2018

Classification Using Association Rules

Colin Kane
Technological University Dublin

Classification Using Association Rules

Colin Kane

MSc. in Computing (Data Analytics) March 2018

Declaration of Authorship

I, Colin Kane certify that this dissertation which I now submit for examination for the award of MSc in Computing (Data Analytics), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Dublin Institute of Technology and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institutes guidelines for ethics in research.

Signed: _____

Date: 03 March 2018

Abstract

This research investigates the use of an unsupervised learning technique, association rules, to make class predictions. The use of association rules to make class predictions is a growing area of focus within data mining research. The research to date has focused predominately on balanced datasets or synthetized imbalanced datasets. There have been concerns raised that the algorithms using association rules to make classifications do not perform well on imbalanced datasets.

This research comprehensively evaluates the accuracy of a number of association rule classifiers in predicting home loan sales in an Irish retail banking context. The experiments designed test three associative classifier algorithms CBA, CMAR and SPARCCC against two benchmark algorithms conditional inference trees and random forests on a naturally imbalanced dataset.

The experiments implemented and evaluated show that the benchmark tree based algorithms conditional inference trees and random forests outperform the associative classifier models across a range of balanced accuracy measures. This research contributes to the growing body of research in extending association rules to make class predictions.

Key words: association rule, associative classifiers, Apriori, predictive analytics, KDD, data mining, unsupervised learning

Acknowledgments

Thanks to my wife, Sarah, for giving me so much support throughout the full MSc course. She has kept my spirits up at all time.

Sincere thanks to my Supervisor, Brian Leahy, for the help and encouragement he provided to me in this project.

Thanks to my father, Martin, for helping with the proof reading and general support throughout the course.

Contents

List of Figures

List of Tables

Abbreviations

ABT	Analytics Base Table
AUC	Area Under the Curve
BOI	Bank of Ireland
CAR	Classification Association Rules
CCR	Class Correlation Ratio
CMAR	Classification Based on Multiple Association Rules
CBA	Classification Based Association Rules
CRM	Customer Relationship Management
CRISP-DM	Cross Industry Process for Data Mining
CRM	Customer Relationship Management
CPU	Computer Processing Unit
EDW	Enterprise Data Warehouse
FN	False Negative
FP	False Positive
GDPR	General Data Protection Regulation
GNU	GNU's not Unix
I/O	Input / Output
JSON	JavaScript Object Notation
LOOCV	Leave-one-out cross-validation
KDD	Knowledge Discovery in Databases
PER	Pessimistic Error Rate
R	The R Project for Statistical Computing
ROC	Receiver Operating Characteristic
SEMMA	Sample, Explore, Modify, Model and Assess
SMOTE	Synthetic Minority Oversampling Technique
SPARCCC	Significant, Positively Associated and Relatively Class Correlated Classification
SQL	Structured Query Language
TN	True Negative
TP	True Positive
SAS	Statistical Analysis System
WEKA	Waikato Environment for Knowledge Analysis

1. INTRODUCTION

1.1 Overview of Research Area

Customer expectations of their retail banking experiences are growing. As customers receive a greater level of personalised customer experience across many of their daily brand interactions from companies such as Starbucks, Netflix, Amazon, and Spotify, they increasingly expect this same level of personalised service from retail banks. Therefore, it is becoming increasingly important for retail banks to become customer centric and offer personalised customer experiences[1].

To meet these growing customer expectations, banks are leveraging their data and the growing global data footprint to better understand existing customers and new customer prospects. Banks are using the vast amounts of data they have available to develop deep understanding of their customers and build advanced analytical models to predict an individual's future needs and behaviours. With deep customer understanding and more advanced models to predict consumer behaviour banks can interact with customers in a more personalised way, improve the accuracy of marketing campaigns and offer personalised loyalty programmes to retain customers.

