote for the best model.
```

Therefore bagging can be said to decrease the error associated with generalization by lowering the variance of the base classifier. While sharing some similarities to Metacost, Metacost does not follow exactly the same methodology in terms of sampling approach and the number of samples in each re-sample can be less than in the original training set. However similar to bagging, Metacost creates a number of bootstrap samples of the training set and trains a model on each. It approximates every class's probability based upon the percentage of votes it received and then by applying the Bayes optimal prediction of example x belonging to the class I being minimized according to its conditional risk (Duda and Hart 1973) to reassign each example with its approximated correct class and then retrain the learner using the relabelled data. This type of methodology ,while reported in the literature to be vastly superior to sampling methods, suffers from a number of disadvantages. The most problematic being that the cost of misclassification must be known in advance. However there are techniques to generate optimal values for the cost of misclassification, such as genetic algorithms, these can be very computationally expensive to apply. Without accurate knowledge of the cost of misclassification, use of cost sensitive methods may result in over fitting to the training set. In terms of advantages to using the Metacost procedure, it can be widely implemented across a number of learners.

2.6 Cluster based over sampling (CBOS) and cluster based under sampling (CBUS)

CBOS is an holistic approach to overcome between class and within class skewed data problems. CBOS tries to locate smaller sets of a class that are removed in the feature space form the main body of examples (Jo and Japkowicz 2004). These smaller subsets of data are referred to as disjuncts and if included in a training set result in poorer predictive accuracy. These small disjuncts have been found to be more error prone than bigger disjuncts (Holte, Acker and Porter 1989). What makes these more prone to error from a classifier is that classifiers are very strongly biased towards them. Through the use of unsupervised techniques, samples that belong to these disjuncts are located. By oversampling from non disjuncts these examples can be removed, ensuring a more representative sample. (Yen and Lee 2009) have also experimented with cluster based sampling methods in their implementation of CBUS. Their approach is based on clustering of all the examples in the training set into a number of clusters. By examining each clusters distribution of majority class to minority class examples, it can be determined whether each cluster is representative of a majority class sample or minority class sample. If a cluster contains more majority samples, all samples in that cluster will act like majority samples. Similarly if a cluster contains more minority class samples as opposed to majority class samples, samples within the cluster will behave like minority samples. Therefore their approach to sampling involved selecting samples from clusters based on the ratio of samples of the majority class to samples of the minority class label.

2.7 Boosting

Under sampling of the minority class has been shown to be an extremely useful method for treating the class imbalance problem. It is described mathematically as follows: Given the minority sample set X and the Majority sample set Y, under sampling reduces the sample set Y by randomly sampling the

majority class till the number of samples in Y' equals the number of samples in X. However this can cause problems as potentially useful examples from the majority class are discarded, these samples can be represented as $Y \cap \overline{Y'}$. To overcome this problem a number of methods have been suggested based on boosting , one of the most simple methods is EasyEnsemble (Liu et al. 2009). Boosting is the process of taking a number of weak learners and from them synthesizing a strong learner. AdaBoost is the most widely cited boosting algorithm and is employed to improve the performance of weak or poor performing learners. EasyEnsemble is one of the simplest methods to include samples from the majority class which are not in the training set, $Y \cap \overline{Y'}$. To overcome this a number of independent subsets Y' ($Y_1..Y_T$) are derived from the majority sample set Y. For each subset a classifier is trained using Y_i and P (a set minority class samples) and all synthesized classifiers are then amalgamated to form a final decision. The EasyEnsemble algorithm is shown below:

Inputs: A sample set of the minority class X, a set of majority class examples Y, such that $|X| < |Y|$ and Z, the number of subsets to be selected from Y.

i¡-0

repeat

i¡-i+1

Randomly sample a subset Y_i from Y, so that:

$$|Y_i| = |X| \tag{2.3}$$

Utilizing Adaboost learn a model H_i using X and Y_i H_i is the resulting ensemble of s_i number of classifiers $h_{i,j}$. The threshold is equal to :

$$\theta_i$$

$$H_i(x) = sgn\left(\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \theta_i\right) \tag{2.4}$$

Until i=T

Output: the resulting output is an ensemble of learners

$$H(x) = sgn\left(\sum_{i=1}^{T}\right) \sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j} - \sum_{i=1}^{T} \theta_i \tag{2.5}$$

Sun et al. (2007) has adapted the AdaBoost algorithm for cost sensitive learning in their Realboost algorithm. In the Realboost implementation, the performance of the weak classifiers are increased through the use of response binning. This is based on the Haar-features (Viola and Jones 2001).

2.8 Use of learners which can cope with class imbalance in the data

Weighted random forests were developed due to random forest learners being more inclined to the majority class when learning in skewed datasets (however random forests have been cited as being resistant to the class imbalance problem (Berkeley 2012)). A variation of the random forest learner has been developed where a heavier penalty is placed on misclassifying the class with fewer examples (Chen, Liaw and Breiman 2004). This is done by placing or assigning a weight to each class. The weights assigned to each class are utilized in two different places in the tree induction procedure. In the induction stage, classes associated with each weight are used to bias the splitting criteria. In creation of the terminal nodes, this is done by using the "weighted majority vote", the weighted vote being the weight of the class multiplied by the number of cases of that class. Another algorithmic approach to solving the class imbalance

problem is through the use of simple rule induction (RI) learning. An example of this is the use of the RIPPER algorithm. The RIPPER algorithm is a rule induction system which makes use of a divide and conquer strategy to create a series of rules which describe a specific class. In doing so it builds a series of rules for each class even very rare classes. Raskutti and Kowalczyk (2004) have shown its particular use, especially with the highly skewed noisy datasets containing many dimensions. Raskutti and Kowalczyk (2004) claim that there method is similar to feature selection but offers a number of advantages namely it is far less resource hungry in its deployment. The RIPPER algorithm is available in the Weka data mining tool and can be run from RapidMiner. K-NN has been internally modified to desensitize it to the class imbalance in datasets through the use of a weighted distance function. The weighted distance function is used to make up for the skew in the dataset between the classes. As opposed to a normal weighted K-NN learner the weights are applied to classes and not to each separate prototype. This results in the weighting factor being the most for the majority class when compared to the minority class. This causes a decrease in the distance to prototypes of the minority class when compared to prototypes of the majority class. This leads to the model locating their nearest neighbour amongst prototypes within the minority class (Barandela et al. 2003). Similarly, the SVM (Support Vector Machine) (Vapnik 1998) can be internally modified so that the constructed hyperplane is moved a greater distance from the minority class. This acts to counterbalance the class imbalance which causes the hyperplane to be closer to the minority class. SVM's are built on a rather straightforward concept borrowed from statistics. SVM's simplicity derives from although it is a linear algorithm in a feature space consisting of a large number of dimensions. It does not involve calculations to be carried out in this feature space which consists of a large number of dimensions. SVM's work by partitioning the different classes of data through the use of a hyperplane. The implicit mapping Φ is used to transform the input data into a feature space consisting of a large number of dimensions through the use of a kernel function. This Kernel function returns the product of x and x' the point in the feature space. The hyperplane is given by the function:

$$(w, \Phi(x)) + b = 0 \tag{2.6}$$

This is related to the decision function:

$$f(x) = sign((x, \Phi(x)) + b) \tag{2.7}$$

The hyper-plane which results in greatest classification performance is the one which delivers the maximum degree of separation of the two classes. The hyperplane is created through the solution to the constrained optimization problem which has the solution with the expansion $w = \sum \alpha_i \Phi(x_i)$ describing a set of training patterns that lie on the margin. These are referred to as training patterns. The final decision function is conditional on the dot products between the training patterns. The coefficients α_i are derived by finding the solution to the quadratic programming problem.

$$\textbf{maximize } :W(\alpha) = \sum_{i=1}^{m} -\frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \tag{2.8}$$

$$0 \leq \alpha_i \leq \frac{C}{m} (i = 1, ..., m) \tag{2.9}$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \tag{2.10}$$

The variable C (Karatzoglou, Meyer and Hornik 2006) which is related to the cost of misclassification and therefore how complex the resulting kernel function will be (see equation above). The variable C determines the cost of the SVM classifying incorrectly a training point and therefore determines how complicated the prediction function will be. A large value for C will result in the formulation of a kernel function of sufficient complexity so as to classify incorrectly the smallest number of training samples.

2.9 Summary

The class imbalance problem, while being a major problem in fraud detection studies, it is also a problem in a number of different disparate domains. These domains include quality assurance, medical diagnosis and credit scoring. There are two different approaches to overcoming the class imbalance problem. The first approach is known as data methodologies, which involves the use of various sampling techniques to rebalance the dataset, thereby removing the imbalance in the dataset. The sampling methods maybe focussed or random in nature. They can also involve the synthesis of new artificial replicants of the minority class in the case of SMOTE. The second approach can be deemed an algorithmic approach and involves either introducing a cost of misclassification to the learner or else using learners that are resistant to the class imbalance problem. These learners are either resistant to the class imbalance problem through inherent properties of the learner, as in the case of the RIPPER algorithm, or else are hardened against the problem through internal modification as in the case of K-NN or the SVM learners.

Chapter 3

Implementation of methods to overcome the class imbalance problem

3.1 Introduction

RapidMiner, R and Weka were used to study the various methods for overcoming the class imbalance problem. As outlined in the previous chapter there are two approaches to dealing with the class imbalance problem, these are data methods and algorithmic methods. Data methods utilized in this study include:

1. ROS (Random over sampling of the minority class). ROS can be described as the random sampling of the minority class with replacement. This randomly samples with replacement the minority class and adds them to the minority class sample-set until the size of the minority class is the same size as the majority class. With replacement means that after each re-sample the samples are placed back in the 'pot' (the minority sample set) and can be re-sampled again.

2. RUS (Random under sampling of the majority class). RUS can be described as the removal of samples till the number of the samples of the minority class equals the the number of samples of the majority class.

3. ROS/RUS (Random over sampling of the minority class/Random under sampling of the majority class). ROS/RUS is a combination of random over sampling and random under sampling, in the case of all experiments carried out the ratio of random over sampling to random under sampling is 50/50 (ling and Li 1998).

4. SMOTE. Synthetic Minority Over Sampling Technique can be considered a special case of over sampling, with the SMOTE algorithm used to create a number of artificial replicants of the minority class samples (Chawla 2003).

All methods except SMOTE were implemented directly in RapidMiner, while SMOTE was implemented in R and the 'SMOTED' dataset (the dataset with the SMOTE algorithm applied to it) was then imported to RapidMiner. It should be noted that through the use of the R-extension, the SMOTE algorithm of the DMwR (Torgo 2012a) package could be called directly through RapidMiner. The algorithmic methods used involved attaching a mis-classification cost to each class, the algorithmic methods for dealing with the class imbalance problem were:

1. The use of the Metacost procedure. The Metacost procedure considers the learner it is going to be used in conjunction with as a black box as it does not need any information regarding how the

learner works. Metacost wraps a procedure around the learner, making any learner cost sensitive. The use of the Metacost procedure is implemented as described by Dataprix (Dataprix 2012a)

2. The use of Metacost thresholds is implemented as described by (Dataprix 2012b).

With the consumer fraud and Thyroid dataset, additional algorithmic methods were experimented with which involved using learners which are not susceptible to the class imbalance problem, such as simple RI methods like Ridor or RIPPER. These were also combined with ensemble methods or with other classifiers to boost their performance.

3.2 The datasets used in the study

To asses the different methods, a number of different datasets were utilized. Due to the scarcity of datasets on the subject of fraud, only two of the datasets used were on fraud, the third dataset was on the Thyroid disease and related thyroid based problems. The three datasets used in the study below are (these are described in more detail in each of the chapters describing the experimental results):

1. A car insurance fraud dataset (Phua 2004).
2. A consumer fraud insurance dataset (Torgo 2011).
3. A Thyroid disease dataset (Quinlan 1986 a and b) and downloaded from (UCI 2012).

3.3 Assessing the performance of a learner

As previously explained in chapter three, predictive accuracy is a poor accuracy measure for class imbalanced datasets. This is best shown using an example. Take a dataset with a binary label, 1000 tuples, within the dataset there is a very large class imbalance, with the ratio of the majority class to the minority class being 99:1. Therefore there are a 990 class items a and 10 class items b. Take for example if none of class b have been correctly populated the predictive accuracy would be calculated to be (990+0)/(990+0+10+0), this results in a predictive accuracy of 99 percent. This shows how poor a measure of performance predictive accuracy is when analysing performance in skewed datasets. Therefore predictive accuracy cannot be used to asses the performance of imbalanced datasets.

Table 3.1: Contingency table showing the four possible outcomes.

	Predicted Negative	Predicted Positive
Actual Negative	TN	FP
Actual Positive	FN	TP

where:
TP = number of true positives, FP = number of false positives, FN = number of false negatives, TN = number of true negatives.

Alternatives must be used when considering performance measures for assessing the performance of a learner on a class imbalanced dataset. One measure that is useful when working on imbalanced datasets is the F-measure. The F-measure is the harmonic mean of the precision and recall score (Han 2005). Precision, also referred to as the positive predictive rate, is the amount of positive predictions that were

correctly made over the number of correct positive predictions and incorrect positive predictions (Myatt and Johnson 2009). It can be represented by the following equation:

$$Precision = (TP)/(TP + FP) \tag{3.1}$$

(0)/(0+0)=0

Recall, also known as sensitivity, is equal to the number of true positives divided by the sum of the number of true positives plus the number of false negatives:

$$Recall = (TP)/(TP + FN) \tag{3.2}$$

(0)/(0+10)=0.

Therefore the F-measure is calculated from the following equation:

$$F - measure = 2 * (Precision * Recall)/(Precision + Recall) \tag{3.3}$$

F=0.

Another useful measure when assessing the performance of learners with imbalanced datasets is ROC (Receiver Operator Characteristics) curves. The ROC curve, which was first used in signal processing, is a plot of true positive rate against the false positive rate.

A ROC curve for a purely random classification would be a straight line running from the the origin to point (1,1). Better performance than a random guess will lie above the straight line.

The AUC (Area Under the Curve) value, which is the area beneath the ROC curve, can also be used as performance measure. It is related to the ranking quality of the classification Wilcoxin-Mann-Whitney statistic. However ROC curves can only be constructed for classification problems were there is a bi-nominal label, however as one of the datasets is a poly nominal classification problem, AUC values cannot be used and the F-measure is used instead. Other useful measures that were not used are specificity, sensitivity and G-mean. The sensitivity measure is the number of true positives divided by the sum of the true positives and false negatives. Specificity is the number of true negatives divided by the sum of the number true negatives and false positives (Tang et al. 2006). G-mean (Vapnik 1998) is the square root of the product of the specificity and sensitivity. The three equations to calculate sensitivity, specificity and g-mean are given below:

$$sensitivity = TP/(TP + FN) \tag{3.4}$$

$$specificity = TN/(TN + TP) \tag{3.5}$$

$$G - Mean = \sqrt{sensitivity * specificity} \tag{3.6}$$

3.4 Implementing the data pre-processing methods

3.4.1 RUS (Random Under Sampling of the data pre-processing methods)

RUS was implemented by simply selecting a random sample of the majority class which matches the number of minority class examples. Random under sampling of the majority class was implemented using the methodology shown below.

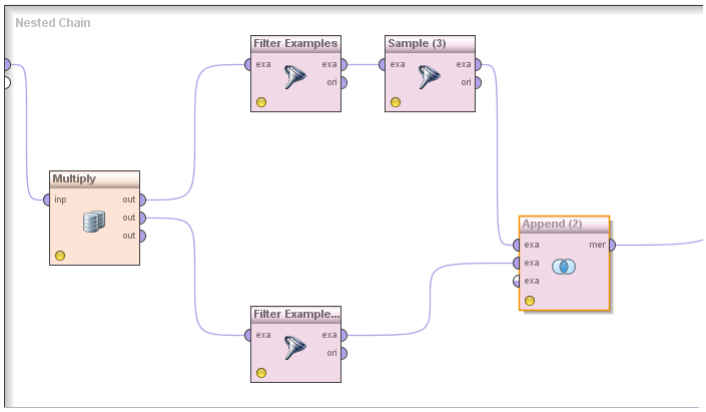

Figure 3.1: Random under sampling of the majority class in RapidMiner.

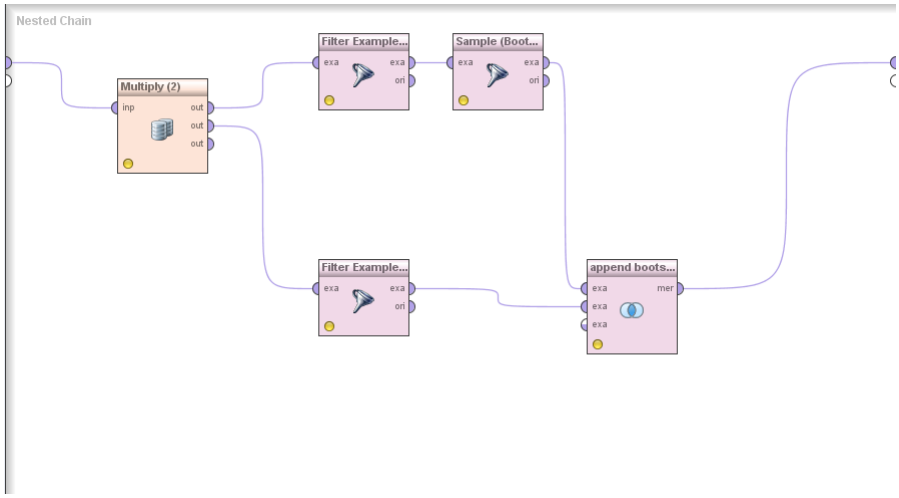

Figure 3.2: Random over sampling of the minority class in RapidMiner

3.4.2 ROS (Random Over Sampling of the minority class, through the use of the bootstrapping operator)

ROS was implemented by creating a process which creates a number of replicants of the minority class a equal to the number of samples of the majority class. This is done through the use of the RapidMiner bootstrapping operator. This operator creates a bootstrapped sample from the original dataset by creating a number of re-samples of the original dataset though the use of random oversampling of the minority class with replacement (rapid-i.com 2012).

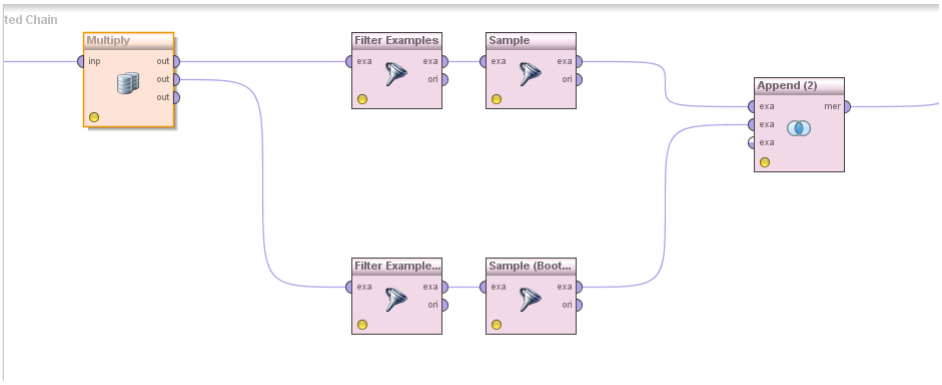

Figure 3.3: Random over sampling of the minority class/Random under sampling of the majority class method in RapidMiner.

3.4.3 The ROS/RUS (Random Over Sampling of the minority class (through the use of the bootstrapping operator) and the Random Under Sampling of the majority class)

The ROS/RUS involves an adaptation of the above method, where a sample size of the majority and minority are chosen according to what ratio the user requires. Unless stated otherwise a 50/50, ROS/RUS sample was chosen.

3.4.4 SMOTE

SMOTE was implemented through R. The dataset had the SMOTE algorithm applied to it until the number of minority samples equals the number of majority samples. The SMOTE algorithm was applied using R (see appendix 1 for the R code used to apply the SMOTE dataset).

3.5 Implementing the algorithmic pre-processing methods to deal with the class imbalance

3.5.1 Metacost

Metacost is implemented in RapidMiner as a wrapper class where the Metacost procedure is implemented as a wrapper around the learner. The costs for mis-classification must be empirically derived, but (in the case of the car insurance fraud dataset) as a starting point approximations of the average cost per missed fraudulent claim was estimated at between approximately 2300 US dollars to 2600 US dollars (the misclassification cost of fraud)and the cost of an unnecessary investigation at approximately 200 US dollars (the misclassification cost of non-fraud) can be used. Phua et al. (2004) has recommended using a ratio of 2640 (cost of misclassification of fraud) to 203 (misclassification of a non-fraudulent case) for the car insurance fraud dataset, this corresponds to a ratio of misclassification of majority class to minority class of (0.07689). These values however were only useful as a baseline and the optimal values had to be empirically derived. For all other datasets misclassification costs must be empirically derived.

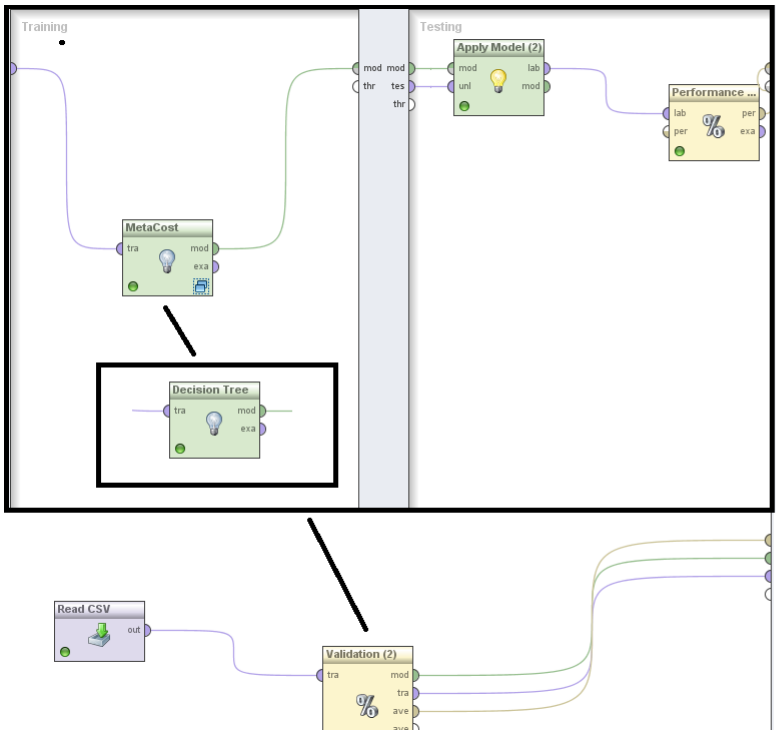

Figure 3.4: Metacost wrapper implemented in RapidMiner.

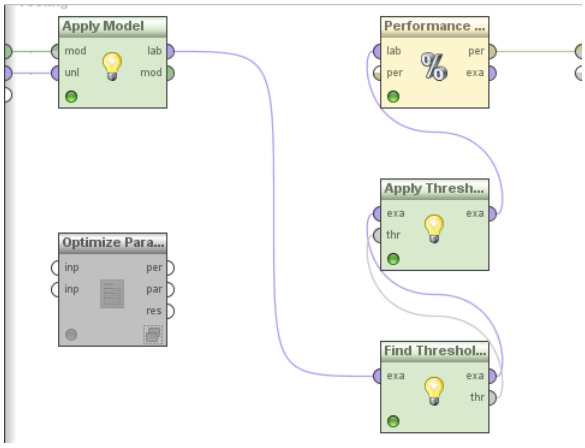

Figure 3.5: Implementing the misclassification costs,Metacost threholds implemented in RapidMiner.

3.5.2 Metacost thresholds

Metacost thresholds are implemented in RapidMiner using the ”find threshold” operator to derive a threshold for a provided prediction confidence, the costs of misclassification and probability distribution to result in optimal classification. Similarly to the Metacost procedure these values must be empirically derived. The workflow for using metacost thresholds is shown below and is adopted from that on (Dataprix 2012 b).

3.6 Different methodologies for evaluating the data methods, learnings from the car insurance dataset

A number of learners were evaluated with data-centric methods, these included ID3, K-NN, Naive Bayes, C4.5 (Weka WJ-48), decision tree and random forests. However a number of problems were noted when using a number of different experimental set-ups. These problems were highlighted when comparing the results obtained when using 10 fold cross validation using the balanced dataset to train and test the model and a process which used the balanced data to train the model (again using 10 fold cross validations) and a portion of the original data (which had not been used to train the model) to evaluate the performance of the model. A 70:30 split was used (70 per cent of the data used to train the model and 30 per cent of the data used to test the model). As mentioned previously, originally all experiments were ran using the 10 fold cross fold validation operator with both the same dataset (i.e the one that has been modified except in cases where no pre-processing step applied) used to train the model and test the model. However this caused a number of problems when evaluating the performance of the learner. A second method methodology was then tried which utilized a 70:30 split to train the model and evaluate it. The latter is shown in figure 3.12. This experimental approach was chosen as it still allowed the use of 10 fold cross validation to be used to train the model. It also allowed a portion of data which was not used to train the model to be used to asses the performance of the model. The results are collated in table 3.2. When examining the results obtained by training and testing the data with the balanced and unbalanced data a number of anomalies become clear. On going from (a) training and testing the model with the same balanced dataset. (b) Training with just the balanced dataset and assessing the performance of the model with the a ’blind’ (blind as it was not used in training the model) sample of the original dataset. a number of issues arise are seen to arise with measuring the performance.

A substantial difference in the apparent performance for the RUS, SMOTE and ROS/RUS data methods across all learners is seen.

This difference in performance is best highlighted graphically and the performance differences are summarized below:

1. The K-NN learner showed substantial differences in performance when training and testing the model using the balanced dataset for RUS and ROS methods. For the SMOTE algorithm, values were approximately the same, see figure 3.6.

2. The Naive Bayes learner showed a large difference in performance for all data methods see figure 3.7.ROS, ROS/RUS and SMOTE showed a large fall off in performance when evaluating using; (a) the balanced dataset or (b) the blind

3. The random forest learner showed a sizeable difference in performance measure for RUS and SMOTE and less so far ROS/RUS, see figure 3.8.

4. The C 4.5 learner showed a sizeable difference in performance measure for all data methods, see figure 3.9.

5. The ID3 learner showed a sizeable difference in performance measure for RUS and SMOTE and less so for ROS and ROS/RUS see figure 3.10.

6. Similarly the decision tree learner showed a sizeable difference in performance measure for all learners, see figure 3.11.

From this analysis, it can be seen that all methods surveyed showed a sizeable difference in performance when evaluating the performance using the balanced dataset compared to when using a 'blind sample' of the original dataset.

Taking first the analysis of the first methodology (training and testing the model using the same balanced dataset). The results obtained by this methodology are represented by the blue bars in the figures 3.6, 3.7, 3.8, 3.9, 3.10 and 3.11. Across all learners surveyed, RUS delivered the poorest performance. In the case of RUS, this can be explained as training the model with much fewer majority samples, a large proportion of the data in the original dataset is ignored. Thereby when training the model, the model is exposed to a lower number of the majority cases. This results in the model being unable to recognize members of the majority class, thereby resulting in the misclassification of majority class members (not fraud) as members of the minority class (fraud). This increase the number of false positives (FP). Resulting in the lowering of the precision of the learner (Batista et al. 2004). This problem is also somewhat apparent in the case of ROS/RUS. In the case of ROS/RUS not as many of the majority class are removed so the problem is not as severe and the model is still able to recognize the majority class, albeit not as proficiently as if all (or a sizeable number of samples from the majority class remained) of the majority class remained. While for SMOTE (Chawla et al. 2000) a further complication is added as artificially synthesized data is used. If the SMOTE algorithm does not prove accurate at synthesizing accurately, examples (Wang and Japkowicz 2004) of the minority class, problems may occur. When training the model with the 'SMOTED' data it will deliver a model which is accurate at classifying minority classes with attributes which are not atypical of the minority class when compared to the attributes of the majority class. Thus when testing the model on (a blind sample of) the original dataset, the model is unable to recognize the majority class and confuses them for the minority class. This results in a high number of false positives, thereby delivering poor precision and subsequently a poor F-measure for the model.

ROS delivered near perfect performance across all learners. However ROS has been cited as leading to the development of models which are over-fitted on the minority classes (Batista et al. 2004). RUS performed poorly for both methodologies. In the case of RUS, the effect of reducing the number of

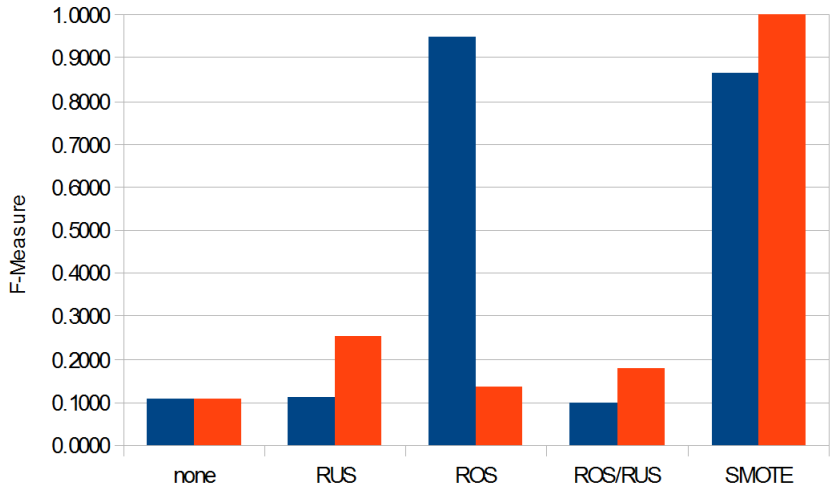

Figure 3.6: A plot of the effects on the F-measure of a number of data methods for dealing with the class imbalance problem, when training and testing with the same balanced dataset and training with the balanced dataset but testing the performance with the original dataset, using the K-NN learner.

majority samples results in the learner not having enough majority samples to train an effective model (Drummond and Holte 2003). This is due to the learner not being exposed to enough of the majority samples in the training dataset and ignoring most of the population of the majority class examples (as these are simply discarded) while training the model.

The second methodology proved problematic due to it removing a large portion of data (30 percent of the data was used evaluate the performance of the model) so as to evaluate the model. The results for this methodology are represented by the red bars in the figures 3.6, 3.7, 3.8, 3.9, 3.10 and 3.11.

This methodology proved to be problematic in the case of data methods that utilize an oversampling approach and resulted in them delivering an artificially low performance evaluation. This is due to a significant number of minority samples being excluded from the training set (as they are removed and used to evaluate the performance of the model), thereby not providing the learner with significant number of samples of the minority class to train an effective model. This was not a problem with the oversampling methods due to relative abundance of majority samples. It should be noted that this second methodology is similar to the one favoured by Phua et al. (2004) in his evaluation of a number of different methods for addressing the class imbalance in fraud detection. Kotsiantis et al. (2006) noted that excessively decreasing the amount of the minority class may result in a sample size that is not of ample size to train an accurate model. This results in the model not being able to recognize the minority samples (especially very rare cases) and this results in poor performance. To this end, all data methods were re-assessed, training the data on the balanced dataset using 10 fold cross validation to train the model and all of the original dataset to asses the performance of the model. The third approach proved to be the optimal experimental design for evaluating as accurately as possible the effectiveness of a particular data method. The third approach which involves training the learner on the balanced dataset but evaluating it on the original dataset avoids the problems of leaving out valuable minority samples which are necessary to train an effective model at classifying members of the minority class. The methodology is shown in figure 3.13. In conclusion, the first two methodologies proved problematic. The first because of training

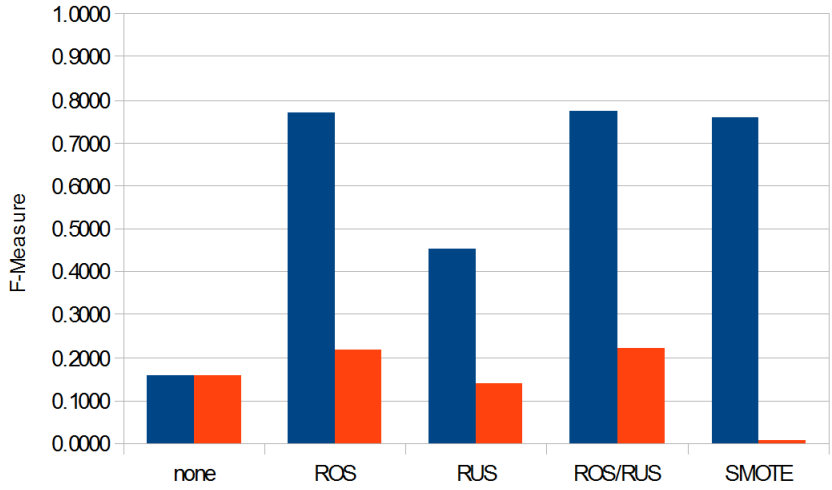

Figure 3.7: A plot of the effects on the F-measure of a number of data methods for dealing with the class imbalance problem, when training and testing with the same balanced dataset and training with the balanced dataset but testing the performance with the (blind sample of the) original dataset, using the Naive Bayes learner.

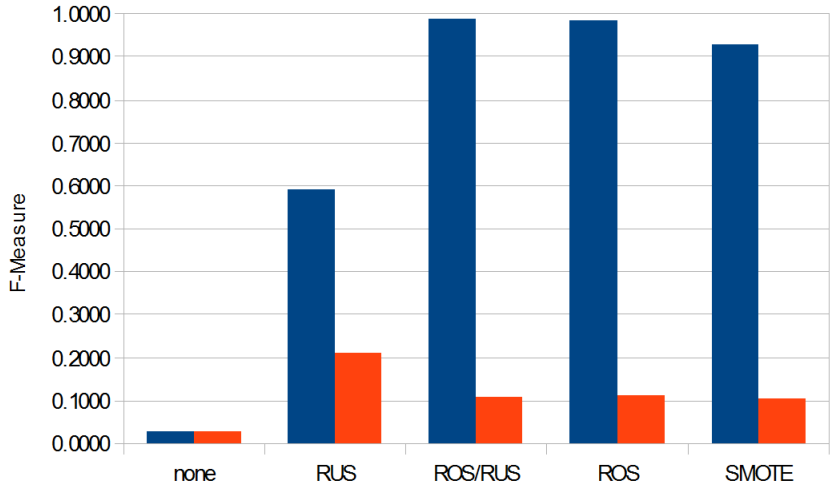

Figure 3.8: A plot of the effects on the F-measure of a number of data methods for dealing with the class imbalance problem, when training and testing with the same balanced dataset and training with the balanced dataset but testing the performance with the (blind sample of the) original dataset, using the random forest learner.

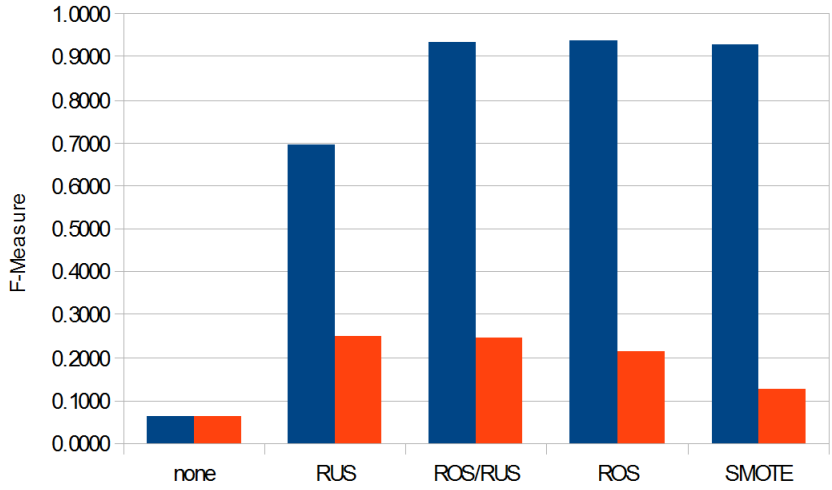

Figure 3.9: A plot of the effects on the F-measure of a number of data methods for dealing with the class imbalance problem, when training and testing with the same balanced dataset and training with the balanced dataset but testing the performance with the (blind sample of the) original dataset, using the WJ-48 (C4.5) learner.

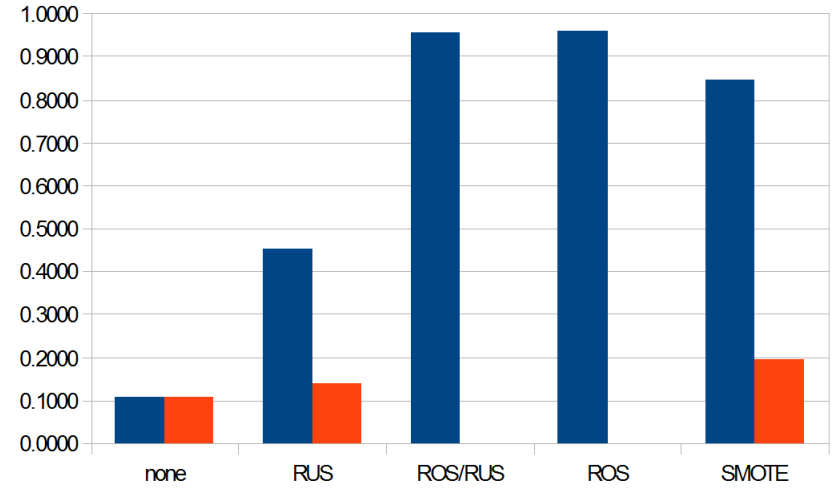

Figure 3.10: A plot of the effects on the F-measure of a number of data methods for dealing with the class imbalance problem, when training and testing with the same balanced dataset and training with the balanced dataset but testing the performance with the (blind sample of the) original dataset, using the ID3 learner.

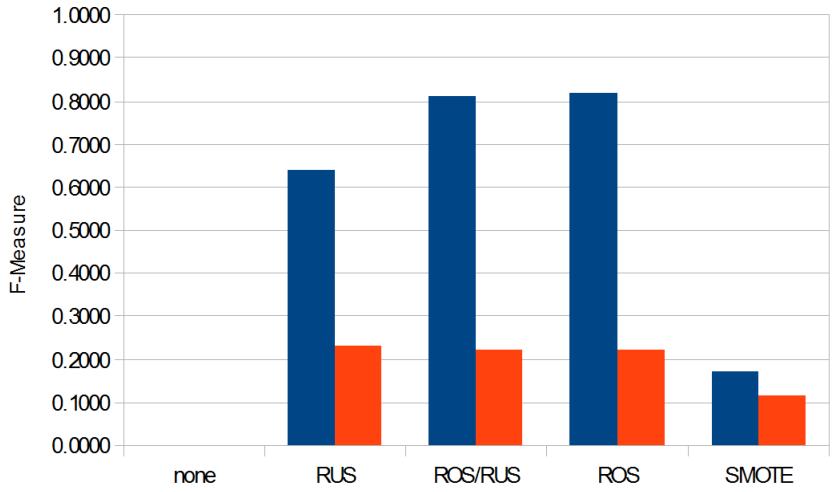

Figure 3.11: A plot of the effects on the F-measure of a number of data methods for dealing with the class imbalance problem, when training and testing with the same balanced dataset and training with the balanced dataset but testing the performance with the (blind sample of the) original dataset, using the decision tree learner.

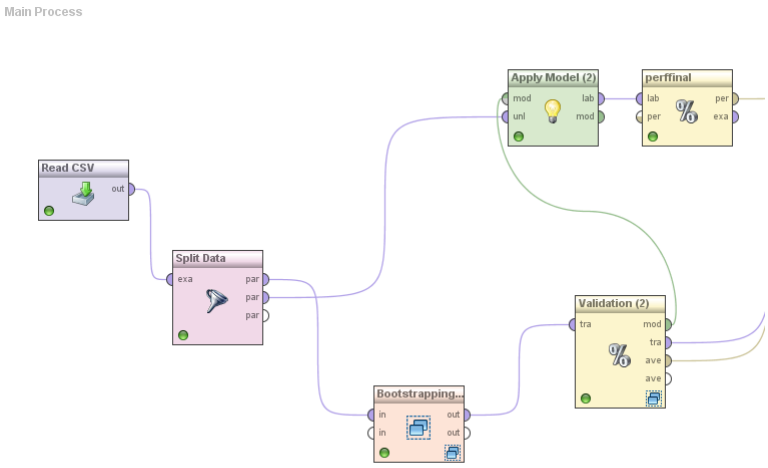

Figure 3.12: Training the dataset on the balanced dataset but evaluating its performance on the original dataset.

Table 3.2: Results of Precision, Recall and F-measure recorded when using both the balanced dataset for training and testing and when using the balanced dataset for training the model while using the a 'blind split' of the original dataset to test the model.

Learner	Methods	Precision(xval)	Recall(xval)	F-Measure(xval)	Precision	Recall	F-Measure
K-NN	none	0.106	0.1095	0.1077	0.106	0.1095	0.1077
K-NN	RUS	0.111	0.1073	0.1091	0.1457	1	0.2543
K-NN	ROS	0.9002	1	0.9475	0.1111	0.1661	0.1331
K-NN	ROS/RUS	0.9464	1	0.0996	0.0975	1	0.1777
K-NN	SMOTE	0.8861	0.8453	0.8652	1	1	1.0000
Nave Bayes	none	0.2024	0.129	0.1576	0.2024	0.129	0.1576
Nave Bayes	ROS	0.6918	0.8744	0.7725	0.1258	0.8736	0.2199
Nave Bayes	RUS	0.4829	0.426	0.4527	0.0808	0.4585	0.1374
Nave Bayes	ROS/RUS	0.6912	0.8783	0.7736	0.1262	0.8736	0.2205
Nave Bayes	SMOTE	0.7829	0.7345	0.7579	0.15	0.0033	0.0065
ID3	none	0.0981	0.1149	0.1058	0.0981	0.1149	0.1058
ID3	RUS	0.4833	0.4257	0.4527	0.0808	0.4585	0.1374
ID3	ROS/RUS	0.9172	0.9992	0.9564	-	-	-
ID3	ROS	0.9279	0.9991	0.9622	-	-	-
ID3	SMOTE	0.8555	0.8371	0.8462	0.1076	1	0.1943
C4.5	none	0.9688	0.0336	0.0649	0.9688	0.0336	0.0649
C4.5	RUS	0.647	0.7542	0.6965	0.1496	0.7292	0.2483
C4.5	ROS/RUS	0.8765	0.9948	0.9319	0.1899	0.3538	0.2471
C4.5	ROS	0.8857	0.9954	0.9374	0.1796	0.2671	0.2148
C4.5	SMOTE	0.8965	0.9636	0.9288	0.0682	0.9217	0.1270
random forest	none	0.2676	0.0141	0.0268	0.2676	0.0141	0.0268
random forest	RUS	0.6338	0.5573	0.5931	0.1298	0.556	0.2105
random forest	ROS/RUS	0.976	1	0.9879	0.1765	0.0758	0.1061
random forest	ROS	0.9739	1	0.9868	0.2639	0.0686	0.1089
random forest	SMOTE	0.9207	0.9409	0.9307	0.0535	0.9951	0.1015
decision tree	none	0	0	0.0000	0	0	0.0000
decision tree	RUS	0.6391	0.6391	0.6391	0.1518	0.4729	0.2298
decision tree	ROS/RUS	0.6967	0.9654	0.8093	0.1269	0.9531	0.2240
decision tree	ROS	0.6993	0.9839	0.8175	0.1269	0.9531	0.2240
decision tree	SMOTE	0.7848	0.0957	0.1706	0.0599	1	0.1130

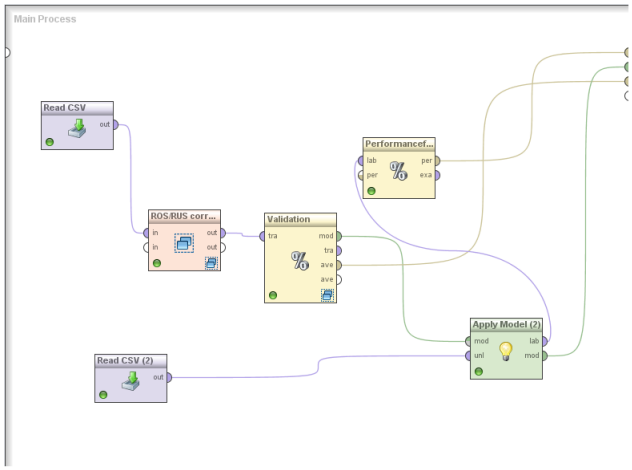

Figure 3.13: Workflow for assessing the performance of learner using the balanced dataset to train the model but the original dataset to asses the performance.

on the balanced dataset and testing on the same dataset causes a false measurement of the performance of the SMOTE data methods. The point regarding the over estimation of the performance of the SMOTE algorithm is again due to training the model with the same data that was used to test the model. As the SMOTE algorithm generates synthetic samples with no regard to the minority sample, there is danger of the SMOTE algorithm over-generalizing the minority samples. When the model is trained using these synthetic samples and tested using these samples it may be able to recognize the differences between the synthetic samples (of the minority class) and the majority class. However if the SMOTE algorithm has not created representative examples of the minority samples, when the model is tested against previously unseen majority class samples it is not able to accurately classify them. This results in the mis-classification of the majority samples as members of the minority class. This causes a high number of false positives. This results in a drop in precision, lowering the F-measure score. This explains the difference in the measured performance when evaluating the performance of the SMOTE data method using blind sample of data (i.e the data was not present when testing the data) to test the model. The drop in performance for the ROS learners was also noted when going from the first methodology to the second methodology. This can be explained in so far as when ROS is implemented none of the minority class is removed or no artificial replicants are introduced. Therefore the model has a full set of minority classes to train with on, thereby allowing it to train an accurate model.

Chapter 4

The study of the class imbalance problem and the car fraud dataset

4.1 Data understanding of the car insurance fraud dataset

The dataset used is a car insurance fraud dataset previously used (Phua et al. 2004). The dataset contains a total of 15420 examples with 14497 examples of non-fraud and 923 examples of fraud. The dataset includes 11338 examples from nineteen ninety four to nineteen ninety five (1994-1995) and 4083 records from 1996. The imbalance in the data has approximately 6 percent fraudulent cases and 94 percent non-fraudulent cases. The dataset has 6 numerical attributes, 25 categorical attributes and a binary class label, i.e fraud and non-fraudulent. The different attributes are shown below. The data quality of the dataset was excellent only requiring some very minor processing to correct. These were due to the presence of spelling mistakes of manufacturers names which were corrected. The spelling mistakes are Accura (Acura), Mecedes (Mercedes), Nisson (Nissan), and Porche (Porsche). These were corrected by loading the data into a database and applying an update statement to correct the particular spelling mistake.

4.2 Data preparation: The car insurance fraud dataset

Apart from the dealing with class imbalance in the dataset no other pre-processing methods were used. As previously discussed no considerable data quality issues existed in the dataset (only spelling mistakes of car manufacturers were detected). The following data methods were used:

1. ROS
2. RUS
3. ROS/RUS
4. SMOTE

The algorithmic method(s) used were the Metacost procedure.

Table 4.1: Results of Precision, Recall and F-measure recorded when using the balanced dataset for training the model, while using all of the original dataset to test the model.

Learner	Methods	Precision	Recall	F-Measure
K-NN	none	0.106	0.1095	0.1077
K-NN	RUS	0.1457	1	0.2543
K-NN	ROS	1	1	1.0000
K-NN	ROS/RUS	0.6125	1	0.0996
K-NN	SMOTE	1	1	1.0000
Nave Bayes	none	0.2024	0.129	0.1576
Nave Bayes	ROS	0.1277	0.8894	0.2233
Nave Bayes	RUS	0.1207	0.8917	0.2126
Nave Bayes	ROS/RUS	0.125	0.8938	0.2193
Nave Bayes	SMOTE	0.15	0.0033	0.0065
ID3	none	0.0981	0.1149	0.1058
ID3	RUS	0.19	0.9967	0.3192
ID3	ROS/RUS	0.6122	0.9989	0.7591
ID3	ROS	1	0.998	0.9990
ID3	SMOTE	0		0
C4.5	none	0.9688	0.0336	0.0649
C4.5	RUS	0.1511	0.9242	0.2597
C4.5	ROS/RUS	0.4123	0.9957	0.5831
C4.5	ROS	0.7183	1	0.8361
C4.5	SMOTE	0.0682	0.9217	0.1270
random forest	none	0.2676	0.0141	0.0268
random forest	RUS	0.1649	0.9967	0.2830
random forest	ROS/RUS	0.7486	1	0.8562
random forest	ROS	1	1	1.0000
random forest	SMOTE	0.0535	0.9351	0.1012
decision tree	none	0	0	0.0000
decision tree	RUS	0.1265	0.9588	0.2235
decision tree	ROS/RUS	0.1269	1	0.2252
decision tree	ROS	0.1314	1	0.2323
decision tree	SMOTE	0.0599	1	0.1130

4.3 Experimental methods for assessing the performance of the data methods for overcoming the class imbalance problem in the car insurance fraud dataset

Five learners were evaluated with the different data methods. These are the K-NN, Nave Bayes, ID3, C4.5 and random forest learners. The results are contained within table 4.1.

Examination of the table 4.1 shows that the methods utilizing over sampling delivered the greatest levels of performance increase. The methods that delivered the poorest levels of performance were those methods that utilized under sampling of the majority class. Also it should be noted the poor performance of the SMOTE algorithm. The best in class and worst in class methods are shown in table 4.2.

The success of the ROS methods in increasing the performance of the different learners can be ascribed to it utilizing all the members of the majority class and the random replication of the minority class

Table 4.2: Best and worst data methods.

Learner	BEST	WORST
K-NN	ROS	RUS
Nave Bayes	ROS	SMOTE
ID3	ROS	SMOTE
C4.5	ROS	SMOTE
random forest	ROS	SMOTE
decisiotn tree	ROS	SMOTE

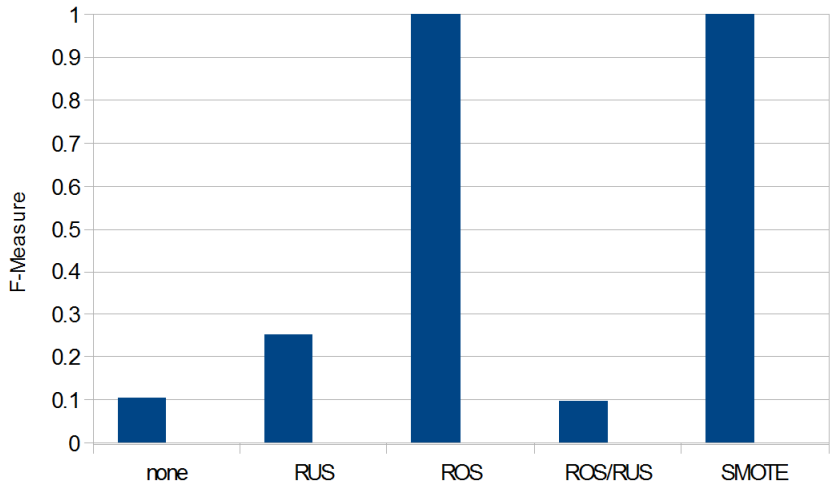

Figure 4.1: The effects of the different data methods on the K-NN learner with the car fraud insurance dataset.

until it is the same size as the majority class. This results in an increase of the feature space of the minority class to an equal size of the majority sample. This redistribution of the size of the classes provides the learner sufficient samples so as to be able to train a model to recognize the minority class. However the near perfect performance of the ROS method must be tempered by the fact that over sampling can result in over fitting (Guo et al. 2008). RUS and SMOTE proved to be the least useful methods in overcoming the class imbalance issue. The poor performance of the SMOTE algorithm can be put down to it creating minority samples which are over generalized with regards to the majority class members. The SMOTE algorithm utilizes a K-NN learner to replicate artificial replicants of the minority class. It does this without regard to the majority class. This can lead to the over generalization of the minority class. This over generalization leads to the minority class attributes being to similar to the majority samples for the learner to differentiate between the two. The methods that utilized a under sampling approach also delivered poor performance. When carrying out an under sampling approach, the members of the majority class are discarded until the number of the samples equals the number of minority class samples. This can result in discarding potentially useful samples when training the classifier.

The effect of the different data methods on the performance of the different learners are shown graphically in figures 4.1, 4.2, 4.3, 4.4 , 4.5.

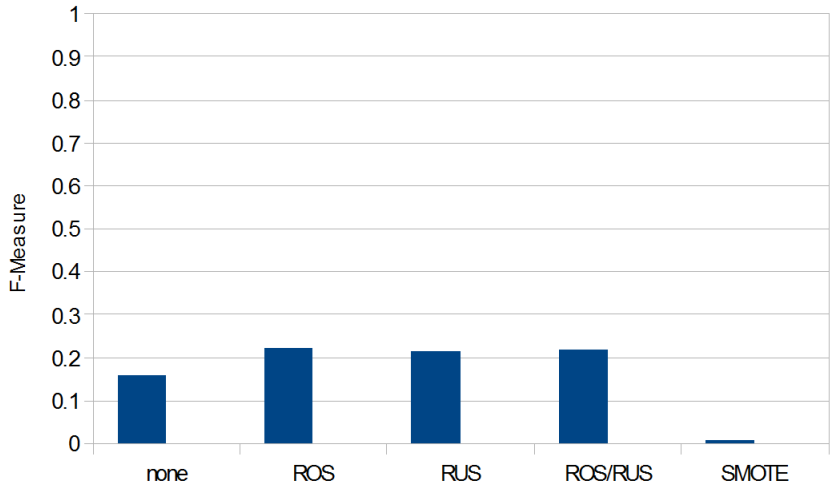

Figure 4.2: The effects of the different data methods on the Naive Bayes learner with the car fraud insurance dataset.

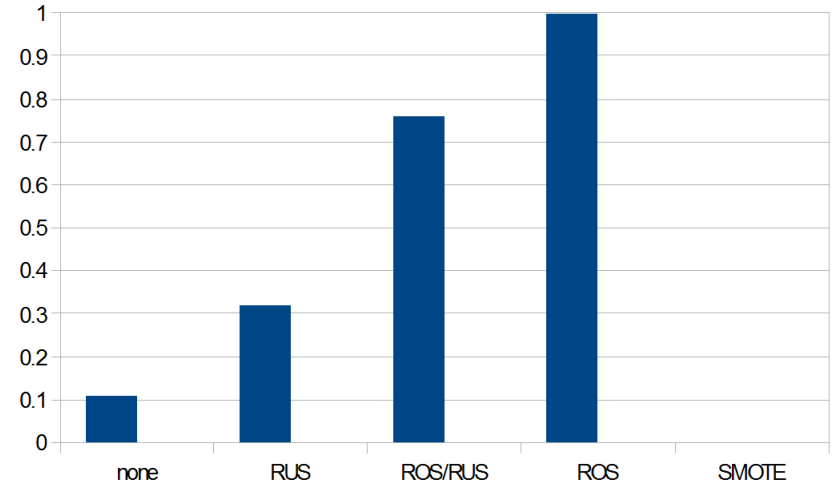

Figure 4.3: The effects of the different data methods on the ID3 learner with the car fraud insurance dataset.