For example, a bank may develop an analytical model that identifies which customers are likely to leave the bank and switch to a competitor (Xie, Li, Ngai & Ying, 2008). The bank can then use the outputs of this model to offer discounts to high value customers to prevent them switching to another bank. Banks also build complex models to predict which product or service the customer is likely to require next. These models power tailored communications with customers across all of the bank's channels whether that is marketing, in branch or in the contact centres. The objective is to truly understand each individual customer and offer a personalised customer experience to retain each customer and grow the banking relationship.

[1] https://www.technative.io/new-banking-study-highlights-expectations-of-todays-customers/

In order to build these analytical models, banks are using advanced statistical analysis and machine learning algorithms. The analytics process typically involves collecting and aggregating data about customers, transforming the data so it can be used for analytics and using that data to build predictive models that determine an individual's propensity to carry out some behaviour. With growing customer expectations and new data regulations, additional pressure is being placed on these analytics departments within banks to improve the accuracy of these models. Banks are investigating new models and approaches to increase the accuracy of their models enabling this personalised experience.

The new data regulation GDPR ("General Data Protection Regulation"), which comes in to effect on 25th May 2018[2], means that customers now have to clearly demonstrate their consent and willingness for organisations to collect, store and analyse their data. Customers will only do that if they feel they are getting value for handing over their personal data to retail banks. In order to convince customers to allow a particular organisation to analyse their data customers will need to feel they are getting considerable value in exchange for this data processing. If they don't feel they are getting value then they are unlikely to 'opt in' to this type of data processing. One way to provide value is to use data to truly understand each customer and give each customer a personalised experience with tailored products and propositions. If customers believe the organisation is using their data to help them or provide personalised offers and service then this may entice customers to provide consent to process their data for analytics. This is another area where accurate advanced analytical models play a key role.

1.2 Background

Banks are using data mining techniques to predict when a customer is likely to be interested in a particular product and then contact or advertise to the customer with a relevant marketing message (Kamakura, Wedel, De Rosa, & Mazzon, 2003). To make these predictions for individual customers, banks are using supervised learning classification models such as decision trees (Quinlan, 1986), logistic regression (McCullagh, 1984), and random forests (Breiman, 2001). A typical example is the construction of a model to predict which customers

[2] http://ec.europa.eu/justice/data-protection/

are likely to take out a loan for a house purchase in the next twelve months using data such as demographics, current and previous product holdings, transactional data and savings patterns.

There may also be an opportunity to use unsupervised learning models to make these predictions. Unsupervised learning is a machine learning approach to find patterns and trends in data where the input data does not include labelled responses. The most common unsupervised learning methods include clustering (Jain, Murty, & Flynn, 1999), anomaly detection (Chandola, Banerjee, & Kumar, 2009), and association rules (Agrawal, Imieliński, & Swami, 1993).

Association rules are used to identify interesting rules in a dataset. The classic application of Association Rule algorithms is the identification of rules within retail store transactions, also known as Market Basket Analysis. The general concept is to identify rules, such as a customer who buys product A also buys product B. Classification using association rules is an extension whereby association rules are used to make class predictions. Classification Association Rules (CARs) is an alternative prediction approach to supervised learning models to make class predictions.

The motivation behind this research is to test the accuracy of classifications using association rules with traditional classification methods. The scope involves testing the predictions made by association rules on real-world retail banking sales data. In this research, the focus will be on the prediction of loans for home purchase. The research will aim to address the problem as to whether association rules can make better predictions for product sales compared to traditional classification algorithms. If the research proves successful it will support the consideration of association rule learning for classification problems in the future.

1.3 Research Problem

The key research problem of this dissertation is to assess whether association rule algorithms can produce statistically better classifications of mortgage sales than alternative classification algorithms in an Irish retail banking context.

1.4 Research Objectives

The primary goal of this research is to assess the predictive capability of association rule learning in predicting mortgage sales in an Irish retail bank.

The three primary objectives of this research are as follows:

Implement the **Classification Based on Association Rules** ('CBA') (Liu, Hsu & Ma, 1998) algorithm to predict mortgage sales and compare its performance to the performance of the conditional inference trees, Classification Based on Multiple Association Rules ('CMAR') algorithm, Significant, Positively Associated and Relatively Class Correlated Classification ('SPARCCC') algorithm and random forests. The results will be evaluated using a comprehensive assessment across multiple model performance metrics.