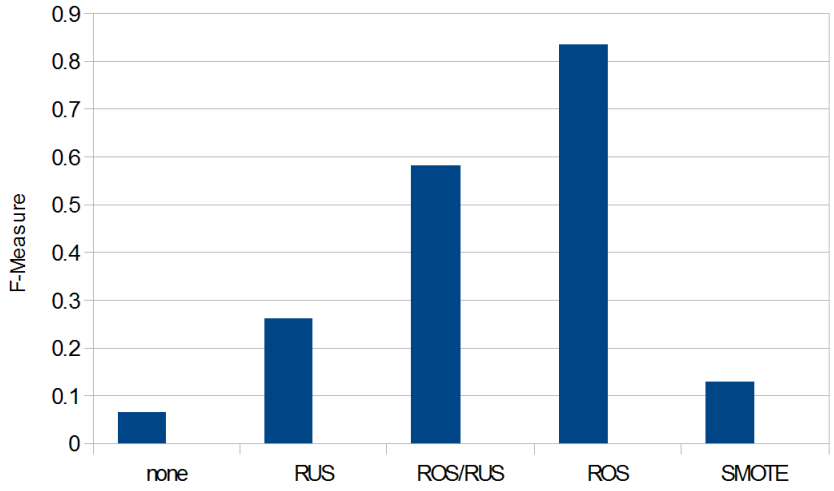

Figure 4.4: The effects of the different data methods on the C4.5 learner with the car fraud insurance dataset.

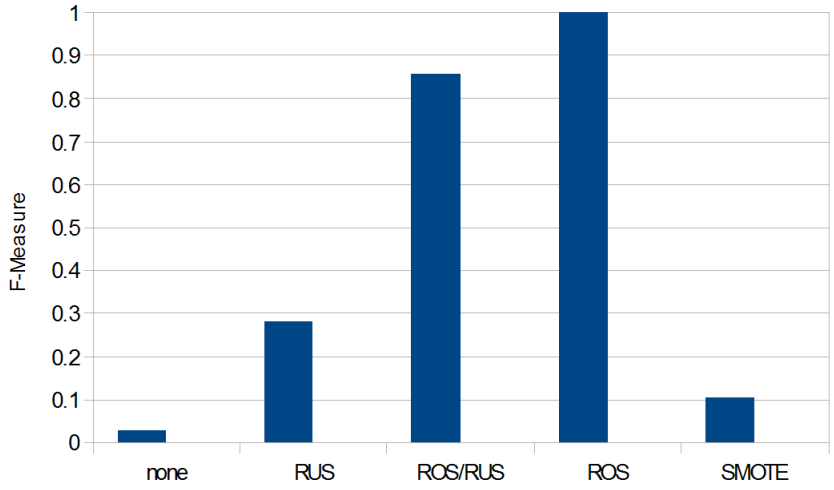

Figure 4.5: The effects of the different data methods on the random forest learner with the car fraud insurance dataset.

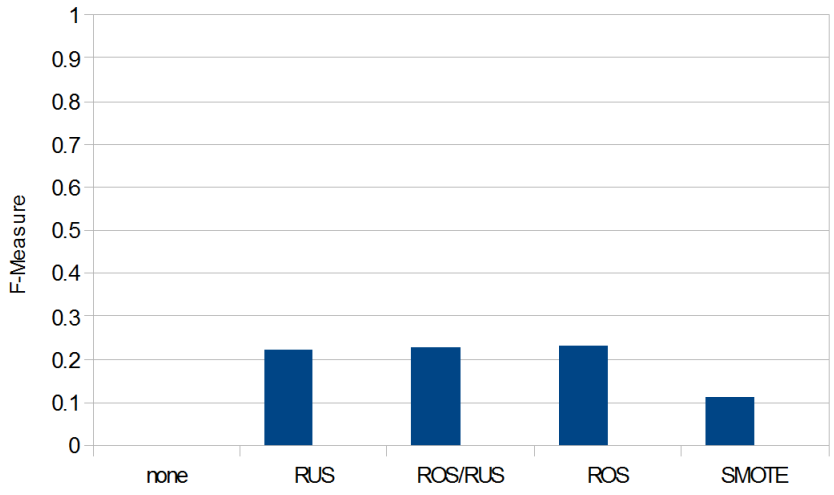

Figure 4.6: The effects of the different data methods on the decision tree learner with the car fraud insurance dataset.

From examination of table 4.1 and figures 4.1, 4.2, 4.3, 4.4 , 4.5, it is clear that not only did ROS deliver the best performance, the level of performance attained was far superior to that of other methods.

Figure 4.1 shows the effect of the different data methods for overcoming the class imbalance problem when used in conjunction with the K-NN learner. RUS and to a lesser extent ROS/RUS delivered poor performance. This can be attributed to both methods discarding too much of the examples of the majority class to allow the training of a classifier that can classify the majority class accurately. This results in them being misclassified as members of the minority class. This leads to the high number of false positives resulting in a drop in precision. This can be seen in table 4.1, where the precision of the RUS method is 0.1457 and ROS/RUS method is 0.6125. The results for the SMOTE data method should be disregarded as a K-NN learner was used to create the artificial replicants of the minority class.

Figure 4.2 delivered poor performance for all the data methods when used in conjunction with the Nave Bayes learner. However, with the exception of SMOTE, all the data methods resulted in relatively high values of recall but poor precision scores. The poor precision scores were due to the high numbers of false positives that the classifier predicted.

Figure 4.3 delivered almost perfect performance for the ROS method, with high precision and recall. The RUS method although delivering high recall, indicating it can accurately predict the majority class and they are not misclassified as minority members. However the classifier delivered poor precision. This was because it misclassified members of the majority class as members of the minority class. This leads to a high number of false positives and a resulting poor precision score. This leads to the low F-measure score. This problem is seen also with ROS/RUS method which delivers a relatively poor precision score. However it is size-ably more (RUS precision score 0.19, ROS/RUS precision score 0.6122). This increase in precision is because not as many of the samples of the majority sample are discarded. As less of the majority samples are discarded, the learner can train a model that is better at recognizing the majority class. The SMOTE algorithm proved totally incapable of generating accurate minority class samples. The application of the SMOTE algorithm caused an over generalization of the minority class, which resulted in the poor performance of the SMOTE data method. The SMOTE algorithm delivers poor precision and

poor recall scores indicative of a high number of false positive and false negatives.

Figure 4.4 showed that the ROS methods delivered the best level of performance, however the performance of the ROS data method for the C4.5 learner was significantly less than the random forest, ID3 or K-NN learner. RUS and SMOTE delivered the poorest level of performance. Again the RUS method delivered high recall but low precision scores indicative of misclassification of majority samples as minority samples. This was also seen for the ROS/RUS methods but not to the same extent, as not as many of the majority samples were discarded (thus resulting in the learner being able to train a model that is more accurate at correctly classifying the majority class). Like all the learners surveyed with the exception of the K-NN learner, the SMOTE data method delivered poor performance. The SMOTE algorithm delivered high recall but poor precision, indicative of a low number of false negatives but a high number of false positives. The high number of false positives are due to incorrect classification of the majority class as the minority class. This is due to the over generalization of the minority class. This results in the synthetic minority samples being too similar to those of the majority class and the learner is not able to evaluate correctly members of the majority class correctly and they are misclassified as members of the minority class. This results in large number of false positives, leading to poor precision and the resultant low F-measures.

Figure 4.5 shows the effect of the different data methods on the random forest learner. The perfect performance of the ROS data method is immediately clear. This is indicative of an over fitted model. The ROS/RUS methodology may be favoured instead as although the F-measure score is less, the model does not suffer from over fitting. The RUS methodology as was described previously discards too many of the majority samples and this results in a classifier which cannot classify accurately the majority class and instead predicts them to be minority class samples. This results in the poor performance of the model (due to a high number of false positives leading to poor precision and resulting in a low F-measure).

The results for the decision tree were plotted in Figure 4.6 and across all data methods, poor performance was achieved. This was similar to what was seen with the Nave Bayes learner. Further analysis of the data methods used in conjunction with the decision tree show that all the data methods showed an increase in recall but poor levels of precision. This is indicative of a large number of false positives i.e predicted minority class samples that are in fact majority class samples. This is not seen with the other tree learners, ID3 and WJ-48.

4.4 Overview of results of algorithmic-centric methods for dealing with class imbalance in the car insurance fraud dataset

Metaost (Domingos 1999) regards the appropriate classifier as a black box and therefore requires little information about how the classifier works. It can be used with varying a number of classes and its accompanying cost matrix. Metacost has been shown to be highly successful in terms of increasing predictive accuracy of the minority class. Metacost works in the following manner; a number of bootstrapped copies of the original dataset are created. The classifier is then learned on each of the new replicate datasets. Next each class's probability is approximated for each example in the dataset by calculating the fraction of votes it receives from the ensemble. Next, a conditional risk equation is utilized to rename all training examples with the estimated best class. Finally, the classifier is trained with the relabelled training set.

The Metacost learner gave some interesting results. These are summarized in table 4.3. It should be noted that figure 4.3 shows no results for ID3 or decision tree learners, this is due to them for no matter what combination of majority and minority cost of mis-classification were used, only 0 F-measure values were recorded (i.e either the precision or recall of the learner returned 0).

From tables 4.3 it is clear that a ratio similar to real life costs of misclassification gave the optimal results. To aid in the analysis of the results using the Metacost wrapper procedure, graphs of the perfor-

mance measure where plotted against the ratio of the majority to minority cost. In doing this it became easier to identify optimal Metacost misclassification cost ratios for the different learners. The 0 point on the x-axis is the application of no learner. All experiments were carried out using ten fold cross validation with the Metacost procedure wrapped around the learner. With each Metacost procedure taking ten bootstrap samples of the training data (ten had been previously empirically derived by (Domingos 1999) to be the ideal value). These plots are shown in figures:

1. Figure 4.7, detailing the effects of varying the majority/minority cost on the random forest learner. The ideal ratio of the majority to minority misclassification cost was found to be in the region of 0.1 to 0.2. Before and after these values performance is seen to deteriorate. The best F-measure score achieved with the random forest learner was 0.2451. This score is categorized by a relatively recall score of 0.7124 and a poor precision score of 0.148. The poor precision score can be attributed to a high number of false positives. This is due to a high number of incorrectly classified minority classes which are actually majority class samples (false positives).

2. Figure 4.8, for the C4.5 learner details the effects of varying the majority/minority cost. For the C4.5 learner the values of 0.1 and and 0.14 for the ratio of majority minority cost delivered optimal performance. The best F-measure score achieved was 0.2654 for a ratio of the majority to minority cost of mis-classification of three hundred to two thousand three hundred. This is characterized by an average recall score of (0.35) and an average precision score of (0.2137). This results in a relatively high number of both false positives and false negatives. By adjusting the ratio to 200/2300, the recall score of the learner is dramatically improved but the precision score is drastically lowered. This results in a very high recall score (0.9632) but a very poor precision score (0.1408).

3. Figure 4.9, detailing the effects of varying the majority/minority cost on the K-NN learner. K-NN was found to not be responsive to any change in the ratio of the majority minority misclassification cost, and a flat baseline was seen. For the Metacost procedure (Domingos 1999), it has noted that for learners that are prone to fluctuate in terms of changes in the dataset may not be suitable for use with Metacost. Furthermore, Li and Zhang (2011) have explained this as being due to fact that the K-NN learner carries out choices of classification by analysis at a relatively local level of dataset instances, where methodologies that introduce a change in cost of mis-classification may have little effect.

4. Figure 4.10, graphically shows the effects of varying the majority/minority cost on the Naive Bayes learner. A ratio of 200/2300 gave a large increase in performance for the recall score (0.7592) but a poor precision score (0.1429). This resulted in an F-measure of 0.2405. The behaviour is characteristic of the behaviour seen for the C4.5 and random forest learner. This is, low precision scores indicative of a high number of false positives. This is due to members of the majority class that are misclassified as members of the minority class.

For all the learners the results are summarized in table 4.3. Its quite clear that from analysis of this table and the figures cited previously, a ratio of 0.1 to 0.14 gives universally across all learners the best performance. It should be noted that this is close to the ratio in reality of the cost of misclassification of fraudulent transaction/misclassification of a non fraudulent transaction which is approximately 0.07.

Apart from the K-NN learner, all learners showed a large performance increase however the increase in performance when compared to that seen with the data methods is small. For all learners surveyed, only a F-measure of approximately 0.25 to 0.26 at best was achieved (this is only similar to the performance gains seen for the worst data methods). While the data methods showed sizeable performance increase across all learners, increase in F-measure were only noted in the case of Naive Bayes, random forest and C4.5 learner. A relatively poor improvement of the precision of the learner but a massive improvement in the recall of the learner was noted across the Naive Bayes, random forest and C4.5 learner. This was

due to the fact that while the learner had been dramatically improved in its ability to classify the minority class it also caused a very large increase in the misclassification of minority class samples that are in fact members of the majority class (false positives), resulting in only a small increase in terms of the precision of the learner. Because of the nature of fraud, this may not be a problem. This is due to the massive cost of not catching fraudulent cases compared to the relatively small cost of investigating a claim. False positives are non fraud cases mistakenly classified as fraud, while false negatives are fraud cases misclassified as not fraud. Therefore a model which minimizes the number of false negatives and returns a high recall would be favoured. This is due to the cost of investigating a non-fraudulent case as a fraud case is small, when compared to not catching a fraudulent case.

Attempts to counter the poor precision scores attained by finding optimal misclassification costs using genetic algorithms did not prove fruitful. For all learners surveyed, the processes were either found to poll out running out of memory or else could not come to a set of optimal values and simply ran indefinitely (2-3 days). Efforts to use simpler iterative approaches by applying ranges of misclassification costs (through the use of a grid operator) also proved unsuccessful, with the processes polling out due to lack of memory.

Table 4.3: The results of varying the metacost ratio for a number of different learners on the performance metrics precision, recall and F-measure. ID3 and the decision tree learner returned no measures.

Leaner	Majority Cost	Minority Cost	Ratio of Maj/Min Cost	Precision	Recall	F-Measure
K-NN	0	0	0.0000	0.106	0.1095	0.1077
K-NN	1	2300	0.0004	0.094	0.2514	0.1368
K-NN	10	2300	0.0043	0.094	0.2514	0.1368
K-NN	50	2300	0.0217	0.094	0.2514	0.1368
K-NN	100	2300	0.0435	0.094	0.2514	0.1368
K-NN	200	2300	0.0870	0.1077	0.1405	0.1219
K-NN	200	2300	0.0870	0.094	0.2514	0.1368
K-NN	400	2300	0.1739	0.0971	0.181	0.1264
K-NN	800	2300	0.3478	0.1023	0.1441	0.1197
K-NN	900	2300	0.3913	0.1038	0.1549	0.1243
K-NN	1000	2300	0.4348	0.1093	0.1246	0.1164
K-NN	1600	2300	0.6957	0.1188	0.1116	0.1151
K-NN	2000	2300	0.8696	0.1077	0.1405	0.1219
Nave Bayes	0	0	0.0000	0.2024	0.129	0.1575
Nave Bayes	1	2300	0.0004	0.0803	0.9632	0.1482
Nave Bayes	10	2300	0.0043	0.0853	0.9556	0.1566
Nave Bayes	50	2300	0.0217	0.1021	0.9491	0.1844
Nave Bayes	200	2300	0.0870	0.1429	0.7592	0.2405
Nave Bayes	200	2300	0.0870	0.1285	0.8202	0.2222
Nave Bayes	300	2300	0.1304	0.1285	0.8202	0.2222
Nave Bayes	400	2300	0.1739	0.1418	0.609	0.2300
Nave Bayes	800	2300	0.3478	0.1625	0.3619	0.2243
Nave Bayes	1200	2300	0.5217	0.1767	0.2437	0.2049
Nave Bayes	2000	2300	0.8696	0.2609	0.1405	0.1826
Nave Bayes	4000	2300	1.7391	0.3214	0.0602	0.1014
random forest	0	0	0.0000	0.2676	0.0141	0.0268
random forest	1	2300	0.0004	0.0681	0.9933	0.1275
random forest	10	2300	0.0043	0.0711	0.9866	0.1326
random forest	100	2300	0.0435	0.1117	0.9097	0.1990
random forest	100	2300	0.0435	0.1117	0.9097	0.1990
random forest	200	2300	0.0870	0.148	0.7124	0.2451
random forest	200	2300	0.0870	0.148	0.7124	0.2451
random forest	400	2300	0.1739	0.2045	0.3043	0.2446
random forest	400	2300	0.1739	0.2	0.3043	0.2414
random forest	600	2300	0.2609	0.1117	0.9097	0.1990
random forest	600	2300	0.2609	0.1117	0.9097	0.1990
random forest	1000	2300	0.4348	0.1117	0.9097	0.1990
random forest	2000	2300	0.8696	0	0	0.0000
C4.5	0	0	0.0000	0.9688	0.0336	0.0649
C4.5	1	2300	0.0004	0.0648	0.9989	0.1217
C4.5	10	2300	0.0043	0.0817	0.9989	0.1510
C4.5	100	2300	0.0435	0.1266	0.9588	0.2237
C4.5	200	2300	0.0870	0.1408	0.9632	0.2457
C4.5	200	2300	0.0870	0.1277	0.921	0.2243
C4.5	300	2300	0.1304	0.2137	0.35	0.2654
C4.5	800	2300	0.3478	0.3292	0.1214	0.1774

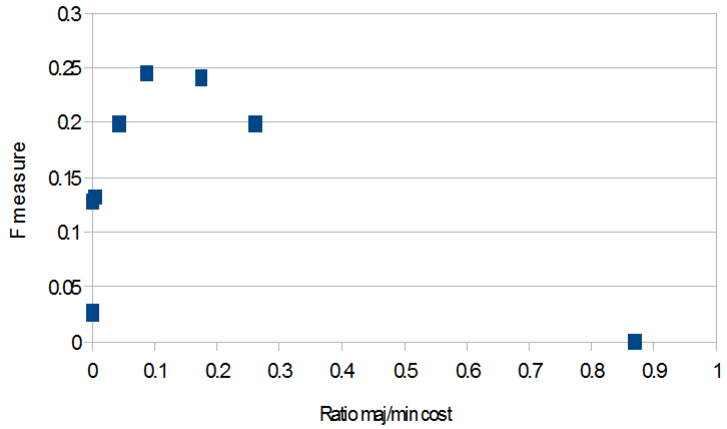

Figure 4.7: The effects of applying random forest with a Metacost wrapper class for dealing with the class imbalance problem.

4.5 Overview of results of data-centric and algorithmic methods for dealing with class imbalance in the car insurance fraud dataset

For the car insurance fraud dataset (Phua et al. 2004), the data methods proved to be superior to the algorithmic methods for all the learners tested. The data methods surveyed were found to be simple to implement and at least some of them were highly effective. The most effective of the data methods were found to be the ROS and ROS/RUS data method. ROS replicates the minority class until the number of minority and majority samples are equal in the training set. Oversampling of the minority class results in the creation of sufficient samples so as to allow the creation of a decision boundary around the minority class. However not all data methods proved as successful in overcoming the class imbalance problem. The RUS method, and to a less extent the ROS/RUS method, proved to result in some of the more useful majority examples being removed, causing the synthesis of a less than perfect model which is unable to classify a large number of the majority class samples. This results in poor precision of the model (Kotsiantis and Pintelas 2003). Potential method(s) to get around this problem is the use of focussed under sampling, selecting on an informed basis, members of the majority samples for the training set. An alternative to this is the use of techniques similar to bagging, where sampling of the dataset is carried out multiple times so as to induce multiple classifiers (Liu et al. 2009). A third approach would be to use a progressive RUS of the majority class. A progressive majority under sampling approach, would not use a sample of the majority class the same as the size of the minority class. Instead a number of experiments would be run to find the optimal majority sample size to minority sample size ratio. This approach is tried using the (Torgo 2011) consumer fraud dataset in the following chapter. SMOTE proved to be ineffectual at creating artificial replicants of the minority class. This was due to the artificial replicants it created based on the minority class being too similar to the majority class (Chumpol 2012). This is the opposite of what has been seen in the literature where SMOTE has proved to be a very useful method for dealing with the class imbalance problem (Li et al. 2011). What became apparent during the course of the experiments was that when using data methods it was necessary to train the model using the balanced

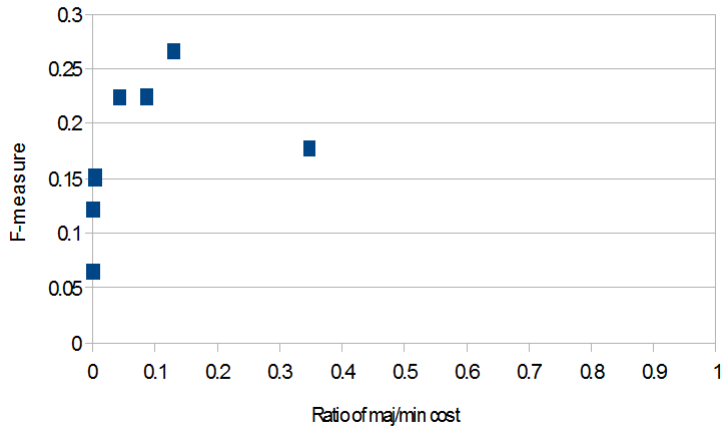

Figure 4.8: The effects of applying C4.5 (Weka J-48) with a Metacost wrapper class for dealing with the class imbalance problem.

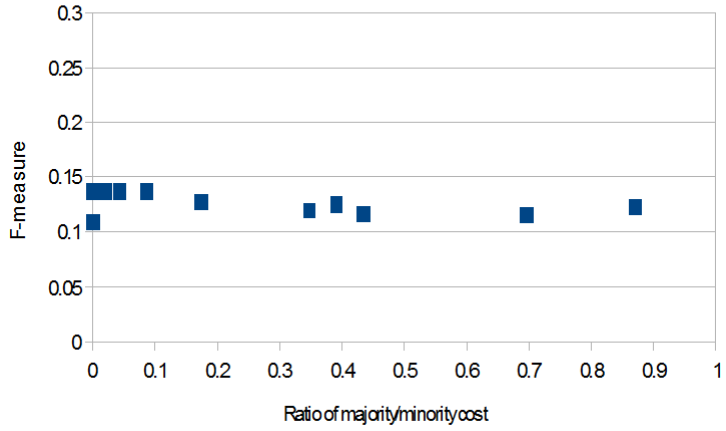

Figure 4.9: The effects of applying K-NN with a Metacost wrapper class for dealing with the class imbalance problem.

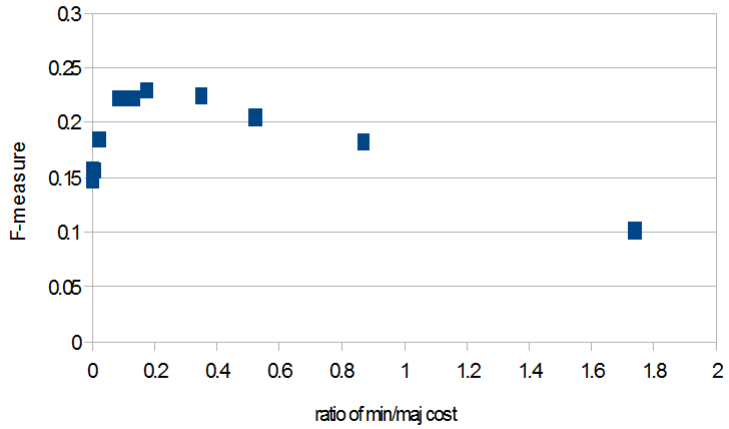

Figure 4.10: The effects of applying Naive Bayes with a Metacost wrapper class for dealing with the class imbalance problem.

dataset but to test the model using the original dataset so as to accurately gauge the performance of the model. Algorithmic methods were not as effective as data methods for dealing with the class imbalance problem. The Metacost procedure, as an approach to solve the class imbalance problem, did not live up to its prestigious reputation and proved to be incapable of improving the performance of all the learners surveyed substantially. In general, it improved the recall but had little or no effect on the precision of the learner. This resulted in learners which could differentiate the minority class from the majority class accurately, resulting in a low number of false negatives. They tended to be very poor at differentiating the majority from the minority class and misclassifying the majority class as minority class members (false positives). This led to poor precision of the model. It also proved more difficult to implement than the data methods. As for each learner, the metacost ratio of the majority to the minority class misclassification cost, had to be derived empirically. This takes considerable time and effort. Efforts to automate this optimization step did not prove fruitful, with both the use of genetic algorithms and more simple application of different metacost misclassification ratios through the use of a grid operator proving computationally prohibitive and often failing to execute. However in the case of fraud detection, high recall scores are more important than precision scores. As the cost of not catching a fraudulent claim has a high cost and will also tend to amplify the problem further and encourage more fraud to take place. While misclassifying non-fraudulent claims as fraudulent will pose less of a problem, as the cost of investigating a falsely classified fraudulent case is relatively low.

Chapter 5

The study of the class imbalance problem and the consumer fraud dataset

5.1 Describing the Torgo dataset

The Dataset provided by Torgo (2012 b) has five attributes these are:

1. Salesman id which can be described as nominal value.
2. Product id which is a nominal value.
3. Qaunt; quantity which is the number of units sold and the data type is integer.
4. Val; the total monetary value of the sale, the data type is integer.
5. Insp; the inspected label, the labels are ok or fraud.

5.2 Data preparation steps carried out with the Torgo dataset

This gave a dataset with 14462 OK labelled examples and 1270 fraud labelled examples, this gave a class imbalance of 0.0807 of minority samples to majority samples. There was no other data quality issues with the dataset. The data was provided as a R dataset with a number of unlabelled samples. The unlabelled data was removed, before exporting the labelled data to RapidMiner for experimentation. When applying the SMOTE algorithm to the data the SMOTE algorithm was applied in R and then exported from R and loaded into RapidMiner.

5.3 Implementing the data pre-processing methods for the Torgo dataset

Again the data methods used were:

1. ROS

2. RUS
3. ROS/RUS
4. SMOTE

The data methods used were:

1. Metacost.
2. Metacost thresholds.

The following learners were used; Nave Bayes, K-NN, ID-3, random forest (Weka), decision tree and C4.5 (Weka). Similar to the previous fraud dataset whenever a balanced dataset was used, the model was trained on the balanced dataset but tested for performance on the original dataset for reasons already stated (see the chapter 3 for the reasoning behind this).

The F-measure was again used to ascertain the effectiveness of the data pre-processing method. The F-measure is the harmonic mean of the precision and recall. Again the data methods proved to be the most effective means of dealing with the class imbalance problem. All data methods delivered superior performance when compared to the algorithmic methods.

5.4 The assessment of the performance of the data-centric methods in overcoming the class imbalance problem

As can be seen from the table 5.1, the performance of the methods involving over sampling of the minority class dramatically increases the performance of the learner. Also the SMOTE algorithm delivers large increases in performance. This was not the case for the previous dataset were the SMOTE algorithm was found to over-generalize the minority class. Also like the car insurance fraud dataset, investigated in the previous chapter, methods that utilized under sampling delivered poor performance. To overcome this, a progressive under sampling approach was taken and these results are analysed and explained later in the chapter in section 5.5.

Figures 5.1, 5.2, 5.3, 5.4, 5.5, 5.6 show the effect of the different data methods on the performance of the different learners.

Examination of these graphical representations show the universal increase in performance of the oversampling methods using real data (ROS). However in the case of the decision tree and the ID3 learners, SMOTE delivered superior performance. For the random forest learner, the performance delivered by ROS and SMOTE was practically the same. SMOTE delivered a F-measure of 0.9909 and ROS 0.9956. Analysis of the individual learners with the data methods surveyed is listed below:

1. Figure 5.1 shows graphically the performance of the different data methods with the C4.5 learner. SMOTE and ROS showed the largest increase in performance. ROS/RUS increased performance substantially, however RUS showed a relatively poor increase in performance. While ROS (F-measure 0.9039) did deliver the best performance, the performance of the SMOTE algorithm (F-measure 0.8062) was relatively close to it. Both the precision score and recall score were found to be lower for the SMOTE algorithm than ROS. This would indicate both a higher number of majority class members predicted to be minority class members(false positive) and minority class members predicted to be majority class members(false negatives). In the case of this dataset, as in the previous dataset, the minority class is the fraud class. For fraud detection not catching fraud is far more expensive than the misclassification of 'ok' examples as fraud. This is due to relatively low cost of an investigation compared to the cost of monetary reward gained by fraudulent activity. For this reason the ROS/RUS method although achieving a lower score in terms of F-measure

(0.7426 as opposed to 0.8062), as it has a far higher recall (less false positives) than the SMOTE methodology it may result in the superior model. Again the under sampling methodology delivered poor performance compared to the other methods. This had been seen with the previous dataset. The reasoning behind this is that the sampling method discards too many of the majority class for it to accurately train a model to classify the majority class.

2. Figure 5.2 shows clearly that SMOTE was the best performing data method for the decision tree learner. ROS and ROS/RUS showed the same performance increases, while RUS delivered no poor performance gains. SMOTE delivered the highest level of performance gain and achieved a score very similar to that achieved for the C4.5 decision tree. Again over sampling methods proved highly capable at overcoming the class imbalance problem.

3. Figure 5.3 shows clearly that SMOTE was the most performant data method for the ID3 learner. This was also the case with the decision tree learner. Unlike in the previous study, the SMOTE algorithm did not result in over-generalization of the minority samples. ROS and ROS/RUS data methods delivered similar performance gains, though ROS was superior. This was due to a much lower precision score achieved with the ROS/RUS methods due to a higher number of false positives. This was due to the increased misclassification of majority class samples (ok) as minority class samples (fraud). This is presumably due to not sufficient number of majority class samples being available to the learner when training the model as they are discarded in the under sampling of the majority class. Again RUS delivered poor performance categorized by poor precision (0.1905) but high recall scores (0.9967). This was due to a high number of false positives as actual 'ok' class samples were mis-categorized as 'fraud'.

4. Figure 5.4 shows that both the SMOTE and ROS are the most performant data measures. ROS/RUS and RUS also delivered large increases in performance but not to the same extent as the other methods. While RUS for all other methods generally delivered very poor performance this was not the case for the K-NN were under sampling of the majority class delivered a strong performance increase.

5. Figure 5.5 shows the performance of the different data methods with the Naive Bayes learner. What is immediately striking about the graph is the relative poor performance when compared to all other learners with all data methods. .All data method delivered only mediocre performance gains. The ROS/RUS method delivered the largest gains in performance. The ROS/RUS method also delivered a very high precision score. High precision scores are important in fraud detection, as high precision results in a low number of false positives. Low numbers of false positives are important in fraud detection, as false positives, the incorrect classification of fraud as non fraud activities are very costly to a company. While false negatives, the incorrect classification of a non-fraudulent activity as a fraudulent activity is less costly. As this only results in a an unnecessary investigation.

6. Figure 5.6 shows that both SMOTE and ROS data methods delivered the largest gains in performance for the random forest learner. While SMOTE and ROS delivered almost perfect performance, SMOTE would be favoured as through the use of artificial replicants, the very high performance levels seen are less likely to be due to overfitting a learner to recognize a small number of samples. RUS again delivered very poor performance gains.

What is clear form the above analysis is that the methods based on oversampling delivered the largest performance gains. While under sampling delivered the poorest performance gains, if any. ROS, increases the number of number of minority classes by randomly copying members of the minority class or adding artificial replicants in the case of SMOTE. ROS has been criticized for it leading to over-fitting (Chawla

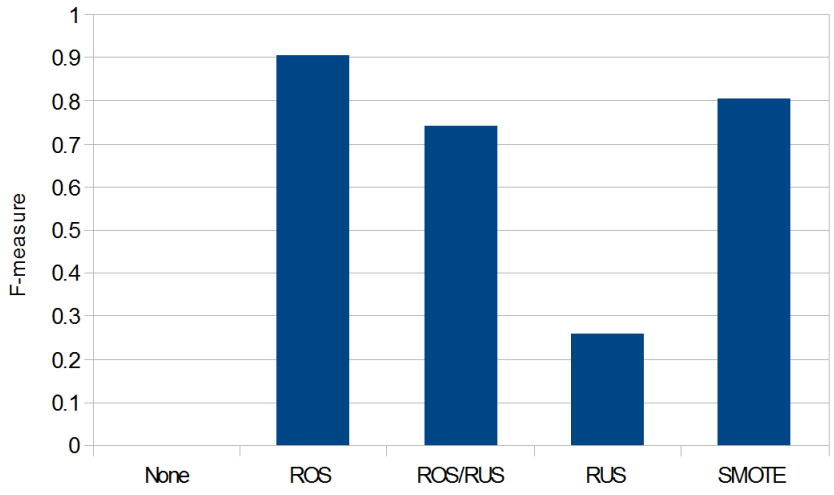

Figure 5.1: Plot of F-measure against data method for the C4.5 learner, when training using the balanced dataset and testing using the original dataset.

et al. 2002). This would therefore favour the use of the SMOTE algorithm, which as it does not use copies of the minority class, will not suffer from over-fitting. Under sampling methods have been found to deliver poor performance in previous studies (Kubat, Holte and Matwin 1998). This is due to the fact that a large number of the majority class are thrown away resulting in the learner not being able to classify the majority class accurately. This results in the model not being exposed to important members of the majority class which are discarded by the RUS method and which are important in training an effective classifier.

Table 5.1: The precision and recall and F-measures for all learners surveyed for the consumer fraud dataset, training on the balanced dataset, testing on the original dataset.

Learner	Pre-process	Precision	Recall	F-measure
C4.5	-	0	0	0.0000
C4.5	ROS	0.8296	0.9929	0.9039
C4.5	ROS/RUS	0.5989	0.9772	0.7426
C4.5	RUS	0.1511	0.9242	0.0000
C4.5	SMOTE	0.7589	0.8598	0.8062
decision tree	-	0	0	0.0000
decision tree	ROS	0.626	0.974	0.7622
decision tree	ROS/RUS	0.626	0.974	0.7622
decision tree	RUS	0	0	0.0000
decision tree	SMOTE	0.7839	0.9252	0.8487
ID3	-	0.5711	0.3535	0.4367
ID3	ROS	0.9921	0.937	0.9638
ID3	ROS/RUS	0.8246	0.9921	0.9006
ID3	RUS	0.1905	0.9967	0.7330
ID3	SMOTE	0.983	1	0.9914
K-NN	-	0.1429	0.024	0.0411
K-NN	ROS	0.9975	0.9917	0.9946
K-NN	ROS/RUS	0.5797	0.5515	0.5652
K-NN	RUS	0.9506	0.9992	0.9743
K-NN	SMOTE	0.9922	1	0.9961
Nave Bayes	-	0.1384	0.0378	0.0594
Nave Bayes	ROS	0.5562	0.4362	0.4889
Nave bayes	ROS/RUS	0.9007	0.4514	0.6014
Nave Bayes	RUS	0.2646	0.7409	0.3899
Nave Bayes	SMOTE	0.7016	0.3055	0.4257
random forest	-	0.5794	0.3378	0.4268
random forest	ROS	0.9914	1	0.9957
random forest	ROS/RUS	0.9262	0.9967	0.9602
random forest	RUS	0.1649	0.9967	0.2830
random forest	SMOTE	0.9867	0.9953	0.9910

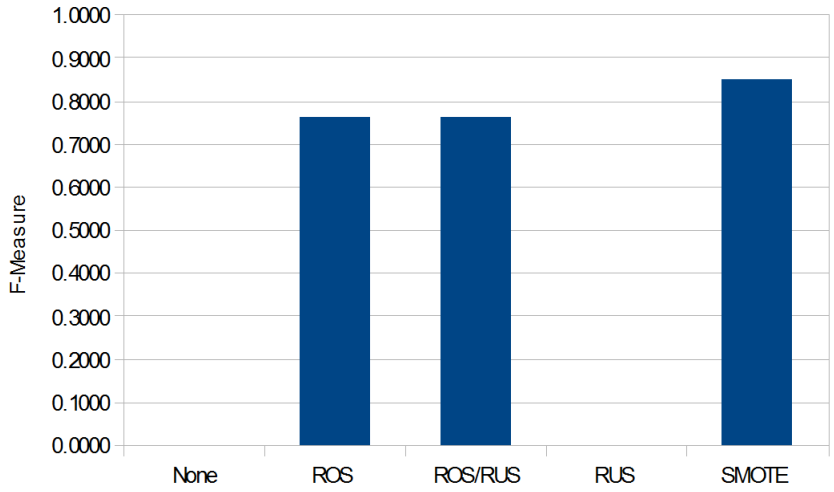

Figure 5.2: Plot of F-measure against data method for the decision tree learner, when training using the balanced dataset and testing using the original dataset.

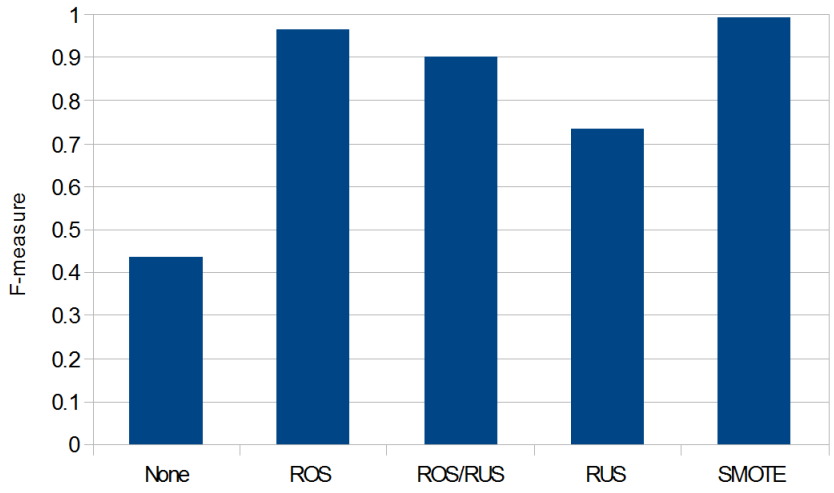

Figure 5.3: Plot of F-measure against data method for the ID3 learner, when training using the balanced dataset and testing using the original dataset.

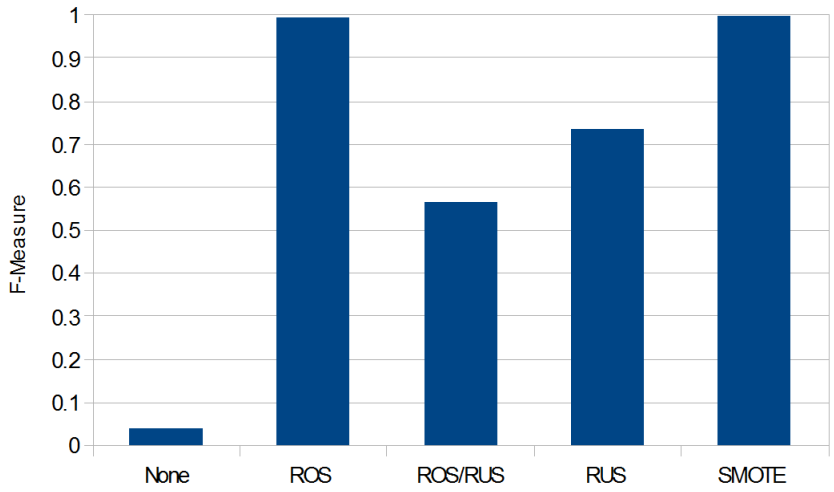

Figure 5.4: Plot of F-measure against data method for the K-NN learner, when training using the balanced dataset and testing using the original dataset.

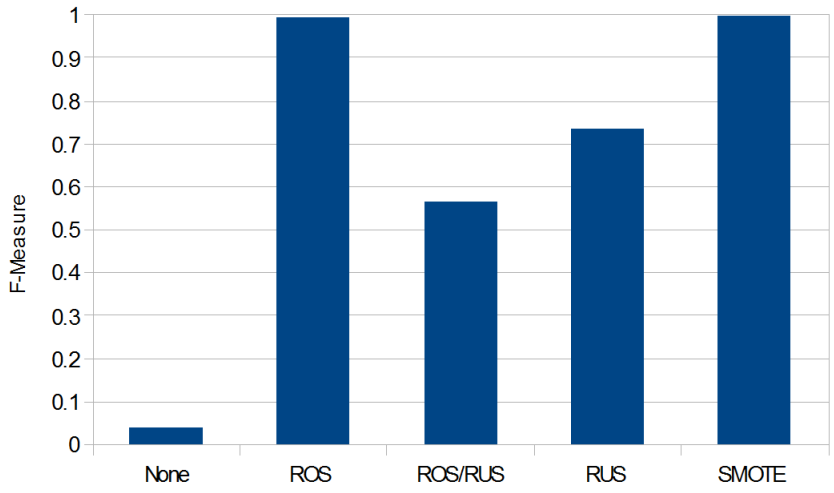

Figure 5.5: Plot of F-measure against data method for the Naive Bayes learner, when training using the balanced dataset and testing using the original dataset.

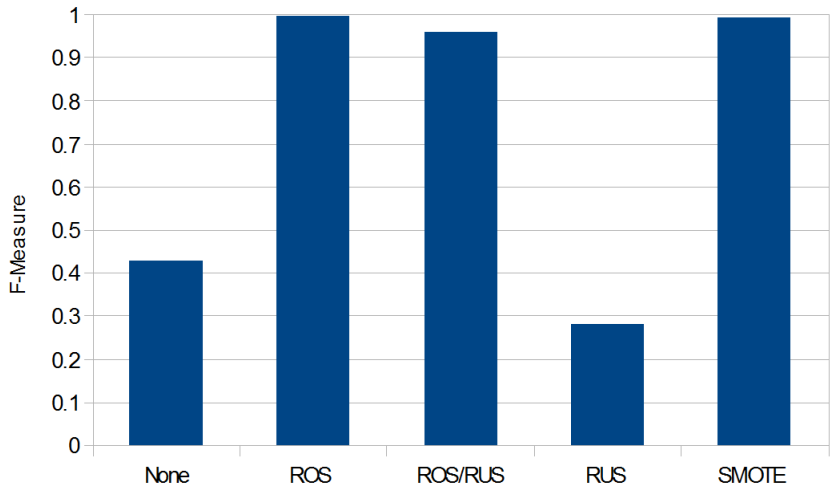

Figure 5.6: Plot of F-measure against data method for the random forest learner, when training using the balanced dataset and testing using the original dataset.

5.5 The use of progressive RUS strategy in overcoming the class imbalance problem in the consumer fraud dataset

The RUS method had been criticized because it discards too much of the majority sample (Chawla et al. 2002). Two learners were further experimented with, the ID3 and random forest learner. By increasing the the ratio of majority class to minority class, it was found to be able to increase the performance of a model. This because as more of the majority class is used to train the model and less of it is discarded, as is normal in under sampling of the majority class. The results are shown in table 5.2. What's clear from analysis of the table is that as the amount of the majority samples are increased the performance increases. When examining the precision and recall scores, one can see that the recall score stays the same but the precision increases. This is due to a drop in the number of false positives, majority class members (ok) being mistakenly classified as members of the minority class (fraud). The reason for the poor precision has been reported to be to do with the learner not being exposed to a high enough number of samples of the majority class to accurately classify them. By optimizing the amount of under sampling one can significantly improve the performance of the classifier. This is done by avoiding the discarding of potentially useful samples through a progressive under sampling approach being utilized. Various ratios of the number of majority samples to the number of samples of the minority samples were examined to find the one that delivered the highest performance levels (F-measure, precision, recall). The ideal ratio for both learners was found to be 7:1 (majority class/minority class).

The results shown in tabular form in table 5.2 are also shown in graphic format in figure 5.7.

Table 5.2: The precision and recall and F-measures for the random forest and ID3 learners surveyed for the Torga dataset using progressive under sampling.

Learner	Ratio of maj class/ratio of min class	Precision	Recall	F-measure
ID3	1	0.1905	0.9967	0.3198
ID3	2	0.539	0.9961	0.6994
ID3	3	0.6306	0.9906	0.7706
ID3	4	0.6998	0.9913	0.8204
ID3	5	0.7719	0.9913	0.8679
ID3	6	0.8165	0.9913	0.8954
ID3	7	0.8771	0.989	0.9296
ID3	11.387	0.5711	0.3535	0.4367
random forest	1	0.1649	0.9967	0.2830
random forest	2	0.5326	0.9961	0.6941
random forest	3	0.651	0.9843	0.7837
random forest	4	0.7092	0.9795	0.8227
random forest	5	0.8197	0.9843	0.8945
random forest	6	0.8638	0.9835	0.9198
random forest	7	0.8004	0.9756	0.8794
random forest	11.387	0.5794	0.3378	0.4268

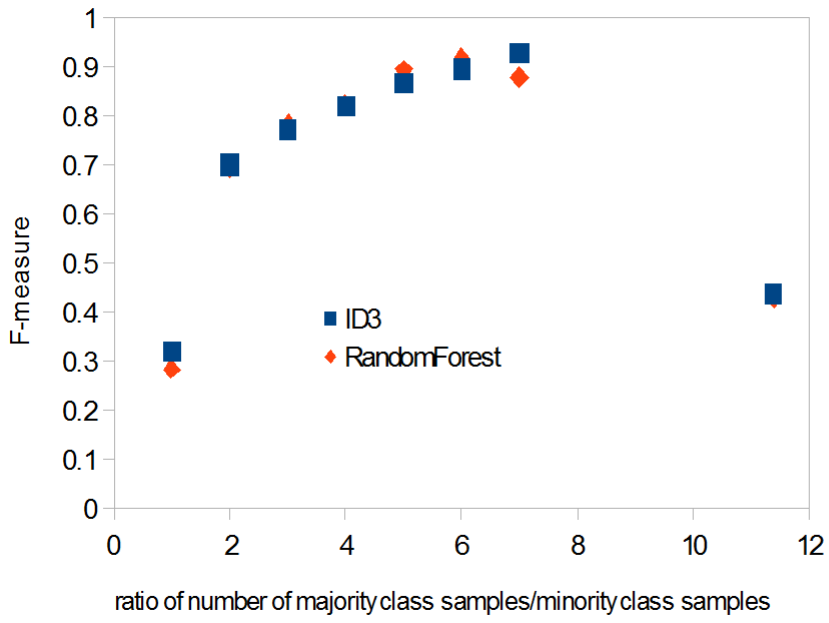

Figure 5.7: A plot of number of majority/ number of minority samples against F-measure for random forest and ID3 learner.

5.6 Overview of results of algorithmic methods for dealing with class imbalance in the consumer fraud dataset

Both the Metacost and metacost threshold methods were experimented with for The consumer fraud dataset.

5.6.1 The effects of the Metacost procedure on the consumer fraud dataset

As in the previous study utilizing the consumer fraud dataset

A summary of all results are displayed in table 5.4. Analysis of these results are listed below:

1. Figure 5.8 shows the plot of the F-measure against the ratio of the majority/minority cost for the Naive Bayes learner. For Nave Bayes, the optimal ratio was found to be 1:500 which gave a F-measure of 0.1961. This is the result of a very poor precision score (0.1323) and a mediocre recall score (0.3787). This is due to a high number of false positives and a high number of false negatives. This was a substantial increase in F-measure, from 0.05938 when no Metacost procedure is applied. The increase in performance was not as high as those achieved for the optimal data measures (ROS/RUS) with the Nave Bayes learner.

2. Figure 5.10 shows the plot of the F-measure against the ratio of the majority/minority cost for the K-NN learner. For the K-NN learner, ratios greater than 1:600 delivered optimal performance. Similar to the car insurance dataset, the F-measure score could not be improved substantially with the Metacost procedure. A F-measure of only 0.1138 could be achieved with a precision score of 0.0789 and recall score of 0.2039. The poor precision resulting in a very large number of false positives and the poor recall as a result of a high number of false negatives. The gain in performance in terms of F-measure was not as high as those seen for the optimal data measures (ROS and SMOTE).

3. Figure 5.9 shows the plot of the F-measure against the ratio of the majority/minority cost for the random forest learner. random forests when surveyed with all majority/minority cost combinations, no increase in performance was noted for all ratios. In fact only a drop in performance could be seen. This was in contrast to the data methods, were the ROS, ROS/RUS and SMOTE algorithm showed significant performance gains.

4. Figure 5.11 shows the plot of the F-measure against the ratio of the majority/minority cost for the ID3 learner. The ID3 learner was found not to deliver any increase in performance no matter what ratio was used. This was similar too what was seen previously with the random forest learner.

5. Figure 5.12 shows the plot of the F-measure against the ratio of the majority/minority cost for the C4.5 learner. The C4.5 learner when used with the Metacost procedure was found to deliver significant performance gains (from 0 to 0.3676). A ratio of 0.11 to 0.2 (1:7 and 1:5) delivered the optimal performance gains. These performance gains were not as great as those seen for the data methods.

6. The decision tree learner when used with the Metacost delivered zero performance, no matter what metacost ratio was used.

Analysis of the above results reveals some interesting trends. The Metacost procedure proved less capable than the data methods in its ability to increase the performance of the the learners, especially the ROS and SMOTE data methods. However it did prove capable of significant performance increases in the case of the C4.5 and the Nave Bayes learner.

Table 5.3: The effects of the Metacost procedure on the different learners.

Learner	Increase in performance	Deterioration in performance	No change
random forest		X	
ID3		X	
decision tree			X
K-NN			X
C4.5	X		
Nave Bayes	X		

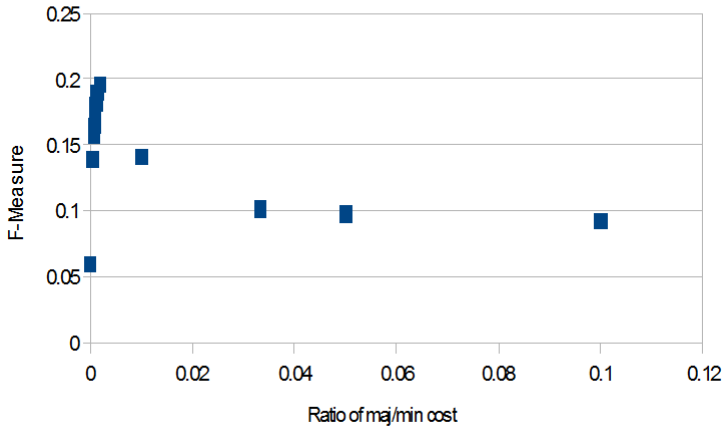

Figure 5.8: A plot of performance (F-measure) -versus- the ratio of majority cost/minority cost for the Naive Bayes learner (Metacost Procedure).

While in the case of the random forest and ID3, these learners only showed a deterioration of performance. The performance gains are summarized in the table 5.3.

So while Metacost could deliver performance gains for some of the learners (C4.5 and ID3), the complexities involved in deriving the optimal metacost ratio proved to consume a considerable amount of time. If used in a commercial environment the time needed to derive the optimal metacost ratio of the majority cost to the minimum cost, may prove prohibitive.

Table 5.4: Table showing varying majority,minority costs for the Metacost procedure and the effect on the F-measure of a number of different learners (ID3, decision tree, and C4.5) for the consumer fraud dataset.

Learner	Pre-process	Precision	Recall	F-measure	maj cost	min cost	ratio
C4.5	None	0	0	0.0000	0	0	0.0000
C4.5	Metacost	0.5965	0.2677	0.3696	1	10	0.1000
C4.5	Metacost	0.0807	1	0.1493	1	100	0.0100
C4.5	Metacost	0.0807	1	0.1493	1	20	0.0500
C4.5	Metacost	0.0857	1	0.1579	1	15	0.0667
C4.5	Metacost	0.6469	0.2409	0.3511	1	5	0.2000
C4.5	Metacost	0.6286	0.2598	0.3677	1	7	0.1429
C4.5	Metacost	0.6264	0.2654	0.3728	1	9	0.1111
C4.5	Metacost	0.899	0.1402	0.2426			0.0000
decision tree	None	0	0	0.0000	0	0	0.0000
decision tree	Metacost	0.0807	1	0.1493	1	2300	0.0004
decision tree	Metacost	0.08007	1	0.1483	10	2300	0.0043
decision tree	Metacost	0.08007	1	0.1483	100	2300	0.0435
decision tree	Metacost	0.0807	0.9	0.1481	200	2300	0.0870
decision tree	Metacost	0	0	0.0000	300	2300	0.1304
decision tree	Metacost	0	0	0.0000	500	2300	0.2174
decision tree	Metacost	0.0807	1	0.1493	175	2300	0.0761
decision tree	Metacost	0	0	0.0000	1	5	0.2000
decision tree	Metacost	0	0	0.0000	1	10	0.1000
decision tree	Metacost	0	0	0.0000	1	2.5	0.4000
decision tree	Metacost	0.0807	1	0.1493	1	500	0.0020
decision tree	Metacost	0.0807	1	0.1493	1	250	0.0040
decision tree	Metacost	0.0807	1	0.1493	1	100	0.0100
ID3	None	0.5711	0.3535	0.4366	0	0	0.0000
ID3	Metacost	0.4094	0.3717	0.3896	1	10	0.1000
ID3	Metacost	0.316	0.3724	0.3419	1	50	0.0200
ID3	Metacost	0.2647	0.3835	0.3132	1	100	0.0100
ID3	Metacost	0.2144	0.3882	0.2762	1	500	0.0020
ID3	Metacost	0.193	0.3835	0.2568	1	1000	0.0010
ID3	Metacost	0.2038	0.3795	0.2652	1	2000	0.0005
ID3	Metacost	0.2044	0.3803	0.2659	1	3000	0.0003
ID3	Metacost	0.1951	0.3913	0.2604	1	6000	0.0002
ID3	Metacost	0.4291	0.3394	0.3790	1	2.5	0.4000
ID3	Metacost	0.4291	0.3362	0.3770	1	5	0.2000
ID3	Metacost	0.42	0.3417	0.3768	1	8	0.1250
ID3	Metacost	0.3969	0.3394	0.3659	1	20	0.0500
ID3	Metacost	0.8613	0.1858	0.3057	1	2.5	0.4000
ID3	Metacost	0.9282	0.1323	0.2316	1	1.5	0.6667

Table 5.5: Table showing varying majority/minority costs for the Metacost procedure and the effect on the F-measure of a number of different learners (KNN,Naive Bayes and random forest) for the consumer fraud dataset.