Implement the **Classification Based on Multiple Class-Association Rules** ('CMAR') (Li, Han, & Pei, 2001) association rule algorithm to predict mortgage sales and compare its performance to the performance of the conditional inference trees, random forests, SPARCCC and the CBA algorithm. The results will be evaluated using a comprehensive assessment across multiple model performance metrics.

Implement the **Significant, Positively Associated and Relatively Class Correlated** ('SPARCCC') (Verhein and Chawla, 2007) association rule algorithm to predict mortgage sales and compare its performance to the performance of the conditional inference trees, CMAR, random forests and the CBA algorithm. The results will be evaluated using a comprehensive assessment across multiple model performance metrics.

These objectives will be achieved by the completion of the following steps:

- Researching the relevant state of the art literature and industry best practices for association rule learning and classification using association rules.

- Acquire, prepare and transform customer and sales data for analysis.

- Generate an Analytics Base Table (‘ABT’) for model development and testing.

- Design experiments to test the three hypotheses.

- Train benchmark prediction models to compare and evaluate the associative classifier models.

- Design and build the classification using association rule models.

- Critically evaluate the results from the association rule classification models and compare the results with the benchmark classification models to evaluate if classification using association rules should be considered when building predictive models in retail banking.

- Identify areas for future research to be undertaken in this area.

1.5 Research Methodologies

The research method that will be employed in this dissertation is an empirical evaluation of classification using association rules. This research will compare the performance of algorithms using association rules to make class predictions to a number of benchmark classification approaches. For project direction and idea generation, the research will review the state-of-the-art experiments completed in the field of classification using association rules.

To perform the experiment numerous disparate datasets will be acquired, cleansed, transformed and integrated together to develop an ABT. The datasets will include, socio-demographic data (age, sex, location), transactional spend (debit, credit, credit card, direct debits) and current and previous product holdings. This will be supplemented with certain semi-structured web behavioural feature and unstructured textual features to complete the ABT. The ABT will form the basis for the development of numerous prediction models.

As part of the experiment, benchmark prediction models will be trained using traditional classification models. Prediction models will also be built using association rules (CBA, CMAR and SPARCCC). The prediction models using association rules will be compared and assessed against the benchmark classification models. Should the predictions from association rules perform better than the benchmark classification model this research will provide evidence that association rules should be considered for future classification problems.

1.6 Scope and Limitations

The scope of this project is to implement and evaluate three classification models using association rules, CBA, CMAR and SPARCCC on real-world Irish retail banking data. The classification results of these three models will be assessed against two benchmark classification models to provide evidence as to whether associated rules should be considered in classification problems in the future.

The data to be included in this project will be retrieved from Bank of Ireland ('The Bank') CRM databases and product sales databases. These multiple datasets will be acquired, cleansed and aggregated to build the ABT for the experiments.

To assess the capability of association rules to make accurate classification predictions this research will also include the development of a number of benchmark classification algorithms using traditional classification models such as decision trees and random forests. If the performance of the association rules models is better than the traditional classification models then CBA, CMAR and SPARCCC should be considered for inclusion in future customer behaviour prediction problems.

The real-world dataset for use in these experiments is a naturally imbalanced dataset. This research is limited to providing analysis and results on imbalanced data. This research will not provide a comparison of the performance of association rule classifiers on real-world balanced datasets. This is a potential area for future research.

1.7 Document Outline

The remaining chapters of this thesis are organised as follows:

- Chapter 2 documents and evaluates the current state of the art in the field of association rules, the use of association rules for classification, and the general field of data mining which includes predictive modelling, performance measurement and handling imbalanced datasets. Techniques and methods for feature transformation are also discussed here.