Learner	Pre-process	Precision	Recall	F-measure	maj cost	min cost	ratio
K-NN	None	0.1429	0.024	0.0411	0	0	0.0000
K-NN	Metacost	0.0789	0.2039	0.1138	1	900	0.0011
K-NN	Metacost	0.0789	0.2039	0.1138	1	1800	0.0006
K-NN	Metacost	0.0789	0.2039	0.1138	1	2700	0.0004
K-NN	Metacost	0.0789	0.2039	0.1138	1	600	0.0017
K-NN	Metacost	0.0789	0.2039	0.1138	1	300	0.0033
K-NN	Metacost	0.0789	0.2039	0.1138	1	100	0.0100
K-NN	Metacost	0.0789	0.2039	0.1138	1	10	0.1000
K-NN	Metacost	0.0996	0.0528	0.0690	1	5	0.2000
K-NN	Metacost	0.0996	0.0528	0.0690	1	7.5	0.1333
K-NN	Metacost	0.1145	0.0323	0.0504	1	2.5	0.4000
Nave Bayes	None	0.1384	0.0378	0.0594	0	0	0.0000
Nave Bayes	Metacost	0.1135	0.0772	0.0919	1	10	0.1000
Nave Bayes	Metacost	0.116	0.0843	0.0976	1	20	0.0500
Nave Bayes	Metacost	0.118	0.089	0.1015	1	30	0.0333
Nave Bayes	Metacost	0.1387	0.1441	0.1413	1	100	0.0100
Nave Bayes	Metacost	0.1323	0.3787	0.1961	1	500	0.0020
Nave Bayes	Metacost	0.1083	0.485	0.1771	1	1000	0.0010
Nave Bayes	Metacost	0.0922	0.5402	0.1575	1	1500	0.0007
Nave Bayes	Metacost	0.0794	0.5693	0.1394	1	2000	0.0005
Nave Bayes	Metacost	0.0986	0.5094	0.1652	1	1250	0.0008
Nave Bayes	Metacost	0.1206	0.4441	0.1897	1	750	0.0013
Nave Bayes	Metacost	0.1123	0.4709	0.1814	1	900	0.0011
random forest	None	0.5794	0.3378	0.4267	0	0	0.0000
random forest	Metacost	0.2943	0.3937	0.3368	1	10	0.1000
random forest	Metacost	0.2236	0.537	0.3157	1	100	0.0100
random forest	Metacost	0.2381	0.6079	0.3422	1	500	0.0020
random forest	Metacost	0.2386	0.6165	0.3440	1	1000	0.0010
random forest	Metacost	0.2407	0.6378	0.3495	1	1500	0.0007
random forest	Metacost	0.2407	0.6378	0.3495	1	2000	0.0005
random forest	Metacost	0.2407	0.6378	0.3495	1	2500	0.0004
random forest	Metacost	0.2407	0.6378	0.3495	1	2500	0.0004
random forest	Metacost	0.2407	0.6378	0.3495	1	6000	0.0002
random forest	Metacost	0.3694	0.3858	0.3774	1	5	0.2000
random forest	Metacost	0.3535	0.389	0.3704	1	6	0.1667
random forest	Metacost	0.3206	0.3913	0.3524	1	8	0.1250
random forest	Metacost	0.3957	0.3756	0.3854	1	4	0.2500
random forest	Metacost	0.4381	0.3677	0.3998	1	3	0.3333
random forest	Metacost	0.4938	0.3528	0.4116	1	2	0.5000
random forest	Metacost	0.5339	0.3307	0.4084	1	1.5	0.6667
random forest	Metacost	0.514	0.3433	0.4117	1	1.75	0.5714
random forest	Metacost	0.5644	0.3173	0.4062	1	1.25	0.8000
random forest	Metacost	0.6616	0.2425	0.3549	1	0.75	1.3333
random forest	Metacost	0.7567	0.1921	0.3064	1	0.5	2.0000

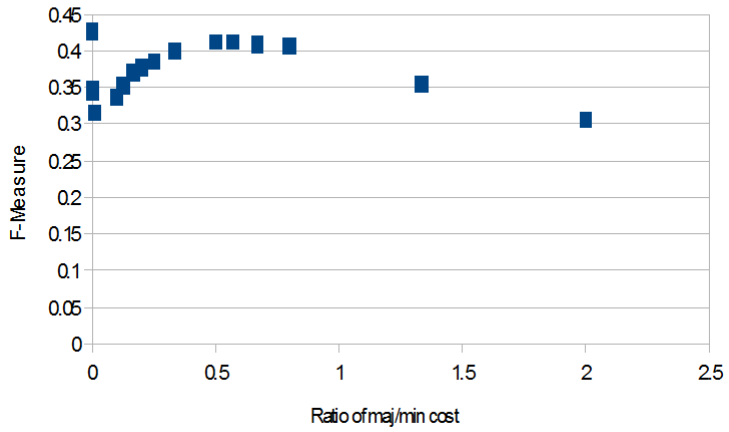

Figure 5.9: A plot of performance (F-measure) -versus- the ratio of majority cost/minority cost for the random forest (Weka) learner (Metacost Procedure).

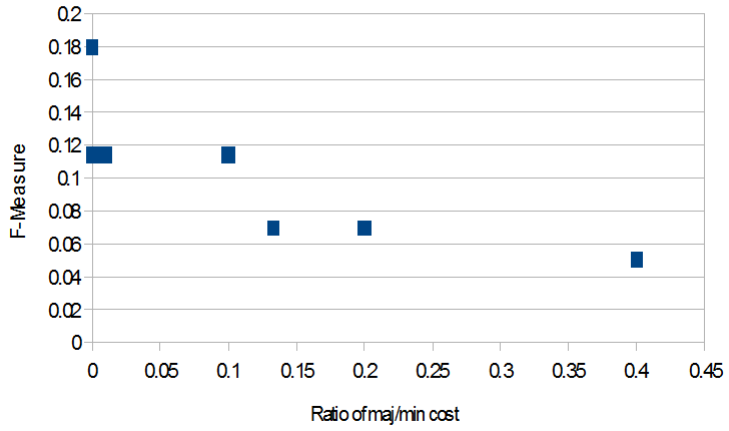

Figure 5.10: A plot of performance (F-measure) -versus- the ratio of majority cost/minority cost for the K-NN (Weka) learner (Metacost Procedure).

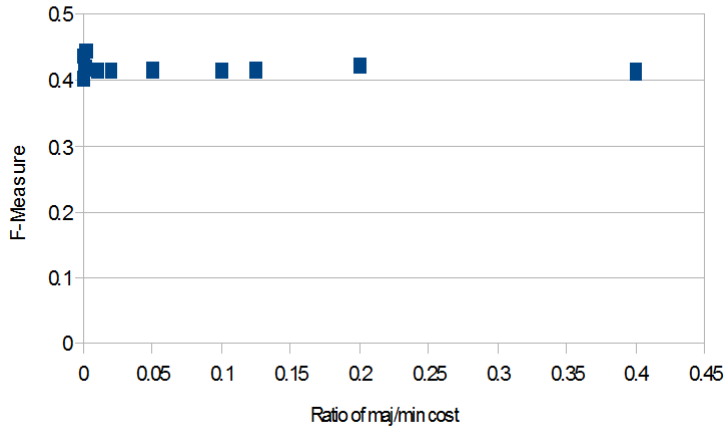

Figure 5.11: A plot of performance (F-measure) -versus- the ratio of majority cost/minority cost for the ID3 learner (Metacost Procedure).

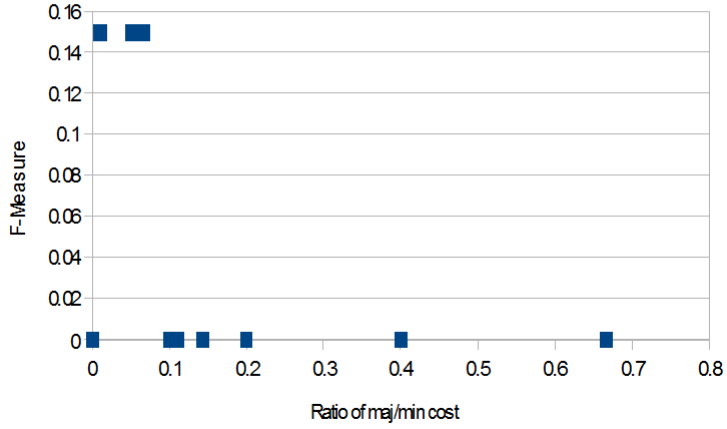

Figure 5.12: A plot of performance (F-measure) -versus- the ratio of majority cost/minority cost for the C4.5 (Weka C4.5) learner (Metacost Procedure).

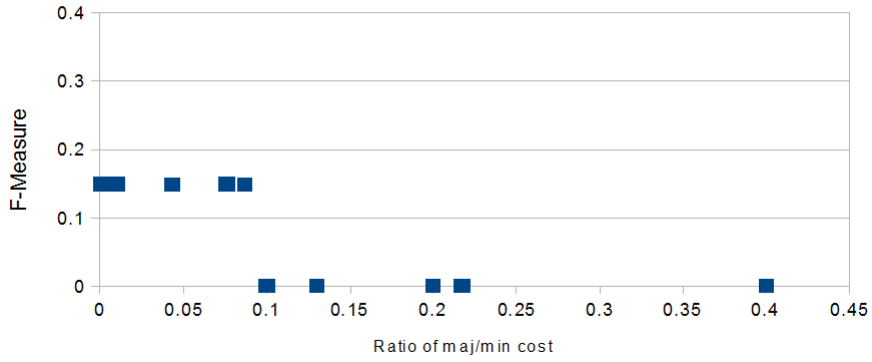

Figure 5.13: A plot of performance (F-measure) -versus- the ratio of majority cost/minority threshold for the decision tree learner (Metacost Procedure).

5.6.2 The effect of the metacost thresholds algorithmic method with the consumer fraud dataset

Metacost thresholds were also experimented with, again the ideal metacost thresholds had to be derived empirically. The results are summarized in table 5.7. These results were also represented graphically in figures 5.14, 5.15 and 5.16 .

1. Figure 5.14 shows the plot of the ratio of the majority thresholds against the minority threshold values and its effect on the performance measure for the Nave Bayes learner. The ideal ratio was found to be 1.5:1. This gave a F-measure of 0.1889 as opposed to 0.1961 achieved with the Metacost procedure. The performance gains were found to be similar to those obtained using the Metacost procedure and were not comparable to the performance gains achieved with the data methods.

2. Figure 5.15 shows the plot of the ratio of the majority thresholds against the minority threshold values and its effect on the performance measure for the random forest learner. Only very small gains were observed using metacost thresholds. The ideal ratio was found to be 2.5:1. This gave a F-measure of 0.4486, which showed a very minor increase in performance over using no metacost thresholds operator at all. While The metacost thresholds did prove superior to the Metacost procedure, only negligible increases in performance were seen. The performance gains delivered by metacost threshold were not comparable to those seen for the data methods.

3. Figure 5.16 shows the plot of the ratio of the majority thresholds against the minority threshold values and its effect on the performance measure for the C4.5 learner. No significant increases in performance were seen with the use of the metacost thresholds method. This was not the case with the Metacost procedure were significant increases were seen. The highest F-measure score seen was 0.1493. The highest F-measure gains achieved with the Metacost procedure was 0.3676.

4. Figure 5.17 shows the plot of the ratio of the majority thresholds against the minority threshold values and its effect on the performance measure for the ID3 learner. The ideal ratio was found to be 2.5:1. Metacosts thresholds were found to deliver somewhat better performance then the Metacost procedure. The Metacost procedure delivered a F-measure of 0.3896 as opposed to 0.4486 for

Table 5.6: The effects of the Metacost procedure and metacost thresholds on the performance of the different learners.

Learner	None	Metacost	Metacost Thresholds
Nave Bayes	0.0594	0.1961	0.1889
random forest	0.4267	0.4116	0.4486
C4.5	0	0.3676	0.1493
ID3	0.4366	0.3896	0.4486
K-NN	0.0411	0.1138	0.1855
decision tree	0	0.1493	0.1493

metacost thresholds. Again the performance gains delivered using metacost thresholds were not found to be as great as those seen for the the data methods.

5. Both the decision tree and K-NN learners showed none or little increase in performance and were not plotted. For K-NN the Metacost procedure delivered a F-measure score of 0.1138, while the metacost thresholds method delivered a best F-measure of 0.1493. For the decision tree learner, the Metacost procedure delivered a best F-measure score of 0.1493 score, for metacost thresholds was 0.1493.

The results obtained with the different misclassification cost algorithmic methods are shown in table 5.6. While the metacost thresholds method could improve the performance of learners that could not be improved with the Metacost procedure, i.e the random forest and ID3 learner the level of performance gain attained over not applying any algorithmic method is negligible. Only the Metacost C4.5 combination showed any significant gains in performance.

Table 5.7: The application of metacost thresholds meta learners

Learner	Type	Pre-process	Precision	Recall	F-measure	maj threshold	min threshold	ratio
random forest	None	None	0.5794	0.3378	0.4267	0	0	0.0000
random forest	Algorithmic	metacost thresholds	0.0809	1	0.1497	1	10	0.1000
random forest	Algorithmic	metacost thresholds	0.1465	0.9094	0.2523	1	2	0.5000
random forest	Algorithmic	metacost thresholds	0.6238	0.3323	0.4336	5	1	5.0000
random forest	Algorithmic	metacost thresholds	0.439	0.452	0.4454	2	1	2.0000
random forest	Algorithmic	metacost thresholds	0.4916	0.4126	0.4486	3	1	3.0000
random forest	Algorithmic	metacost thresholds	0.4916	0.4126	0.4486	2.5	1	2.5000
ID3	None	None	0.5711	0.3535	0.4366	0	0	0.0000
ID3	Algorithmic	metacost thresholds	0.4916	0.4126	0.4486	2.5	1	2.5000
ID3	Algorithmic	metacost thresholds	0.0798	0.9835	0.1476	1	10	0.1000
ID3	Algorithmic	metacost thresholds	0.0798	0.9835	0.1476	1	2	0.5000
ID3	Algorithmic	metacost thresholds	0.5761	0.3472	0.4333	5	1	5.0000
ID3	Algorithmic	metacost thresholds	0	0	0.0000	100	1	100.0000
ID3	Algorithmic	metacost thresholds	0.5922	0.3213	0.4166	10	1	10.0000
Nave Bayes	None	None	0.1384	0.0378	0.0594	0	0	0.0000
Nave Bayes	Algorithmic	metacost thresholds	0.2464	0.152	0.1880	2	1	2.0000
Nave Bayes	Algorithmic	metacost thresholds	0.4815	0.0205	0.0393	5	1	5.0000
Nave Bayes	Algorithmic	Metacost Thresholds	0.081	100	0.1619	0.5	1	0.5000
Nave Bayes	Algorithmic	metacost thresholds	0.1666	0.2181	0.1889	1.5	1	1.5000
Nave Bayes	Algorithmic	metacost thresholds	0.3	0.1055	0.1561	2.2	1	2.2000
Nave Bayes	Algorithmic	Metacost Thresholds	0.2145	0.1732	0.1917	1.8	1	1.8000
K-NN	None	None	0.1429	0.024	0.0410	0	0	0.0000
K-NN	Algorithmic	metacost thresholds	0.1787	0.1929	0.1855	1.8	1	1.8000
K-NN	Algorithmic	metacost thresholds	0.3333	0.0024	0.0048	2	1	2.0000
K-NN	Algorithmic	metacost thresholds	0.0807	1	0.1493	0.8	1	0.8000
K-NN	Algorithmic	metacost thresholds	0.333	0.0024	0.0048	1.5	1	1.5000
K-NN	Algorithmic	metacost thresholds	0.4	0.0016	0.0032	5	1	5.0000
C4.5	None	None	0	0	0.0000	0	0	0.0000
C4.5	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	1.5	0.6667
C4.5	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	2	0.5000
C4.5	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	2.5	0.4000
C4.5	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	3	0.3333
C4.5	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	10	0.1000
C4.5	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	50	0.0200
C4.5	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	100	0.0100
C4.5	Algorithmic	metacost thresholds	0.0807	1	0.1493	0	0	0.0000
decision tree	None	None	0	0	0.0000	0	0	0.0000
decision tree	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	5	0.2000
decision tree	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	2	0.5000
decision tree	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	7.5	0.1333
decision tree	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	30	0.0333
decision tree	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	1000	0.0010
decision tree	Algorithmic	metacost thresholds	0.0807	1	0.1493	1	5000	0.0002
decision tree	Algorithmic	metacost thresholds	0.0807	1	0.1493	1.8	1	1.8000

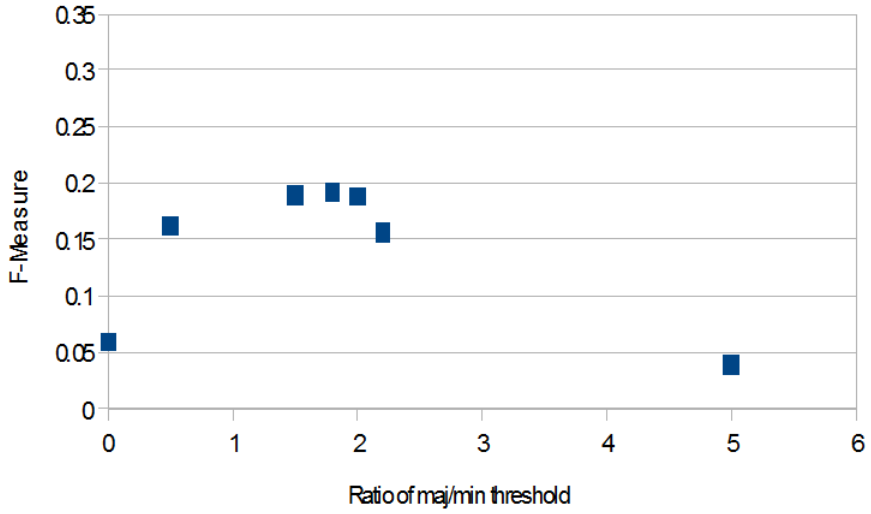

Figure 5.14: A plot of performance (F-measure) -versus- the ratio of majority threshold/minority threshold for the Naive Bayes learner (metacost thresholds).

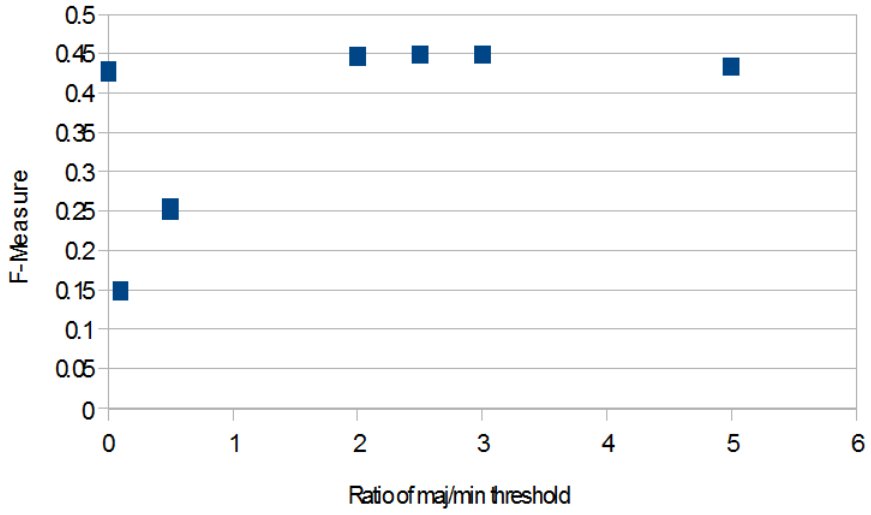

Figure 5.15: A plot of performance (F-measure) -versus- the ratio of majority threshold/minority threshold for the random forest learner (metacost thresholds).

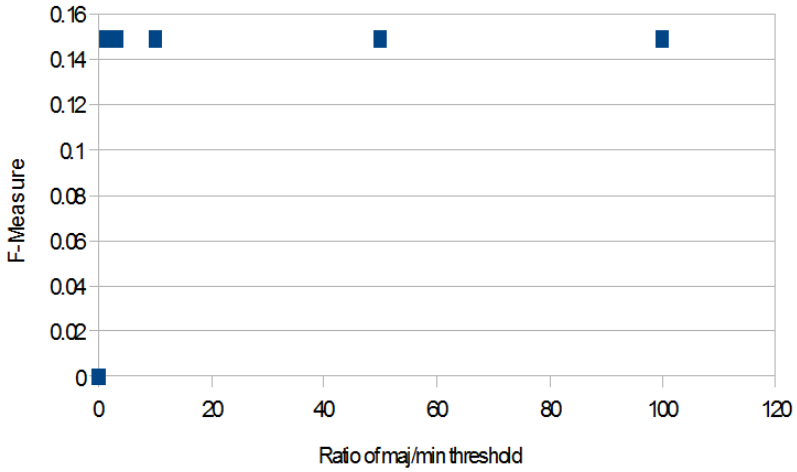

Figure 5.16: A plot of performance (F-measure) -versus- the ratio of majority threshold/minority threshold for the C4.5 learner (metacost thresholds)

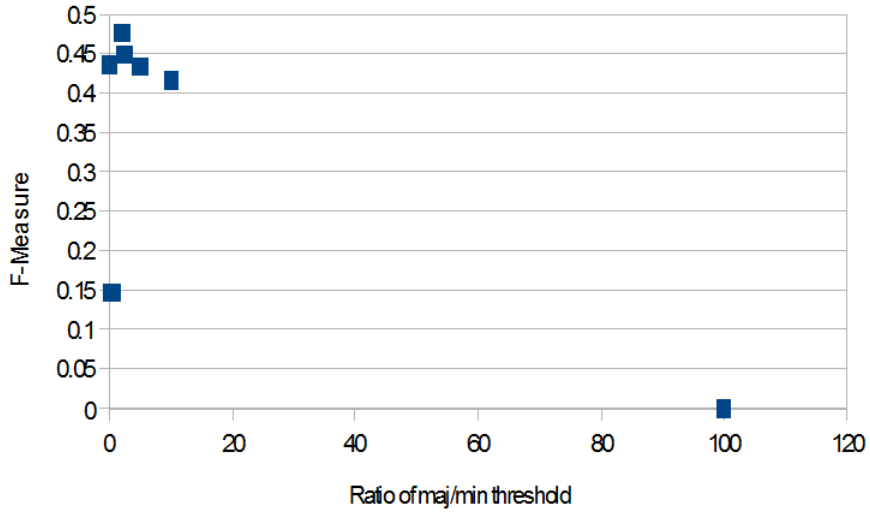

Figure 5.17: A plot of performance (F-measure) -versus- the ratio of majority threshold/minority threshold for the ID3 learner (metacost thresholds).

Table 5.8: The application of the RIPPER algorithm with various methodologies.

Precision	Recall	F-measure	Method
0.6961	0.6039	0.6467	100 rounds
0.7088	0.4677	0.5635	10 rounds
0.7811	0.5008	0.6103	Boosted RIPPER (10 rounds)
1	0.0142	0.0280	Boosted RIPPER info gain weight by (10 rounds)
0.8514	0.5331	0.6557	Boosted RIPPER Weight by value average (10 rounds)

5.7 The use of algorithmic methods to overcome the class imbalance problem in the consumer fraud dataset through the use of learners that are not susceptible to the class imbalance problem.

The previous algorithmic methods delivered relatively poor increases in performance. Only the C4.5 learner, with the application of the Metacost procedure showed a substantial performance increase. This prompted the exploration of alternative algorithmic methods, through the use of learners which were more resistant to the class imbalance problem. Two algorithms known for their lack of susceptibility to the class imbalance problem were evaluated, the RIPPER and the OneR learner.

The RIPPER (Repeated Incremental Pruning to Produce Error Reduction) algorithm is a rule induction based classifier. It uses a divide and conquer strategy to create a series of rules which describe a specific class. The RIPPER algorithm can be described as follows:

1. The training data is partitioned into two groups, one is used for creating the rules and the other is used for pruning.
2. A rule is developed through additions of further conditions. The instances that satisfy the rule are then taken away.
3. The rule developed is then optimized through the discarding of conditions that are over complex but do not deliver much classification power so as to make the rule simpler.
4. Go to step 2.
5. Further performance improvements through iteration are carried out.

The end product is a rule set which describes the dataset. In doing so it builds a series of rules for each class, including the rarest of classes. Raskutti and Kowalczyk (2004) have shown its particular usefulness especially with highly skewed noisy datasets containing many dimensions. The results for the use of the RIPPER learner along with a number of variations on its application are shown in table 5.8. In particle physics analysis, Britsch et al. (2008) have shown how the performance of the RIPPER learner can be further improved through boosting and weighting the examples set. This has also been found for the consumer fraud dataset, were application of the pre-processing methods of weighting the example set by weight by value average and a boosted RIPPER learner resulted in a further increase in performance. This is shown in table 5.8, where the methodology which delivers the optimal performance is a boosted RIPPER algorithm where the dataset is weighted by the value average.

The use of RIPPER prompted the consideration of other simple rule induction classifiers. The OneR algorithm was also evaluated. OneR (Holte 1993) is one of the most straight forward classification algorithms. The OneR learner results in the creation of a simple one-level decision tree. It is capable of constructing basic yet highly precise decision tree. In the study of student drop out predictions (Dekker

Table 5.9: The Application of the RIPPER and Ridor algorithm with Stacking and voting methodologies.

Preprocessing	learner	Precision	Recall	F-Measure
None	OneR	0.5599	0.2024	0.2973
Stacking C One R, RIPPER, random forest	0.7589	0.5346	0.6273	
Vote,OneR,RIPPER,Randomforest	OneR	0.7156	0.2496	0.3701
Vote random forest RIPPER ID3	-	0.7308	0.3446	0.4684
Vote random forestRIPPER ID3 OneR	OneR	0.7874	0.2591	0.3899
None	Ridor	0.6495	0.6192	0.6340
StackingC Ridor, RIPPER	Ridor	0.7615	0.513	0.6130
StackingC Ridor, RIPPERRandomForest	Ridor	0.7692	0.5699	0.6547
Vote Ridor, RIPPER, RandomForest	Ridor	0.8542	0.4249	0.5675

et al. 2009) , OneR's performance was not only well suited to mining class imbalanced data, its level of performance was similar to that of other 'state of the art' classifiers.

The OneR algorithm is described thus. OneR generates a rule for all attribute in the dataset, it then chooses a rule for each attribute in the dataset. This is its OneR rule based on which rule has the smallest error rate. To create a rule for an attribute, the most frequent class for each attribute is determined, this is the class that is the most common for that attribute. The error rate is calculated as the number of training data examples were the predicted label for the attribute rule does not agree with the actual class. The results for the OneR learner are shown in the table below, 5.9. The OneR learner delivered poorer performance than the RIPPER algorithm. The Ridor algorithm or (RIpple DOwn Rule learner) (Gaines and Compton 1995) was also evaluated. Ridor works by creating a default rule and then creates a number of cases that do not conform to this rule. The driving force behind creating these exceptions is that they lead to the lowest error rate. It creates further exceptions for each previous exception until it gives maximum performance. Variations of the RIPPER algorithm were also evaluated. Various ensemble methods were experimented with. These included Boosting and stacking. It was hoped that by combining the OneR and RIPPER classifiers predictions with those of other class imbalance resistant learners predictions, one could create a highly accurate classifier. However, through the combining of these classifiers through the use of stacking or voting, the performance of the RIPPER, Ridor or OneR algorithm could not be improved. Table 5.8 shows the effects of the different combinatorial methods used to combine the classifiers. Stacking or voting was not found to be able to increase the performance of these learners. Stacking (Dzeroski and Zenko 2004) involves the generation of multiple classifiers through the use of a number of learners and combining their predictions. The predictions from the different classifiers serve as an input to a meta-learner, which tries to join together the predictions from all the models resulting in a superior level of classification. StackingC (Ting and Witten 1999) has shown that multi value linear regression (MLR) when used as the meta learner delivers superior performance.

Table 5.10: The best in class and worst in class data methods for the consumer fraud dataset.

Learner	Best in class	Worst in Class
C4.5	ROS	RUS
decision tree	SMOTE	RUS
K-NN	SMOTE	ROS/RUS
ID3	SMOTE	RUS
nave Bayes	ROS/RUS	RUS
RandomForest	ROS/SMOTE	RUS

5.8 Overview of results of the data-centric and algorithmic methods for dealing with the class imbalance in theconsumer fraud dataset

The data methods proved to be the most capable class of methods for overcoming the class imbalance in the consumer fraud dataset. The SMOTE algorithm (Chawla 2000) proved to be the best in class data method when surveyed across all learners , see table 5.10.

SMOTE had not proved successful in treating the class imbalance in the previous study. It was found that the one sided approach to synthesizing the minority class which are not informed by the presence of the majority class, resulted in over-generalization of the synthetic minority examples produced by SMOTE. This had not been the case when the SMOTE algorithm had been used with the consumer fraud dataset, with the SMOTE algorithm proving capable of synthesizing minority classes which were not over-generalized. The samples created by the SMOTE algorithm proved to be accurate representations of the minority class. The method was found to increase the decision space of the minority class(by increasing the number of minority class samples) so as to allow the training of an accurate model in classifying members of the minority class. This resulted in the SMOTE algorithm being a highly effective method in dealing with the class imbalance problem.

ROS also proved to be highly capable data method in improving the performance of the learner(s) in training an effective model. One should recognize though that as oversampling is creating exact replicants of the minority class, over-fitting may occur (Kubat, Holte and Matwin 1998). However the methods involving the use of under sampling the majority class RUS and ROS/RUS proved less capable. Universally across all learners surveyed, with the exception of the K-NN learner, RUS proved to be the least effective method. This is a result of when using RUS, a large number of examples of the majority class are excluded when training the classifier. This can lead to important examples from the majority class being discarded from the training set. This results in the classifier not being able to effectively recognize members of the majority class (Chawla et al. 2002). For the ID3 and random forests learners, both proved to be resistant to the class imbalance in the dataset. The ability of the random forest learner to overcome the class imbalance in the data had been previously noted (Berkely 2012). For both ID-3 and random forest, the RUS data method was found to deteriorate the performance of the learner. The RIPPER learner (and other rule induction methods like RIDOR and OneR) also proved capable of dealing with the class imbalance problem in the data. The RIPPER algorithm has been shown previously to cope very well with class imbalanced data (Britsch et al. 2009). The performance of the RIPPER algorithm was also found to increase through the introduction of bagging. Other rule induction methods such as Ridor proved just as capable at performing well on imbalanced datasets. OneR did not prove as successful.

The data methods surveyed proved to be more capable then algorithmic methods in overcoming the class imbalance. Besides the Metacost procedure, metacost thresholds were tried but did not prove useful. While they did improve the performance of learners which the Metacost could not, they could only do so

very marginally. However while the algorithmic methods utilizing cost of misclassification did not prove useful, choice of learner(s) that are capable of dealing with skewed data proved to be useful (especially the use of rule induction based learners such as RIPPER and Ridor). They were not as capable as the most of the learner data methods combinations in overcoming the class imbalance problem. The use of the Metacost procedure and of metacost thresholds also have certain drawbacks. These drawbacks have been highlighted previously when experimenting with algorithmic methods on the car insurance fraud dataset. The main problem when using both the Metacost procedure and the metacost thresholds methods, the cost of misclassification must be empirically derived. This adds a certain complexity to the use of these methods. As in the case of the car insurance fraud dataset, processes to automate this task using genetic algorithms or through simple iteration (through the use of the optimize parameters operator in RapidMiner), over a range of values did not prove successful. As has been seen previously with the car insurance fraud dataset, data methods are superior to algorithmic methods. SMOTE proved to be the most useful data method in dealing with the class imbalance problem. Not only did it deliver the highest aggregate performance across all learners see (table 5.10), it is also less likely to lead to over-fitting as in the case of over sampling of the minority class (Wang et al. 2009). The most effective algorithmic method for overcoming the class imbalance problem was the use of the RIPPER and Ridor rule induction methods. This methodology (choice of learners which are not susceptible to the class imbalance problem) proved far superior to the Metacost procedure and use of metacost thresholds. These methods were also easier to implement. However attempts to combine classifiers that are resistant to the class imbalance problem to create a highly accurate classifier through the use of stacking or voting proved unsuccessful.

Chapter 6

The study of the class imbalance problem and the Thyroid disease dataset

6.1 Describing the Thyroid dataset

The Thyroid dataset(Quinlan 1986 a and b), downloaded from UCI (2012) was the last dataset to be experimented with. Due to the limited time available to run the experiments, only a limited number of experiments could be run. Therefore all data methods were evaluated. For the algorithmic methods, only the Metacost procedure and the use of learners that are not sensitive to the class imbalance problems were evaluated. The class distribution is shown below:

1. negative. (2723)
2. hyperthyroid. (62)
3. T3 toxic. (8)
4. goitre. (7)

Due to the very small number of all classes except the negative classes, when addressing the class imbalance problem special cases of oversampling was used. For the ROS and SMOTE experiments the T3 toxic and goitre where increased in number to the same number as the hyperthyroid class as opposed to the majority class. This oversampling methodology is somewhat different to the oversampling methodologies used previously. The reason for this was the absolute rarity of the T3 toxic and goitre class. When carrying out the RUS experiments, the majority sample was re-sampled to a size of two hundred, the approximately three times the size of the largest minority class (hyperthyroid).

The dataset is described in table 6.1.

Table 6.1: The thyroid dataset.

Label,attribute	Column Name	Type
label	thyrprob	polynominal
attribute	age:	numeric
attribute	sex:	binominal
attribute	on thyroxine:	binominal
attribute	query on thyroxine:	binominal
attribute	on antithyroid medication:	binominal
attribute	sick:	binominal
attribute	pregnant:	binominal
attribute	thyroid surgery:	binominal
attribute	I131 treatment:	binominal
attribute	query hypothyroid:	binominal
attribute	query hyperthyroid:	binominal
attribute	lithium	binominal
attribute	goitre	binominal
attribute	tumor	binominal
attribute	hypopituitary	binominal
attribute	psych	binominal
attribute	TSH measured	binominal
attribute	TSH	numeric
attribute	T3 measured	binominal
attribute	T3	numeric
attribute	TT4 measured	binominal
attribute	TT4	numeric
attribute	T4U measured	binominal
attribute	T4U	polynominal
attribute	FTI measured	binominal
attribute	FTI	numeric
attribute	TBG measured	binominal
attribute	referral source	polynominal

6.2 The pre-processing steps required for the Thyroid dataset

There were some problems with the dataset, the following attributes contained missing values:

1. Sex attribute: (110) missing value(s).
2. Age attribute: (1) missing value(s).
3. TSH attribute: (284) missing value(s).
4. T3 attribute: (585) missing value(s).
5. TT4 attribute: (184) missing value(s).
6. T4U attribute: (297) missing value(s).
7. FTI attribute: (295) missing value(s).

To overcome this problem, a RapidMiner operator was applied to the dataset, 'replace missing values' with the average value. This is referred to as 'mean imputation', where missing values are replaced by the means of the attribute. Although relatively simple it is one of the most frequently used methods for imputing missing data and also one of the most effective (Batista and Monard 2003).

Like the consumer fraud dataset and car fraud dataset before, all experiments involved using a balanced dataset for training. While testing the model was carried with the original dataset. The model was trained with the balanced dataset using 6 fold cross validation (6 fold cross validation was chosen so there would be at least one of the very rare classes, the goitre and T3 toxic class in each fold) and the performance was measured by applying the model trained to the original dataset and recording the performance. 6 fold cross validation was used so as to ensure there was at least one of the very rare minority classes in each fold. The workflow for this experimental set-up is shown in figure 6.1. Again recall, precision and F-measure are used as they are considered capable performance measures for both binary and multi-class learning (Rijsbergen 1979). So for RUS, ROS/RUS, ROS and SMOTE data experiments, the model was trained with the balanced dataset and tested with the original dataset, using 6 fold cross validation to train the model. For all experiments that did not require the use of a balanced dataset (i.e where no data method was used) 6 fold cross validation using the original dataset to train and test the model was used.

6.3 Overview of results of data-centric methods for dealing with class imbalance in Thyroid dataset

The results are shown in tabular form below in table 6.3. These results are shown graphically in figures 6.2, 6.3, 6.4, 6.5, 6.6, 6.7. The results are summarized below:

1. Figure 6.2 represents the effects of the different data methods on the performance of the C4.5 learner. SMOTE and ROS gave the best performance and achieved perfect performance. The SMOTE algorithm did deliver better performance outperforming the ROS method substantially for the hyperthyroid class and delivering approximately the same level of performance for the goitre and T3 Toxic class. However both the goitre and hyperthyroid class with no pre-processing delivered a very high level of performance (hyperthyroid=0.8065, T3 Toxic=0.5 and goitre 0.7692). In the previous investigations some learners had proved less susceptible to the class imbalance problem, most notably the random forest learner and simple rule induction learners like RIPPER or OneR. For the random forest learner this resistance to the class imbalance in data had been previously noted (Berkeley 2012).

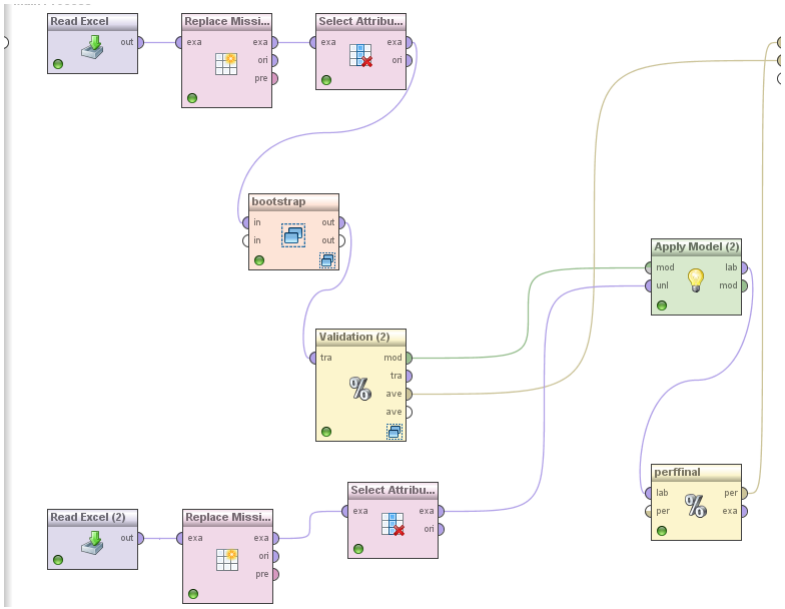

Figure 6.1: The Thyroid dataset data mining workflow.

2. Figure 6.3 represents the effects of the different data methods on the performance of the random forest learner. The SMOTE and ROS learner delivered the poorest level of performance change. Both ROS and SMOTE delivered a severe degradation in performance. While both RUS and ROS/RUS (for some classes) delivered better performance than the base learner. This phenomena is quite rare (amongst the datasets surveyed during the course of this investigation). Methods involving under sampling generally delivered poorer performance than those that utilize over sampling or the creation of artificial replicants. While in the case of the RUS method, under sampling delivered a massive increase in performance especially for the T3 toxic and goitre class.

3. Figure 6.4 represents the effects of the different data methods on the performance of the Nave Bayes learner. SMOTE and ROS again delivered substantial improvement in performance. However SMOTE did deliver superior performance to ROS. This has been attributed to increasing the decision space of the minority class to a sufficient size, so as to be able to train an efficient classifier. However RUS and ROS/RUS showed either no substantial improvement, or in some cases caused some degradation in performance. The poor performance of the RUS and ROS/RUS methods could be attributed to the removal of members of the majority class thereby discarding samples which maybe useful in training a model to classify the majority class. This reduces the ability of the learner to train a model that can recognize the majority class. What should be noted from figure 6.4 is that no matter what data method was utilized it was impossible to improve the performance for the hyperthyroid class.

4. Figure 6.5 represents the effects of the different data methods on the performance of the ID3 learner. Again SMOTE and ROS delivered the largest increase in performance. This had also been noted for other decision tree learners C4.5 (Weka) and RapidMiners decision tree. Both ROS and SMOTE delivered near perfect performance. The other tree learners experimented with also delivered an excellent level of performance. RUS and ROS/RUS showed improvement for the hyperthyroid class, however a degradation in the ability to recognize the T3 toxic and goitre class was noted.

Table 6.2: Results showing best in class and worst in class data methods for hyperthyroid, goitre and T3 Toxic classes.

Learner	Best in class	Worst in class
ID3	SMOTE	ROS/RUS, RUS
decision Tree	ROS, SMOTE*	ROS/RUS, RUS
K-NN	ROS, SMOTE	ROS/RUS
Nave Bayes	SMOTE	RUS
C4.5	SMOTE	RUS
random forest	RUS	SMOTE

5. Figure 6.6 represents the effects of the different data methods on the performance of the decision tree learner. Again SMOTE and ROS gave the largest increase in performance. ROS delivered the largest performance gains. However it should be noted that the SMOTE algorithm also delivered sizeable increases in performance, though not quite as high as those delivered by the ROS method. Methods based on under sampling of the majority class both RUS and ROS/RUS delivered a degradation of performance when compared to the base learner.

6. Figure 6.7 represents the effects of the different data methods on the performance of the K-NN learner. Again ROS and SMOTE gave the greatest gains in performance. Both RUS and ROS/RUS delivered either a slight degradation in performance or a slight increase in performance. Again this can be considered to be an affect of reducing the number of examples from the majority class. The discarding of potentially important majority class members deteriorates the performance of the classifier in successfully classifying members of the minority class from the majority class. However it should be noted that RUS did boost the performance of the model to recognize the T3 Toxic class.

In summary, SMOTE and ROS would appear to deliver the best performance across all the learners. For the C4.5 learner, both ROS,ROS/RUS and SMOTE delivered a very large increases in performance. K-NN also delivered exceptional performance across all classes for the ROS and SMOTE learner.

For all other learners, RUS and ROS/RUS delivered either no improvement in performance or a degradation in performance. This can be explained in terms of removal of a large number of majority examples limits the learner in recognizing the majority class. For SMOTE and ROS the addition of either artificial or replicants of the minority class drastically increased performance of the all learners to train a model. The contingency table 6.2 denotes the best in class and worst in class methods. As can be seen, SMOTE proved by far to be the most effective method for addressing the class imbalance in the data.

When rebalancing the dataset, there are two problems to overcome:

1. What (minority) classes need to be rebalanced.

2. To what extent to rebalance the data. ie the level of rebalancing that is needed.

For the Thyroid dataset the goitre and T3 Toxic classes were particularly small in size. Therefore, when employing a oversampling approach or creating synthetic samples using SMOTE, both classes were increased to the size of the minority classes with the largest number of samples, the hyperthyroid class. This proved to be a very useful strategy with both ROS and SMOTE delivering an excellent level of performance. The data methods, although criticized as a method for overcoming the class imbalance problem (Zhou an Liu 2010), proved to be very capable at increasing the performance of all learners surveyed.

Table 6.3: Results showing F-measures for hyperthyroid, goitre and T3 Toxic classes.

Learner	Method	hyperthyroid	T3 Toxic	goitre
C4.5	NONE	0.8065	0.5	0.7692
C4.5	RUS	0.5865	0.1882	0.1972
C4.5	ROS/RUS	0.5755	0.1882	0.2059
C4.5	ROS	0.5714	1	1
C4.5	SMOTE	0.8721	0.9411	1
random forest	NONE	0.6748	0.2666	0.3333
random forest	RUS	0.7187	0.9612	0.9912
random forest	ROS/RUS	0.5061	0.4848	0.2333
random forest	ROS	0.3846	0	0
random forest	SMOTE	0	0	0
Nave Bayes	NONE	0.6825	0.2666	0.3333
Nave Bayes	RUS	0.2884	0.4571	0.1644
Nave Bayes	ROS/RUS	0.2825	0.3403	0.1905
Nave Bayes	ROS	0.6946	0.6956	0.4118
Nave Bayes	SMOTE	0.6864	0.8235	0.4999
ID3	NONE	0.7049	0.5333	0.7272
ID3	RUS	0.5767	0.2161	0.1972
ID3	ROS/RUS	0.5188	0.2104	0.1972
ID3	ROS	0.923	1	1
ID3	SMOTE	1	1	1
decision tree	NONE	0.6727	0.4	0.2222
decision tree	RUS	0.4841	0.097	0.1972
decision tree	ROS/RUS	0.4841	0.082	0.1972
decision tree	ROS	1	1	1
decision tree	SMOTE	0.7083	0.7142	0.875
K-NN	NONE	0.5826	0	0.2478
K-NN	RUS	0.4901	0.698	0.1037
K-NN	ROS/RUS	0.4901	0.0625	0.0875
K-NN	ROS	1	1	1
K-NN	SMOTE	1	1	1

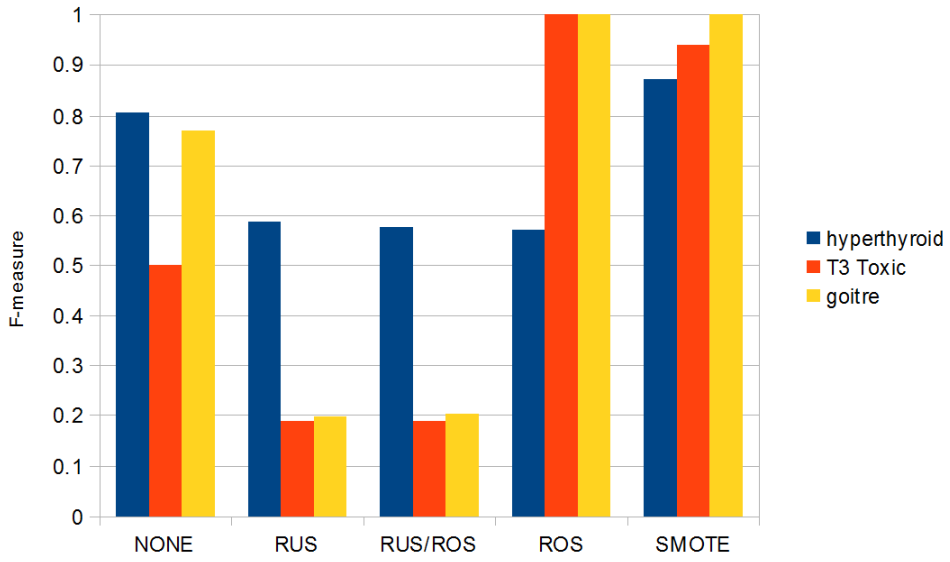

Figure 6.2: A plot of performance (F-measure) -versus- the re-sampling method for the C4.5 learner

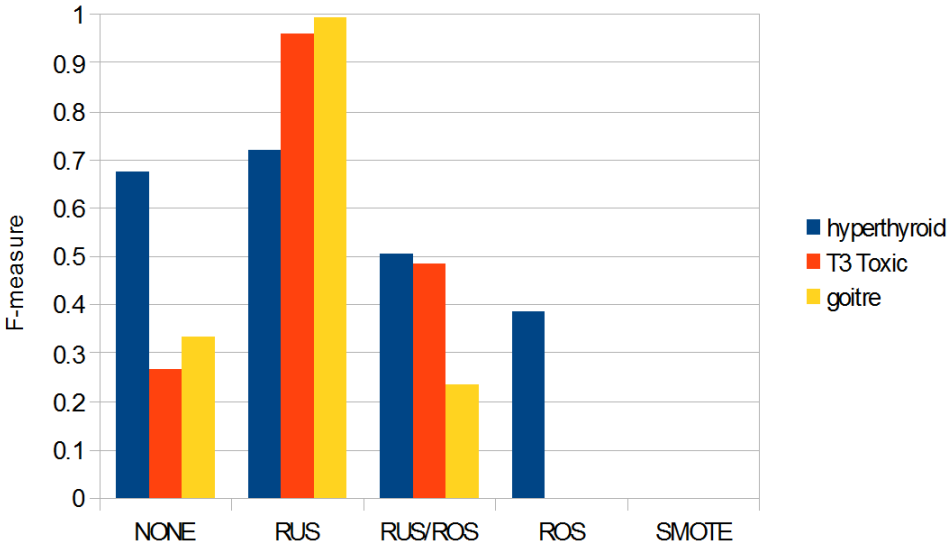

Figure 6.3: A plot of performance (F-measure) -versus- the re-sampling method for the random forest learner.

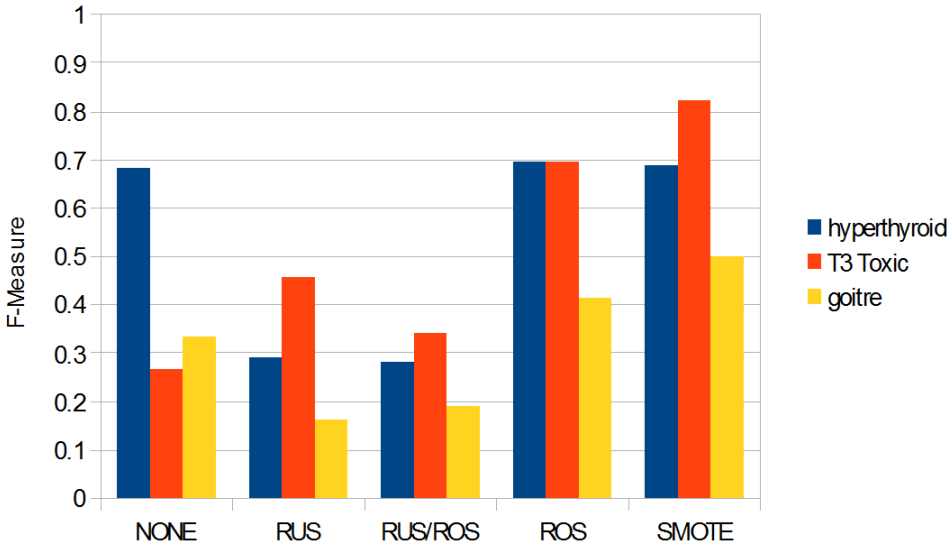

Figure 6.4: A plot of performance (F-measure) -versus- the re-sampling method for the Naive Bayes learner.

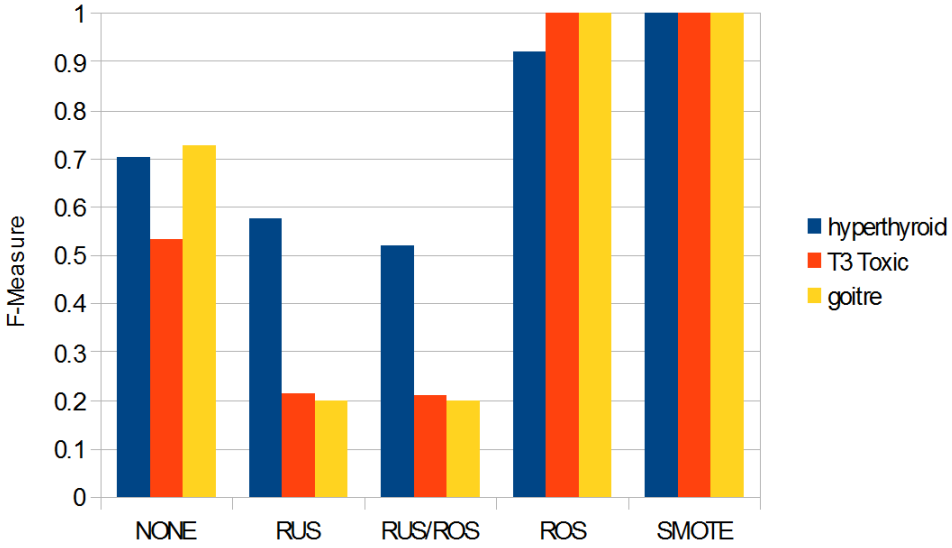

Figure 6.5: A plot of performance (F-measure) -versus- the re-sampling method for the ID3 learner.

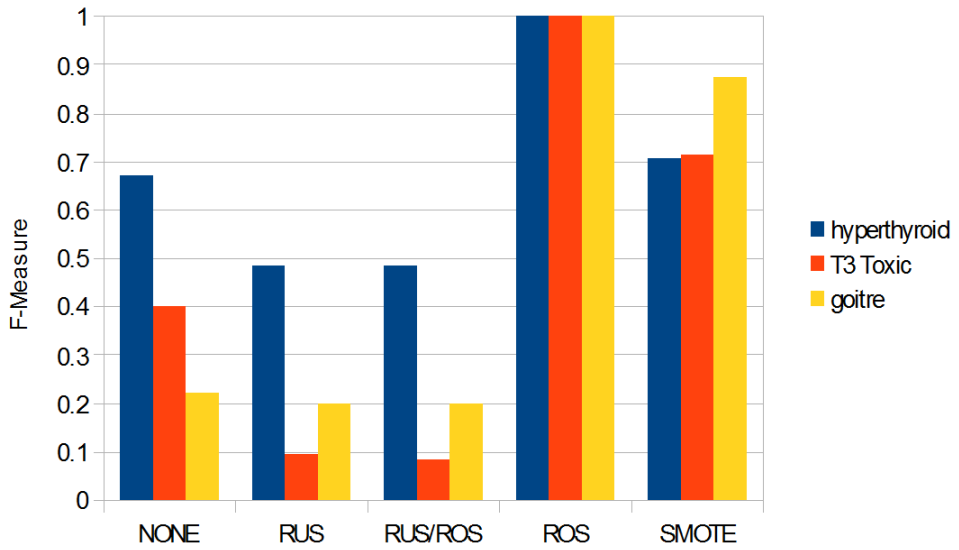

Figure 6.6: A plot of performance (F-measure) -versus- the re-sampling method for the decision tree learner.

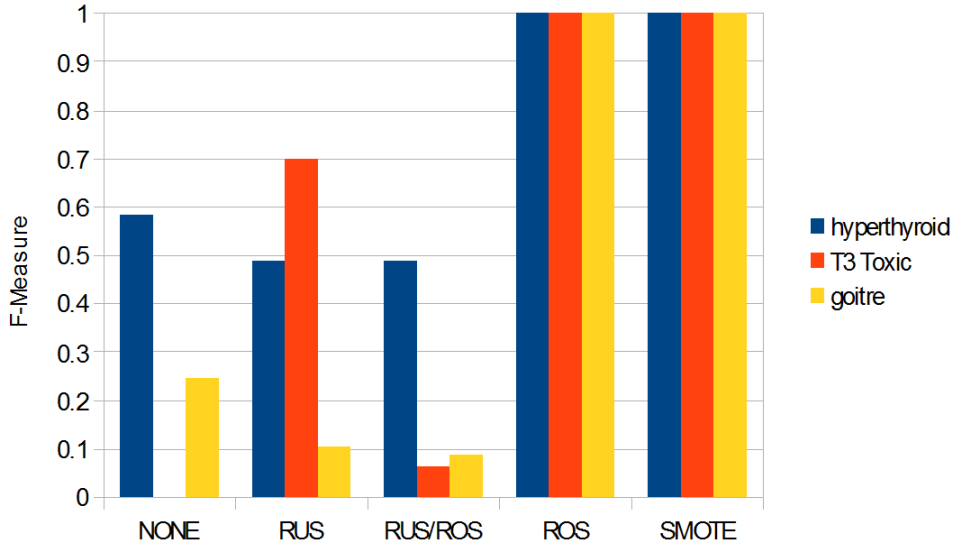

Figure 6.7: A plot of performance (F-measure) -versus- the re-sampling method for the K-NN learner.

6.4 Overview of results of Metacost methods for dealing with class imbalance in the Thyroid dataset

Six different ratios of Metacost misclassification cost are listed below in table 6.4. For each of the learners K-NN, Naive Bayes, random forest, ID3 and decision tree were experimented with each of the ratios. The results of these experiments are listed in table 6.5 and summarized below.

1. random forest. Metacost combination (d) resulted in a large increase in F-measure for the T3 Toxic class and the goitre class. However for all metacost combinations surveyed the F-measure for the hyperthyroid class could not be improved.

2. Nave Bayes. Similar to random forest, the Metacost procedure was able to increase the performance of the random forest learner's F-measure for the T3 Toxic and goitre class.

3. K-NN. For K-NN none of the combinations surveyed with the Metacost procedure resulted in a large increase in F-measure for any of the classes surveyed.

4. C4.5. For C4.5 none of the metacost combinations surveyed resulted in a large increase in F-measure for any of the classes surveyed.

5. Decision tree. For the decision tree learner the (d) combination was the only combination which produced a considerable performance increase and that was only for the goitre class (0.2222 - 0.7778 F-measure increase from when no Metacost procedure used, to when metacost combination d was chosen.)

6. ID3. For ID3 none of the metacost combinations surveyed resulted in a large increase in F-measure for any of the classes surveyed.

Futhermore, what is clear from the above analysis is that the Metacost procedure is not as effective at increasing the performance of learner to train a model to recognize the minority classes, when compared to the data methods surveyed. As one increases the number of classes, it further complicates the choosing of optimal misclassification costs. Only one of the combinations was found to increase the performance of the learner to train a model to recognize the minority class. This was only in the case of two of the learners(random forest and Nave Baye). While this combination proved capable of improving the performances of two of the classes (T3 toxic and goitre), it caused a deterioration of performance in the case of one of the classes (Hyperthyroid). Take for example the random forest learner Metacost combination d (see table 6.4). While the Performance of the T3 Toxic class and the goitre class could be improved, the hyperthyroid class showed a deterioration in performance.

Table 6.6 shows the effects of the metacost procedure to the data methods and

Table 6.4: The metacost misclassification cost for each class with the appropriate marker.

(a)				
Negative	0	100	100	100
hyperthyroid	2	0	10	10
T3 Toxic	2	10	0	10
goitre	2	10	10	0
(b)				
Negative	0	50	50	50
hyperthyroid	2	0	10	10
T3 Toxic	2	10	0	10
goitre	2	10	10	0
(c)				
Negative	0	50	50	50
hyperthyroid	2	0	10	10
T3 Toxic	2	15	0	10
goitre	2	15	10	0
(d)				
Negative	0	50	50	50
hyperthyroid	7	0	10	10
T3 Toxic	7	15	0	10
goitre	7	15	10	0
(e)				
Negative	0	100	100	100
hyperthyroid	2	0	15	15
T3 Toxic	2	10	0	15
goitre	2	10	15	0
(f)				
Negative	0	100	100	100
hyperthyroid	4	0	20	20
T3 Toxic	4	10	0	20
goitre	4	10	20	0

Table 6.5: The effects of varying the metacost misclassification cost on the F-measure for hyperthyroid, T3 Toxic and goitre classes.

Learner	metacost	hyperthyroid F	T3 Toxic F	goitre F
random forest	None	0.6749	0.2667	0.3333
random forest	A	0.2661	0.1352	0.1936
random forest	B	0.3431	0.2041	0.3243
random forest	C	0.3409	0.2000	0.3750
random forest	D	0.5847	0.3158	0.5555
random forest	E	0.2823	0.1349	0.1972
random forest	F	0.3682	0.2089	0.3111
Nave Bayes	None	0.6825	0.2667	0.3333
Nave Bayes	A	0.2661	0.6851	0.1936
Nave Bayes	B	0.4015	0.3333	0.4000
Nave Bayes	C	0.3970	0.3529	0.4444
Nave Bayes	D	0.3407	0.3158	0.3809
Nave Bayes	E	0.3418	0.3158	0.3637
Nave Bayes	F	0.4060	0.3333	0.3478
K-NN	None	0.5827	0.0000	0.2478
K-NN	A	0.5614	0.0770	0.0801
K-NN	B	0.5614	0.0770	0.0801
K-NN	C	0.5614	0.0770	0.0801
K-NN	D	0.5569	0.0000	0.1052
K-NN	E	0.5614	0.0770	0.0801
K-NN	F	0.5614	0.0770	0.0801
C4.5	None	0.8065	0.5000	0.7692
C4.5	A	0.7403	0.3415	0.6316
C4.5	B	0.7500	0.3750	0.6316
C4.5	C	0.7403	0.3871	0.6667
C4.5	D	0.7703	0.5455	0.7500
C4.5	E	0.7403	0.3871	0.6667
C4.5	F	0.7417	0.3636	0.6316
decision tree	None	0.6727	0.4000	0.2223
decision tree	A	0.5253	0.2222	0.5000
decision tree	B	0.6077	0.2381	0.5834
decision tree	C	0.6043	0.2439	0.5834
decision tree	D	0.6980	0.2857	0.7778
decision tree	E	0.5327	0.2128	0.4828
decision tree	F	0.4676	0.3044	0.5217
ID3	None	0.7049	0.5333	0.7272
ID3	A	0.4563	0.1778	0.5454
ID3	B	0.4616	0.2439	0.5714
ID3	C	0.4660	0.2381	0.5454
ID3	D	0.5882	0.2400	0.6316
ID3	E	0.4776	0.2273	0.5217
ID3	F	0.4600	0.3181	0.5217

Table 6.6: Comparing the Metacost procedure against the data methods on the F-measure for hyperthyroid, T3 Toxic and goitre classes.

Learner and pre-processing method	Hyperthyroid	T3 Toxic	goitre
random forest and None	0.6749	0.2667	0.3333
random forest and Metacost	0.5847	0.3158	0.3333
random forest and RUS	0.7187	0.9612	0.9912
nave Bayes and None	0.6825	0.2666	0.3333
nave Bayes and Metacost	0.406	0.3333	0.3478
nave Bayes and SMOTE	0.6864	0.8235	0.4999
K-NN none	0.5827	0	0.2478
K-NN Metacost	0.5614	0.077	0.0801
K-NN ROS	1	1	1
C 4.5 None	0.8065	0.5	0.7692
C4.5 Metacost	0.7703	0.5455	0.75
C.4.5 SMOTE	0.8721	0.9411	1
decision tree	0.6727	0.4	0.2223
decision tree Metacost	0.698	0.2857	0.7778
decision tree ROS	1	1	1
ID3 None	0.7049	0.5333	0.7222
ID3 Metacost	0.5882	0.24	0.6316
Id3 SMOTE	1	1	1

6.5 The use of algorithmic methods to overcome the class imbalance problem in the Thyroid dataset through the use of learners that are not susceptible to the class imbalance problem.

Previous experimentation had shown the use of the RIPPER (Cohen 1995) algorithm to overcome the class imbalance problem through its own implementation of rule learning which favours an incremental approach. The results produced by RIPPER and a number of pre-processing steps are shown below in table 6.7. The RIPPER algorithm is also combined with other learners which proved able to cope with the class imbalance in the data. These results again showed the usefulness of not only the RIPPER algorithm to outperform other algorithmic data methods, it also showed the usefulness of combining the classifications of a number of the classifiers. This was found to deliver a level of performance nearing that achieved by the data methods.

The results for the RIPPER algorithm are shown in table 6.7. When examining the table two points become apparent. The RIPPER algorithm again outperforms any of the other learners surveyed when no data method is applied. The use of the RIPPER algorithm had already been proven (Britsch et al. 2008) to be an excellent method for learning in class imbalanced datasets.

The second point regards the improvement of the performance of the learner through its combination with other class imbalance resistant learners . However the application of boosting or bagging resulted in no increase of the performance of the learner. The use of a stacking ensemble method (StackingC) to combine the predictions from a number of different models did produce gains in performance, resulting in a model that could predict very accurately across all classes. This was the combination of the RIPPER algorithm, the random forest and the C4.5 learner. Not only does this methodology deliver superior performance to other algorithmic methods , it is also easy to implement. Unlike in the case of the Metacost procedure, there is no need to empirically derive complex matrices of misclassification costs.

Table 6.7: The effects of the various pre-processing methods on the RIPPER learner with the Thyroid Dataset.

Pre-processing	Class	Precision	Recall	F-measure
None	hyperthyroid	0.7324	0.8387	0.7820
None	T3 Toxic	0.5	0.5	0.5000
None	goitre	1	0.5714	0.7272
AdaBoost	hyperthyroid	0.7895	0.7258	0.7563
AdaBoost	T3 Toxic	0.6667	0.25	0.3636
AdaBoost	goitre	1	0.5714	0.7272
bagging	hyperthyroid	0.7692	0.8065	0.7874
bagging	T3 Toxic		0	0.0000
bagging	goitre	0.8	0.5714	0.6666
multi boost	hyperthyroid	0.7719	0.7097	0.7395
multi boost	T3 Toxic	0.5	0.125	0.2000
multi boost	goitre	0.8	0.5714	0.6666
StackingC random forest RIPPER	hyperthyroid	0.8103	0.7581	0.7833
StackingC random forest RIPPER	T3 Toxic	0.5	0.125	0.2000
StackingC random forest RIPPER	goitre	1	0.7143	0.8333
StackingC random forest RIPPER C4.5	hyperthyroid	0.7937	0.8065	0.8000
StackingC random forest RIPPER C4.5	T3 Toxic	0.625	0.625	0.6250
StackingC random forest RIPPER C4.5	goitre	0.83333	0.7143	0.7692
StackingC random forest RIPPER C4.5 OneR	hyperthyroid	0.7903	0.7903	0.7903
StackingC random forest RIPPER C4.5 OneR	T3 Toxic	0.625	0.625	0.6250
StackingC random forest RIPPER C4.5 OneR	goitre	0.8333	0.7143	0.7692
StackingC random forest C4.5 RIPPER Nave Bayes	hyperthyroid	0.7937	0.8065	0.8000
StackingC random forest C4.5 RIPPER Nave Bayes	T3 Toxic	0.5714	0.5	0.5333
StackingC random forest C4.5 RIPPER Nave Bayes	goitre	0.8	0.5714	0.6666
StackingC RIPPER, random forest, C4.5 and K-NN	hyperthyroid	0.7937	0.8065	0.8000
StackingC RIPPER, random forest, C4.5 and K-NN	T3 Toxic	0.5714	0.5	0.5333
StackingC RIPPER, random forest, C4.5 and K-NN	goitre	0.8333	0.7143	0.7692

The use of stacking led to a level of performance that was approaching that of the data methods (though data methods are superior). Specifically the StackingC algorithm with random forest, RIPPER and C4.5 learner combination.

6.6 Overview of results of data-centric and algorithmic methods for dealing with the class imbalance problem in the Thyroid dataset

As in the case of the previous datasets surveyed, the data methods proved superior to the algorithmic methods. Again SMOTE and ROS proved to be the superior methods for dealing with the class imbalance problem. This was also the case with the consumer fraud and car insurance fraud dataset. However SMOTE did prove superior to ROS as can been from table 6.2. The SMOTE algorithm proved extremely effective at creating accurate samples of the minority class. SMOTE creates examples of the minority class so as to effectively over sample the minority class. It does this by synthesizing new minority samples by introducing the new samples based on examples of the minority class that are related close together. It accomplishes this through the utilization of the K-NN learner. The SMOTE algorithm had also proved very successful when used in the study of the the consumer fraud dataset. The data methods that utilized under-sampling of the majority class seemed to perform poorly, which may have been due to the very small number of samples the majority class was reduced to (200 examples). This may have inadvertently removed too many of the samples of the majority class thereby making any attempts of the learner to train a model to classify the majority class ineffective. What makes the use of sampling in the case of Thyroid dataset interesting is that experiments utilizing sampling methods to treat the class imbalance problem in multi label datasets are rare. Zhou and Liu (2010) have found the implementation of sampling methods not useful in mulitclass-problems and found them only useful for binary classification problems. They were found to be ineffective in dealing with the class imbalance and even degrading performance (Zhou an Liu 2010). Instead of using sampling methods it was proposed that a multi-class problem should be split into a number of binary classification problems and a dataset for each classification task created and imbalance within the dataset dealt with separately (Lerteerawong and Athimethphat 2008). This did not prove the case with the Thyroid dataset. Rescaling the minority classes proved an extremely powerful method in dealing with the class imbalance problem. SMOTE proved the most capable methodology of dealing with the class imbalance problem out of all the data methods surveyed. Again the algorithmic methods did not prove as useful at overcoming the class imbalance problem in the Thyroid dataset. The algorithmic methods proved not only ineffectual when compared to the data methods surveyed, but also ineffectual when compared to the their use in previous datasets. There effective use was complicated by the nature of the Thyroid dataset, as it is characterized by a multi-class label. This causes the tasks of assigning accurate misclassification costs more difficult. Also as the cost of misclassifying the various classes was not known, a number of different combinations were experimented with. None of these proved effective at dramatically increasing the performance of any of the learners surveyed. Experiments to calculate the optimal misclassification costs using genetic algorithms did not prove successful. While cost sensitive learning is frequently cited as functioning superior to data methods intelligent re-sampling methods have been shown to be more useful, they can provide previously unseen information in the case of SMOTE or remove superfluous or unnecessary information in the case of under sampling (Kotsiantis et al. 2006). The most effective algorithmic methods to treat the class imbalance problem in the Thyorid dataset was the choice of learner. The RIPPER algorithm had proved adept at overcoming the class imbalance problem in the consumer fraud dataset. This was also the case. Using the StackingC ensemble method it was possible to combine it with the random forest and C4.5 classifier to increase its performance. This produced a classifier which approached the performance of the learners trained with balanced datasets attained through the use of ROS and SMOTE.

Chapter 7

Discussion and Conclusion

7.1 The study of the class imbalance problem across all datasets

Fraud is an important problem both in economic terms and in terms of its social-political consequences. Fraud if not combated can lead to a large monetary cost. The political and social consequences of fraud can also be severe. Fraud can lead to legitimization of revenues from serious crimes such as drug trafficking. This leads to criminal organisations gaining not only economic power but also political power as their monies become legitimized. There are two ways to combat fraud, these are:

1. Fraud prevention. This is generally achieved through the use of procedures to prevent fraud taking place.

2. Fraud detection. Detection is generally done through the use of specialized fraud detection teams or more recently, through the use of data mining techniques.

When using a data mining approach there are two approaches to take; an unsupervised data mining approach or a supervised approach. Supervised methods are used in this study. Particularly the classification of transaction as fraudulent or not fraudulent. One of the biggest problems when undertaking a classification data mining problem is the large class imbalance in the data, caused by the relative scarcity of fraudulent transactions when compared to those that are non-fraudulent. Therefore the study concerned itself with methods for overcoming this problem. There are two basic approaches for doing this:

1. Data methods. These methods employ oversampling or under sampling to rebalance the distribution of the different classes.

2. Algorithmic methods. These include introducing a cost of mis-classification or by using learners which have been hardened to deal with the class imbalance problem or have intrinsic ability to deal with the class imbalance.

One of the most important learning outcomes from the study was the realization of the difficulty in designing the correct experiment so as to correctly evaluate the performance of the data methods. When surveying the literature, this is an often overlooked consideration. Three different experimental techniques were evaluated, these were:

1. The use of ten fold cross validation (except in the case of the Tyroid dataset, were a 6 fold cross validation was used) to train and test the model.

2. The use of a blind training set made up of thirty percent of the original dataset. The model is trained with the balanced dataset.

3. The use of all the original data to test the model. Train the model using the balanced dataset using ten fold cross validation (or in the case of the Thyroid dataset six fold cross validation).

The first two methods proved problematic. The first as it over-estimated the performance of the methods which utilised random oversampling and was not useful at evaluating the performance of methods that relied upon replicating artificial samples or under sampling. In the case of the former point, this is due to in the case of under sampling, large numbers of the majority class are discarded in under sampling, training against the balanced dataset will only test against a small subset of the majority class. In the case of the latter (SMOTE) if over generalization of the minority class has occurred when 'SMOTING' the minority dataset, the model might deliver good performance when tested against the balanced dataset containing the artificial replicants. However if tested against the original dataset, if the replicants created are not accurate replicants of the minority class, this will result in poor performance when evaluated against the original dataset.

The second methodology delivers artificially low results for methods that utilize oversampling. This is due to the performance of the model being extremely sensitive to any decrease in the number of samples of the minority sample. The reduction in the sample size by thirty percent to create a blind test sample results in reducing the amount of examples by an excessive amount , thus not allowing sufficient examples of the minority class to train an efficient model (Kotsiantis et al. 2006).

These problems are overcome through the use of the third experimental set-up. This set-up trains against the balanced dataset but evaluates the model against the original dataset. However the third experimental set-up did highlight the over-fitting that can occur when the ROS data method is used. This is due to when applying the ROS methodology, exact copies of existing minority classes are created. These are then added to the existing samples of the minority class until the size of the minority class equals that of the majority class. However while this redistributes the data correctly, it does lead to over-fitting.

Across all datasets surveyed using all learners, data methods proved to be not only easier to implement but also much more proficient at improving the performance of a learner than the algorithmic methods surveyed. For all the datasets surveyed, with all learners, the data methods which proved most adept at improving the performance of the learner were SMOTE and ROS, with SMOTE delivering the greatest increase in performance. The second most proficient method proved to be the ROS data method, however in the case of one of the datasets (car insurance fraud dataset), the SMOTE algorithm proved completely incapable of synthesizing minority classes accurately. This was not the case with the consumer fraud dataset and the Thyroid dataset. This increase in performance has been explained as due to the SMOTE algorithm creating new artificial replicants, thereby increasing the size of the feature space of the minority class. This increase in the size of the feature space of the minority class to a sufficiently large size allows a learner to train an accurate model on the dataset, (Chawla 2003). For data methods utilizing a basic under sampling approach, across all datasets the method universally delivered the poorest level of performance out of all the data methods surveyed. This is due to the large number of majority samples that have been removed, which prove important in training the model. To overcome this problem, a progressive under sampling of the majority class approach was investigated with the consumer fraud dataset. By experimenting with keeping more of the majority class, and not discarding most of it, an increase in performance was noted (Kotsiantis and Pintelas 2003). Progressive under sampling overcomes the main drawback of the under sampling method. That is the discarding of potentially valuable majority class examples that are important in the induction of a model. By calculating the optimal ratio of number of majority samples to the number of minority samples that results in the highest F-measure, the optimal ratio of majority class to minority class can be calculated. For the algorithmic methods, both the Metacost procedure and probability thresholds did not prove as useful at improving performance of all learners

across all three of the datasets surveyed. This is in conflict to reports in the literature which have shown that cost sensitive learning performs better than re-sampling techniques (Domingos 1999 and Japkowicz and Stephen 2002). Further it should be noted that some learners proved to be able to cope better than others with imbalanced datasets. For instance random forests and the C4.5 learner, proved more capable of dealing with the large class imbalance in the consumer fraud detection dataset when compared to other learners. The propensity of ensemble methods for dealing with large class imbalances in data has been noted previously (Chawla 2009). Rule induction learning implemented using RIPPER proved to be able to train a classifier that was especially able to cope with the class imbalance in the data. For the RIPPER algorithm this was explained due to the rule induction approach it employs which effectively learns a set of rules for each class, even the rarest of classes. The performance of the RIPPER algorithm was found to be further improved through the use of ensemble methods. In the case of consumer fraud dataset, through the use of boosting while in the case of the Thyroid dataset through stacking (the StackingC method). Out of the two methodologies surveyed (data and algorithmic), data methods proved to be the most useful. Random oversampling of the minority class has been found to suffer from of the exact replication of the minority class, which can lead to over-fitting. SMOTE, on the other hand, creates a less specific decision space for the minority group (Wang and Japkowicz 2004). However the SMOTE algorithm is far from perfect and can without due care or understanding (of the majority class) generalize the minority class too much without paying due attention to the majority class. This results in creating minority class sample which have qualities in common with the majority class and are not representative of the minority class. Thereby leading to the problem of 'class mixture' (Wang and Japkowicz 2004).

7.2 The study of the data methods for overcoming the class imbalance problem across all datasets

Table 7.1 shows a summary of the different data methods for dealing with the different datasets and different learners. Across all datasets and learners universally SMOTE and ROS proved to be the best data methods and also the best overall method for dealing with the class imbalance problem. The RUS method proved to be universally the least successful method for dealing with the class imbalance.

Table 7.1: The best in class and worst in class data method for all learners and datasets.

Learner	Thyroid Best	Thyroid Worst	consumer fraud Best	consumer fraud Worst	car insurance Best	car insurance Worst
WJ-48	SMOTE	RUS	ROS	RUS	ROS	RUS
random forest	RUS	SMOTE	SMOTE,ROS	RUS	ROS	RUS
nave Byaes	SMOTE	RUS	ROS/RUS	RUS	-	-
ID3	SMOTE	RUS,RUS,ROS/RUS	SMOTE	RUS	ROS	RUS
decision tree	ROS, SMOTE*	RUS,ROS/RUS	SMOTE	RUS	ROS	RUS
K-NN	SMOTE,ROS	ROS/RUS	SMOTE	RUS	ROS	RUS

Table 7.2: The best in class and worst in class algorithmic method for all datasets.

Dataset	Best in class	Worst in class
Car insurance fraud dataset	- (only one evaluated)	- (only one evaluated)
Consumer fraud dataset	Choice of learner	probability thresholds
Thyroid	Choice of learner	Metacost

7.3 The study of the algorithmic class imbalance problems across all datasets

Across all datasets and with all learners surveyed, the algorithmic methods proved inferior to the data methods. The algorithmic methods experimented with were the Metacost procedure, probability thresholds and choice of learners that are not susceptible to the class imbalance problem. The results are shown in 7.2 which show the best and worst in class algorithmic methods methods.

The Metacost procedure (Domingos 1999) works by attaching a mis-classification cost on each class. By loading the Metacost misclassification cost of the minority class much higher than that of the minority class it is hoped that as the label predictions are computed according to the classification costs, the cost of misclassification becomes very expensive. As the learner tries to minimize these misclassification costs the problem of the imbalance in the data is addressed. A number of other problems or complexities arise when using these methods. These are related to deciding the correct misclassification costs to place on the different classes. While this can sometimes be derived form the literature, as in the case of the car insurance fraud dataset, it usually must be derived empirically. This adds a considerable effort in deriving the optimal Metacost procedure misclassification costs or the correct probability threshold. It requires significant time and effort to optimize the mis-classifications cost of the Metacost procedure or the probability thresholds, through the running of a number of experiments to empirically derive the correct cost of misclassification. Efforts to try and overcome this problem by using genetic algorithms to try and optimize the ideal misclassification costs or probability thresholds proved ineffectual. The processes polling out and running out of computational and memory resources. In some cases the process would run for an extended period (two - three days) of time without being able to return an optimal misclassification cost or probability threshold. Choice of learner can also be considered an algorithmic method for dealing with the class imbalance problem. Once again ensemble methods proved extremely useful. random forests proved to be a very capable method of dealing with imbalanced datasets, more so than any of the other learners surveyed apart from simple RI methods. In the case of the consumer fraud dataset and the Thyroid dataset, the random forest learner delivered performance far superior to the other learners, except a number of RI methods. With the consumer fraud dataset, the RIPPER algorithm proved to deliver good performance with no-pre-processing. Attempts to improve the performance of the RIPPER algorithm to deliver better performance, through the use of bagging and weighting the the training set was also evaluated. This methodology proved successful, improving the performance of the learner further. Other simple RI methods, such as Ridor proved equally capable and were able to deliver better performance then more sophisticated learners like SVM or random forests.

7.4 Data methods or algorithmic methods for treating class imbalanced datasets?

In conclusion, data methods proved capable of delivering greater performance (in terms of F-measure), when treating the class imbalance in the datasets surveyed. They also proved easier to implement and did

not lead to sizeable increases in training time or resources needed. However considerable care must be taken when assessing the performance of the data methods. One must be careful to train the model on the balanced dataset but not test the performance on the balanced dataset. If this is done it can lead to an incorrect assessment of the performance of a particular data method with a specific learner. Besides taking care not to test the performance of a particular data method on the balanced dataset, the use of blind samples can also lead to incorrect performance metrics being recorded. This is a result of samples of the minority class being removed from the training dataset to be used in the blind test dataset. Both ROS and SMOTE proved to be the best performing methods for treating the imbalance in the data. However ROS may be criticized as a method as it can lead to over-fitting a model as it over trains a model to recognize a small number of minority classes. SMOTE addresses this by creating artificial replicants and thereby creating a less specific feature space for the minority group. However if the SMOTE algorithm has proved ineffectual at replicating the characteristics of the minority class, it will result in a situation where the minority class samples qualities are too similar to those of the majority class. This results in the problem of 'class mixture' and the resulting model will mis-classify classes of the minority class as members of the majority class. Algorithmic methods did not prove as useful at treating the class imbalance in the data. The Metacost procedure while it was found to improve the performance of the learner on a number of different datasets it suffered from a number of shortcomings; (a) it did not improve performance to the same extent as the data methods or (b) in the case of Thyroid dataset it could only improve the predictive accuracy of one or two classes out of the three minority classes.

Furthermore, the optimal misclassification costs must be derived empirically. This takes substantial time and effort. Problems using probability thresholds also proved similar to those encountered using the Metacost procedure.

Efforts to use a genetic algorithm to calculate the optimal misclassification costs proved not to be successful. However the benefits to using this algorithmic approach (Metacost procedure or probability thresholds) is that no change in the distribution of the data is required or no introduction of artificial samples are required, which can lead to an increase in training time.

It should be noted that the best performing algorithmic method (in the case of the consumer fraud dataset) was the use of the RIPPER, boosted RIPPER and boosted RIPPER were the sample set was weighted by weighting it by average value. RIPPER also proved very useful in learning an effective model in the multi-class, class-imbalanced Thyroid dataset. While the RIPPER algorithm proved to be superior to all other learners surveyed when used without any pre-processing methods, when used combined with other models using a stacking ensemble method it delivered performance quite close to that delivered through the use of data methods (in the case of the thyroid dataset). In the case of the Thyroid dataset, combining the predictions from RIPPER with other classifiers such as C4.5 and random forest proved to be capable of delivering a classifier that was almost as performant as models created with the data methods.

While data methods proved to be the most successful methods for dealing with the class imbalance problem, they are far from the only methods available. Algorithmic methods include choice of learner and introducing a cost of misclassification, either through the Metacost procedure or through the use of probability thresholds. Data methods are relatively simple to implement and widely available in RapidMiner, Weka and R and can be deployed relatively simply without the requirement of substantial development. While the SMOTE algorithm is currently only available in R and Weka, it can be easily integrated into RapidMiner from R through the use of the RapidMiner R extension (Mierswa 2010).

Algorithmic methods proved to be inferior to data methods with the exception of simple RI learners like the RIPPER learner. The performance of the RIPPER learner (choice of learner) was found to be superior to all other algorithmic methods surveyed. The performance of the RIPPER algorithm was found to be increased through the use of ensemble methods. Boosting in the case of the consumer fraud dataset and stacking in the case of the Thyroid dataset resulted in increased performance of the learner. Attempts to combine a number of learners which proved not to be susceptible to the class imbalance

problem through the use of stacking or voting did not considerably improve the performance of the RIPPER algorithm in the case of the consumer fraud dataset but did prove useful in the case of the Thyroid dataset.

7.5 Future work

Any future evaluation should try and utilize real-world closed source data. This would highlight the usefulness of the different data and algorithmic methods. Another factor which could not be evaluated with the studies carried out, is the effects of concept drift. This is the idea of a learning problem changing over time. Fraud is known to be cyclical in nature and new methods are constantly evolving. Learners are said to be able to handle concept drift if they can:

1. Quickly change to take into account concept drift.
2. Not be sensitive to noise and be able to differentiate between concept drift and noise.
3. Be able to distinguish recurring patterns.

Both data methods and algorithmic methods should be evaluated to see which methods are least susceptible to concept drift. This study would have to be carried out over a sizeable period of time so as to evaluate the effectiveness of the learners at overcoming concept drift. The data methodologies proved by far the most useful of all the methods surveyed for handling the class imbalance in datasets. However very basic methodologies were utilized. What should be experimented with further is the use of focussed sampling methods and more advanced developments of the SMOTE algorithm to see if they lead to improved performance. Another area of interest would be that of developing a framework for overcoming the class imbalance problem within a dataset. There is currently no framework for doing so and as it is an important issue in KDD, it should be addressed. Consideration should be taken on how to integrate this into the existing data mining methodologies, such as CRISP-DM or SEMMA.

Data methods proved to be far more successful in overcoming the class imbalance problem than algorithmic methods . While the most successful algorithmic methods proved to be the choice of learner. Both introducing a cost of misclassification and hardening a learner to the class imbalance problem proved to be ineffective. This would favour the use of these methods (data methods) in the design of any methodology for overcoming the class imbalance problem. Consideration should also be given to what performance metrics will be used to judge the performance of the methods.

Bibliography

[1] Barendela, R., Sanchez, J.S., Garcia, V., Rangel, E. (2003) Strategies for learning in class imbalance problems , Pattern Recognition vol.36, pp.849-851.

[2] Barse, E. Kvarnstrom, H. and Johnston E. (2003) Synthesizing data for fraud detection systems, Proceedings of the 19th Annual computer security applications conference, pp.384-395.

[3] Batista,G. E. A. P. A., Prati,R.C. and Monard.,M.C. (2004) A study of the behavior of several methods for balancing machine learning training data. SIGKDD Ex- plorations, vol.6, issue.1, pp.20-29.

[4] Batista, G.E.A.P.A. and Monard, M.C. (2003) An Analysis of Four Missing Data Treatment Methods for Supervised Learning. Applied Artificial Intelligence, vol.17, no.5, p.519-533.

[5] BBC (2012) Fraud levels increased in 2011, says BDO, retrieved from http://www.bbc.co.uk/news/business-16467068 on the 5th of February 2012

[6] Becker, B.G. (1997) using Mineset for knowledge discovery, IEEE Computer graphics and Applications, pp. 75-78.

[7] Berkeley (2012) random forests, retrieved from http://www.stat.berkeley.edu/breiman/RandomForests on the 20th of April 2012.

[8] Brachman, R.J., Khabaza, T., Kloesgen, W., Piatetsky-Shapiro,G., Simoudis, E. (1996) Mining business databases, Communications of the ACM, vol.39, issue.11, pp. 42-48.

[9] Breiman, L. (1996). Bagging predictors. Machine Learning, vol.24, no.2, pp.123-140.

[10] Britsch, M., Gagunashvili, N., Schmelling, M. (2008) Applications of the rule-growing algorithm RIPPER to particle physics analysis, Advanced computing and analysis techniques in physics research.

[11] Brockett, P.L., Xia,.X., Derrig, R.A. (1998) Using Kohonen's Self Organizing Feature Map to Uncover Automobile Bodily Claims Fraud, The Journal of Risk and Insurance, vol. 65, no. 2, pp.245-274.

[12] Busch, R.S. (2008) Healthcare Fraud, Auditing and Detection Guide, Wiley.

[13] Chang,R., Ghoniem,M., Kosara,R.,Ribarsky,W.,Yang,J. Suma,E. Ziemkiewicz, C., Kern,D. and Sudjianto.A. (2007) WireVis: Visualization of Categorical, Time-Varying Data From Financial Transactions. In Proceedings of the 2007 IEEE Symposium on Visual Analytics Science and Technology (VAST '07). IEEE Computer Society, Washington, DC, USA, pp.155-162.

[14] Chawla, N. (2003) C4.5 and Imbalanced Datasets: Investigating the effect of sampling method, probabilistic estimate, and decision tree structure, Workshop on Learning from Imbalanced Datasets II, ICML, Washington DC, pp.1-8.

[15] Chawla, N., Bowyer, K., Hall, L., and Kegelmeyer, P. (2000). SMOTE: Synthetic Minority Over-sampling TEchnique. In International Conference of Knowledge Based Computer Systems, National Center for Software Technology, Mumbai, India,Allied Press, pp. 4657.

[16] Chawla, N.V., Hall, L.O., Bowyer, K.W., Kegelmeyer, W.P. (2002) SMOTE: Synthetic Minority Oversampling Technique. Journal of Artificial Intelligence Research, vol.16, pp.321-357.

[17] Chawla,N. (2009) Mining when classes are imbalanced, rare events matter more and error have cost attached, Data, Inference, Analysis and Learning lab, retrieved from http://www.siam.org/meetings/sdm09/chawla.pdf on the 23rd of April 2012.

[18] Chen, M-u.,Chen,L-s.,Hsu,C-C. and Zeng, W.R. (2008) An information granulation based data mining approach for classifying imbalanced data, Information sciences vol.178, pp.3214-3227.

[19] Chen, C. Liaw,A. and Breiman, L. (2004) Using random forest to learn imbalanced data retrieved from http://www.stat.berkeley.edu/tech-reports/666.pdf on the 23rd April 2012.

[20] Chumphol,B.,Sinapiromsaran,K. and Lursinsap,C. (2012) DBSMOTE: Density-Based Synthetic Minority Over-sampling TEchnique, vol.36, issue.3, pp.664-684.

[21] Cohen, W.W. (1995) Fast effective rule induction. In Proceedings of the Twelfth International Conference on Machine Learning, Lake Tahoe, California.

[22] Conning and Company (1996) Conning's and Company, Insurance fraud: the quiet catastrophe, Hartford, Connecticut.

[23] Dataprix (2012) (a) Cost sensitive learning retrieved from http://www.dataprix.net/en/example-12-cost-sensitive-learning on the 16th of March 2012.

[24] Dataprix (2012) (b) Learning cost-sensitive and graphic ROC, retrieved from retrieved from http://www.dataprix.net/en/example-10-learning-cost-sensitive-and-graphic-roc on the 8th of March of 2012.

[25] Dauer, C. (1993) Automation helps to combat fraud, National underwriter, Property/Casuality/Employee Benefits, vol.97, no.3, pp.3.

[26] Dekker, G.W., Pechenizkiy, M. and Vleeshouwers, J.M. (2009) Predicting Students Drop Out: A Case Study, Educational Data Mining, pp.41-50.

[27] Derrig, R.A., Johnston,D.J. and Sprinkel, E.A. (2006) Auto Insurance Fraud: Measurements and Efforts to Combat It, Risk Management and Insurance Review, vol. 9 no.2, pp.109-130.

[28] Dionne, G. and Belhadji, E.B. (1996) Evaluation de la Fraude lAssurance Automobile au Quebec, Assurances, vol.65, pp. 365-394.

[29] Drazin, S. and Montag,M. (2012) Decision Tree Analysis using Weka, retrieved from http://www.samdrazin.com/classes/een548/project2report.pdf on the 25th of April 2012.

[30] Drummond, C. and Holte, R. C. (2003) C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling, in Proc. Working Notes ICML Workshop Learn.Imbalanced Datasets, Washington DC.

[31] Duda, R.O. and Hart, P.E. (1973) Pattern Classification and scene analysis. Wiley, New York.

[32] Dzeroski, S. and Zenko, B. (2004) Is combining classifiers with stacking better than selecting the best one?, Machine Learning, vol.54, issue.3, pp.255-273.

[33] Fawcett, T.(2003) "In vivo" Spam filtering: a challenge problem for KDD, SIGKDD Explorations, vol.5,no.2, pp.140-148.

[34] FBI (2008) Federal Bureau of Investigation, New York Division, Department of Justice, United States, retrieved from: http://newyork.fbi.gov/dojpressrel/pressrel08/nyfo121108.htm on the 12th of January 2012.

[35] Gaines, B.R. and Paul Compton, P. (1995). Induction of Ripple-Down Rules Applied to Modelling Large Databases. J. Intell. Inf. Syst., vol.5, no.3, pp.211-228.

[36] Ghezzi, S.G. (1983) A private network of social control: Insurance Investigation Units, Social Problems, vol.30, no.5 pp 521-530.

[37] S. Ghosh, D.L. Reilly (1994) Credit card fraud detection with a neural-network, in: Proceedings of the Annual International Conference on System Science, pp. 621630.

[38] The Guardian (2009) Online credit card fraud where you live, postcode by postcode, retrieved from: http://www.guardian.co.uk/news/ datablog/2009/sep/11/credit-card-fraud-online-statistics-postcode-data on the 10th of January 2012.

[39] Guo, X., Yin, Y., Dong, C., Yang, G.., Zhou, G. (2008) On the class imbalance problem, Natural computation, INC 2008, Fourth International Conference on Natural Computing.

[40] Han,H., Wen-Yuan,W. and Bing-Huan, M. (2005) Advances in Intelligent Computing (2005), pp.878-887.

[41] Holte, C., Acker, L.E. and Porter L.E. (1989) Concept Learning and the Problem of Small Disjuncts, In Proceedings of the Eleventh Joint International Conference on Artificial Intelligence , pp.813-818.

[42] Holte, R.C. (1993) Very simple classification rules perform well on most commonly used datasets, Machine Learning, vol.11, pp.63-91.

[43] Holtfreter, K., Van Slyke, S., Bratton, J. and Gertz, M. (2008) Public perceptions of white-collar crime and punishment, Journal of Criminal Justice, vol.36, no.1, pp.50-60.

[44] I2 (2012) Analyst's notebook retrieved from http://www.i2group.com/us/products/analysis-product-line/analysts-notebook on the 18th of January 2012.

[45] inside-R (2012) SMOTE DMwR SMOTE algorithm for unbalanced classification problems, retrieved from http://www.inside-r.org/packages/cran/DMwR/docs/SMOTE on the 12th of February 2012.

[46] Insurance Daily (2011) The cost of insurance fraud: 2 billion a year retrieved from http://www.insurancedaily.co.uk/2010/01/23/the-cost-of-insurance-fraud-2bn-a-year/ on the 10th of january 2012.

[47] Insurancelink (2012) About Insurancelink, retrieved from http://info.insurancelink.ie/default.htm on the 23rd of February 2012.

[48] Insurance Weekly (2010) Three pay costly price for fraud, retrieved from http:// www.insuranceweekly.co.uk/tag/serious-organised-crime-agency on the 10th of the January 2012.

[49] ISO (2012) Fighting Insurance Fraud - Survey of Insurer Anti-Fraud Efforts retrieved on the 10th of January 2012 retrieved from: http://www.iso.com/Research-and-Analyses/Studies-and-Whitepapers/Fighting-Insurance-Fraud-Survey-of-Insurer-Anti-Fraud-Efforts.html on the 10th of January 2012.

[50] Jacobs, F. (2007) Vital statistics of a deadly campaign, the Minard Map, retrieved from http://bigthink.com/ideas/21281 on the 21st of February 2012.

[51] Japkowicz, N. (2000 b) Learning from imbalanced Datasets: a comparison of various strategies, Papers from the AAAI Workshop, Technical Report WS-00-05 , Japkowicz, N. (editor), abstract, paper , pp. 10-15.

[52] Japkowicz, N. and Stephen, S. (2002) The class imbalance problem: a systematic study, Journal Intelligent Data Analysis, vol.6, issue.5, pp.429449.

[53] Japkowicz, N. (2000) The Class Imbalance Problem: Significance and Strategies. In Proceedings of the 2000 International Conference on Artificial Intelligence (IC-AI2000): Special Track on Inductive Learning Las Vegas, Nevada.

[54] Jensen, D. (1997) Prospective assessment of AI technologies for fraud detection and risk management. In Proceedings of the AAAI Workshop on AI Approaches to Fraud Detection and Risk Management, pp.3438.

[55] Jo, T. and Japkowicz, N. (2004) Class imbalances versus small disjuncts. SIGKDD Explorations, vol.6, no.1, pp.4049.

[56] Johnston, M. (2006) Pursuing Auto Insurance Fraud as Organized Crime Pays Off, retrieved from: http://www.insurancejournal.com/news/national/2006/09/01/71947.htm on the 10th of January 2012.

[57] Kang,P., Cho,S. and Maclachlan, D.L. (2012) Improved response modelling based on clustering, under-sampling and ensemble, Expert systems with Applicaitons vol.39 pp.6738-6753.

[58] Karatzoglou, A. and Meyer, D. (2006) Support vector machines in R, Journal of statistical software , vol.15 issue.9, pp.1-28.

[59] Kokkinaki, A. (1997) On atypical database transactions: identification of probable fraud using machine learning for user profiling in: Proceedings of the IEEE knowledge and Data engineering Exchange Workshop, Newport Beach, pp.107-113.

[60] Kotsiantis, S., Kanellopoulos, D. and Panayiotis, P. (2006) Handling imbalanced datasets:a review, GESTS International Transactions on Computer Science and Engineering, vol.30, pp.25-36.

[61] Kotsianitis, S. and Pintelas, P. (2003) Mixture of Expert agents for handling imbalanced Datasets, Annals of Mathematics, Computing and TeleInformatics, vol.1, no.1, pp.46-55.

[62] Kubat, M., Holte, R. and Matwin, S. (1998). Machine Learning for the Detection of Oil Spills in Satellite Radar Images. Machine Learning, vol.30, pp.195-215.

[63] Lanza, R.B. (2000). Using digital analysis to detect fraud: Review of the DATAS statistical analysis tool. Journal of Forensic Accounting. vol.1, pp.291-296.

[64] Lerteerawong, B. and Athimethphat, M. (2011) An Empirical Study of Multiclass Classification with Class Imbalance Problems, International conference on business and Information 2011, retrieved from http://bai-conference.org/BAI2011/Papers/8.Others/8164.pdf on the 22nd of May 2012.

[65] Li,J., Li,H. and Yu,J.L. (2011) Application of Random-SMOTE on Imbalanced Data Mining, bife, 2011 Fourth International Conference on Business Intelligence and Financial Engineering, pp.130-133.

[66] Li,Y. and Zhang.X. (2011) Improving k nearest neighbor with exemplar generalization for imbalanced classification, In Proceedings of the 15th Pacific-Asia conference on Advances in knowledge discovery and data mining, vol.2 (PAKDD'11), pp.321-332.

[67] Ling, C. and Li, C. (1998). Data Mining for Direct Marketing Problems and Solutions, In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98) New York, NY. AAAI Press.

[68] Liu, X.Y., Wu, J. and Zhou, Z.H. (2009) Exploratory undersampling for Class-Imbalance Learning, IEEE transaction on systems, man and cybernetics- part b: cybernetics, vol. 39, no.2, pp.539-550.

[69] Major, J.A., Riedinger, E.F.D (1992) EFD: a hybrid knowledge/ statistical based system for the detection of fraud, International Journal of Intelligent Systems, vol.7, pp.687-703.

[70] Matheus, C.J., Piatetsky-Shapiro, G. (1996) Selecting and reporting what is interesting in: Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurunswamy, R. (Eds.) (1996) Advances in Knowledge Discovery and Data Mining, AAAI PRess/MIT Press, Cambridge, pp.495-515.

[71] Mierswa, R. (2010) R extension for RapidMiner: SneakPeak! retrieved from http://www.analyticbridge.com/group/rprojectandotherfreesoftwaretools/ forum/topics/r-extension-for-RapidMiner on the 12th of April 2012

[72] Myatt, G.J. and Johnson W.P. (2009) Making Sense of Data II, Wiley.

[73] NHCAA (2012) Healthcare fraud, retrieved from: http://www.nhcaa.org on the 7th of February 2012.

[74] Panigrahi, S., Kundu, A., Sural, S. and Majamdar, A.K. (2009) Credit card fraud detection: A fusion approach using Dempster-Shafer theory and Bayesian learning, Information fusion, vol.10, no.4, pp.354-363.

[75] Phua, C., Lee, V., Smith, K., and Gayler, R. (2005) A Comprehensive Survey of Data Mining-based Fraud Detection Research. Artificial Intelligence Review, pp.1-14.

[76] Phua, C., Alahakoon,D. and Lee,V. (2004) Minority Report in Draud Detection: Classification of Skewed Data, SIGKDD Explorations vol.6, no.1 pp.50-59.

[77] Quinlan,J.R., Compton,P.J., Horn,K.A., and Lazurus,L. (1986). Inductive knowledge acquisition: A case study. In Proceedings of the Second Australian Conference on Applications of Expert Systems. Sydney, Australia, pp.137-156.

[78] Quinlan,J.R. (1986). Induction of decision trees. Machine Learning, vol.1, pp.81-106.

[79] rapid-i.com (2012) Uses of Class com.rapidminer. , retrieved from http://rapid-i.com/api/rapidminer-5.1/com/rapidminer/operator/preprocessing/class-use/AbstractDataProcessing.html on the 12th of April 2012.

[80] Raskutti, B. and Kowalczyk,A. (2004) Extreme rebalancing for SVM's: a case study. SIGKDD Explorations, vol.6, no.1 pp.60-69.

[81] Rijsbergen, C. V. (1979) Information Retrieval. London, Butterworths.

[82] Senator, T.E., Goldberg, H.G., Wooton, J., Cottine, M.A., Umar-Khan,A.F., Klinger, C.D., Llamas, W.M., Marrone, M.P. and Wong, R.W.H. (1995) The financial crimes enforcement network AI systems (FAIS) identifying potential money laundering from reports of large cash transactions, AI Magazine, pp. 21-39.

[83] Senigupta, S. (2011)Phone hacking tied to terrorists, retrieved from: http://www.nytimes.com/2011/11/27/world/asia/4-in-philippines-accused-of-hacking-us-phones-to-aid-terrorists.html %_r=3 on the 10th of January 2012.

[84] Sparrow, M.K. (2000) License to Steal, How fraud bleeds america's health care system, Westview Press.

[85] Steinbech, M., Kumar, V., and Pang-Ning, T. (2006) Introduction to data mining, Addison Wesley.

[86] Su, C.-T., Chen, L.-S. and Yih, Y. (2006) Knowledge acquisition through information granulation for imbalanced data, Expert System with Applications, vol.31, pp.531541.

[87] Su, C-T., Chen, L-S. and Chiang, T-l. (2006 b) A neural network based information granulation approach to shorten the cellular phone test process, Computers in Industry, vol.57, pp.412-423.

[88] Sun, Y., Kamela, M.S., Wong, A.K.C. and Wang,Y. (2007)Cost-sensitive boosting for classification of imbalanced data, Pattern Recognition, vol.40, issue.12, pp.3358-3378.

[89] Tang, Y., Krasser, S. and Judge ,P. (2006) Fast and effective spam sender detection with granular SVM on highly imbalanced server behaviour data, 2nd international conference on collaborative computing:networking, Applications and worksharing.

[90] Ting, K.M. and Witten, I.H. (1999) Issues in stacked generalization. Journal of Artificial Intelligence Research, vol.10, pp.271-289.

[91] Tomek, I. (1976). Two Modications of CNN. IEEE Transactions on Systems, Man and Cybernetics, vol.6, pp.769-772.

[92] Torgo, L. (2011) Data mining with R, learning with case studies, CRC press.

[93] Torgo, L. (2012 a) Package DMwR, retrieved from http://cran.r-project.org/web/packages/DMwR/DMwR.pdf on the 29 th of April 2012.

[94] Torgo, L. (2012 b) Consumer fraud detection dataset, retrieved from http://www.liaad.up.pt/ ltorgo/DataMiningWithR/datasets4.html on the 29th of April 2012.

[95] Tufte, E.R. (2001) The Visual Display of Quantitative Information (2nd edition ed.). Graphics Press.

[96] UCI (2012) Thyroid disease dataset retrieved from http://archive.ics.uci.edu/ml/datasets/Thyroid+Disease on the 12th of March 2012.

[97] United States Department of Justice. (2005). Fiscal year 2005 budget and performance summary.

[98] Vapnik, V.N. (1998) Statistical Learning theory, New York, John Wiley and Sons.

[99] Viaene, S., Derrig, R.A.,Baesens, B. and Dedene, G. (2002) A comparison of state of the art classification techniques for expert automobile insurance claim fraud detection, vol.68, no.3, pp.373-421.

[100] Viola, P. and Jones, M. (2001) Rapid object detection using a boosted cascade of simple features, Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on (CVPR),vol.1 pp.511-518.

[101] Wang, J., Liao, Y.,Tsai, T. and Hung ,G. (2006) Technology-based financial frauds in Taiwan: issue and approaches, IEEE Conference on: Systems, Man and Cyberspace Oct pp.11201124.

[102] Wang, S. Tang, K. and Yao, X. (2009) Diversity exploration and negative correlation learning on imbalanced Datasets. In Proceedings of the 2009 international joint conference on Neural Networks (IJCNN'09). IEEE Press, Piscataway, NJ, USA, pp.1796-1803.

[103] Wang, B.X., Japkowicz, N. (2004) Imbalanced Dataset learning with Synthetic Examples retrieved from www.iro.umontreal.ca/ ̃lisa/workshop2004/slides/japkowicz/_slides.ppt on the 1st of April 2012.

[104] Weiss, G. M. (2004). Mining with rarity: A unifying framework. SIGKDD Explorations, vol.6, no.1, pp.719.

[105] Weiss, G.M., McCarthy, K.and Zabar,B. (2007) Cost-Sensitive Learning vs. Sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs?, DMIN, pp.35-41.

[106] Worrall, J.J. (2012) Fight against online fraudsters intensifies, retrieved from http://www.irishtimes.com/newspaper/finance/2012/0510/1224315837448.html on the 10th of May 2012.

[107] Xu, L., Chow, M-Y. (2006) A classification approach for power distribution systems fault cause identification, IEEE Transactions on Power Systems, vol.21, pp.5360.

[108] Xu, L., Chow, M-Y., Timmis, J. and Taylor, L.S. (2007) Power distribution Outage Cause Identification With imbalanced Data Using Artificial Immune Recognition System (AIRS) Algorithm (2007), IEEE Transactions on Power Systems, vol.22, issue.1, pp.198-204.

[109] Yen, S-J., Lee, S-J. (2009) Cluster-based under-sampling approaches for imbalanced data distributions, Expert Systems with Applications, vol.36, pp.5718-5727.

[110] ZhiYong, L., ZhiFeng, H., XiaoWei, Y., XiaoLan, L. (2009) Several SVM Ensemble Methods Integrated with Under-Sampling for Imbalanced Data Learning, Advanced Data Mining and Applications, vol.5678, pp.536-544.

[111] Zhou, Z., Liu, X. (2006) Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Transactions on Knowledge and Data Engineering vol.18, no.1, pp.6377.

[112] Zhou, Z., and Liu, X. (2010). On Multi-Class Cost-Sensitive Learning. Computational Intelligence, vol.26, no.3, pp.232-257.

Appendix A

The car insurance dataset R code to apply the SMOTE algorithm

Dataset ¡- + read.table(”C:/Users/Peter/Desktop/semester 3/final thesis/Datasets/C+ header=TRUE, sep=”,”, na.strings=”NA”, dec=”.”, strip.white=TRUE)

newData1¡-SMOTE(FraudFound ~.,Dataset,k=5,perc.under=1570.63922) fraudonly ¡- subset(newData1, subset=FraudFound!=”No”) okonly¡- subset(Dataset, subset=FraudFound==”No”) smotedfraudphua¡-rbind(fraudonly,okor

write.table(smotedfraudphua, + ”C:/Users/Peter/Desktop/semester 3/final thesis/Datasets/smotedfraudphua.txt”, + sep=”,”, col.names=TRUE, row.names=TRUE, quote=TRUE, na=”NA”)

Appendix B

The Torga Dataset R code to apply the SMOTE algorithm

Dataset ¡- read.table(”C:/Users/Peter/Desktop/semester 3/final thesis/Datasets/torgo datasets sales fraudulent/saleslabelled.txt”, header=TRUE, sep=”,”, na.strings=”NA”, dec=”.”, strip.white=TRUE)

newData1¡-SMOTE(Insp ˜.,Dataset,k=5,perc.over=340) summary(newData1) fraudonly ¡- subset(newData4, subset=Insp!=”ok”) summary(fraudonly) summary(Dataset) okonly¡- subset(Dataset, subset=Insp!=”fraud”) summary(okonly) smotedfraud¡-rbind(fraudonly,okonly) summary(smotedfraud) write.table(smotedfraud, ”C:/Users/Peter/Desktop/semester 3/final thesis/Datasets/torgo datasets sales fraudulent/saleslabelled.txt”, sep=”,”, col.names=TRUE, row.names=TRUE, quote=TRUE, na=””)

Appendix C

The Thyroid dataset R code to apply the SMOTE algorithm

goitresmote ¡- SMOTE(thyrprob ~ .,th12,k=5,perc.over = 700) summary(goitresmote) goitre ¡- subset(goitresmote, subset=(thyrprob==”goitre.”)) th12¡-rbind(goitre,th12) toxicsmote ¡- SMOTE(thyrprob ~ .,th12,k=5,perc.over = 600) toxic¡- subset(toxicsmote, subset=(thyrprob==”T3 toxic.”)) th12¡-rbind(toxic,th12) summary(toxic)
summary(th12) summary(xwerty) write.table(th12, ”C:/Users/Peter/Desktop/semester 3/final thesis/Datasets/thyroid/smoted.csv”,
sep=”,”, col.names=TRUE, row.names=TRUE, quote=TRUE, na=”NA”)
write.table(th12, ”C:/Users/Peter/Documents/th12.txt”, sep=”,”, col.names=TRUE, row.names=TRUE, quote=TRUE, na=”NA”) write.table(th12, ”C:/Users/Peter/Desktop/semester 3/final thesis/Datasets/thyroid/th12.txt”, sep=”,”, col.names=TRUE, row.names=FALSE, quote=TRUE, na=”NA”)