- Chapter 3 presents the design and research methodology for the project. This chapter explains the data used for the experiment and the robust experiment designed to test the accuracy of classifications using association rules and compare the results against benchmark data mining models. The models being employed will be explained here together with the approach to measure the results of the experiments.

- Chapter 4 presents the implementation of the experiments carried out as part of this research. In this chapter, the experiment results will be evaluated and critically assessed. Conclusions and observations will be made where it is possible to do so.

- Chapter 5 presents the results of the experiment in the context of the wider research in the field of classification using association rules. This chapter presents where this research confirms or challenges previous research in this field or presents new evidence.

- Chapter 6 concludes the paper by presenting the contributions made to the problem of classification using association rules. It concludes by discussing limitations to the research, areas for future research that could be considered and some alternative experiments worth implementing.

2 LITERATURE REVIEW AND RELATED WORK

2.1 Introduction

Chapter 2 reviews the research literature in the field of knowledge discovery and data mining in particular association rule learning a form of unsupervised learning and classification using association rules. This Chapter analyses and critiques the state of the art algorithms from the existing body of research in extending association rule learning algorithms to make class predictions. The purpose of this research and the experiments outlined below in Chapter 3 is to extend the existing body of research in this area. Chapter 2 is divided into seven further sections.

In Section 2.2 the state of the art frameworks for knowledge discovery in databases and data mining are presented and critiqued. Within the field of data mining, there are three main forms of algorithmic learning, supervised, unsupervised and reinforcement learning. This research is focused on association rule learning algorithms which is a form of unsupervised learning.

Section 2.3 discusses the background to association rule learning, prior use cases and outlines some of the complexities of this data mining approach. The state of the art research on association rule algorithms is presented and the advantages and disadvantages of each algorithm are identified and discussed. These algorithms are the foundational layer for classification using association rules. These algorithms identify high quality rules which are then used to make class predictions in the associative classifier models presented in Sections 2.4 and 2.5.

Section 2.4 of the literature review outlines the process for extending the association rule algorithms presented in Section 2.3 to make class predictions. The key steps to adapt association rules algorithms to make class predictions are discussed. The state of the art models generally use three steps to extend association rule algorithms to make class predictions, generating interesting rules using an association rule learning algorithm, pruning the rules and using the rules to make classifications. The seminal algorithms CBA and CMAR are contrasted across these three major steps.

Section 2.5 outlines the impact of imbalanced datasets on classification using association rules. Real-world datasets are often imbalanced where the target being predicted is dominated by one class. This is often the case in retail banking product prediction cases similar to this research. In Verhein and Chawla (2007), the authors state that algorithms such as CBA and CMAR built under the *support-confidence* framework do not perform well on imbalanced datasets. Approaches to dealing with imbalanced datasets are discussed here as well as the SPARCCC classifier model which was built particularly to handle this imbalanced dataset problem. This section also describes in detail the existing research on handling imbalanced datasets including over sampling and undersampling.

Section 2.6 presents the research on certain data transformation techniques such as data discretisation that are required for associative classifiers. Various unsupervised and supervised data discretisation techniques to convert continuous attributes into discrete attributes are reviewed and evaluated.

Given the importance of testing and validation of the models, research is presented in Section 2.7 on the techniques for model validation to avoid overfitting and underfitting. Methods for model validation such as cross validation are presented and evaluated.

Section 2.8 of the review outlines the metrics that will be applied in this research to compare the predictive performance of one model to another. Where the underlying dataset is imbalanced the traditional accuracy measure may need to be avoided as it can be biased towards the majority class. More balanced metrics for imbalanced datasets such as F1-score, balanced accuracy and AUC are presented.

2.2 Knowledge Discovery in Databases (KDD) and Data Mining

KDD is the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). The aim of the KDD process is to garner insights from large datasets. Figure 2.1 provides an overview of the steps involved in gathering information and knowledge from sources of data (Fayyad et al., 1996). The KDD process consists of several stages, selection, pre-processing, transformation,

data mining and interpretation/evaluation. Association rule mining is one data mining application to extract patterns in data.

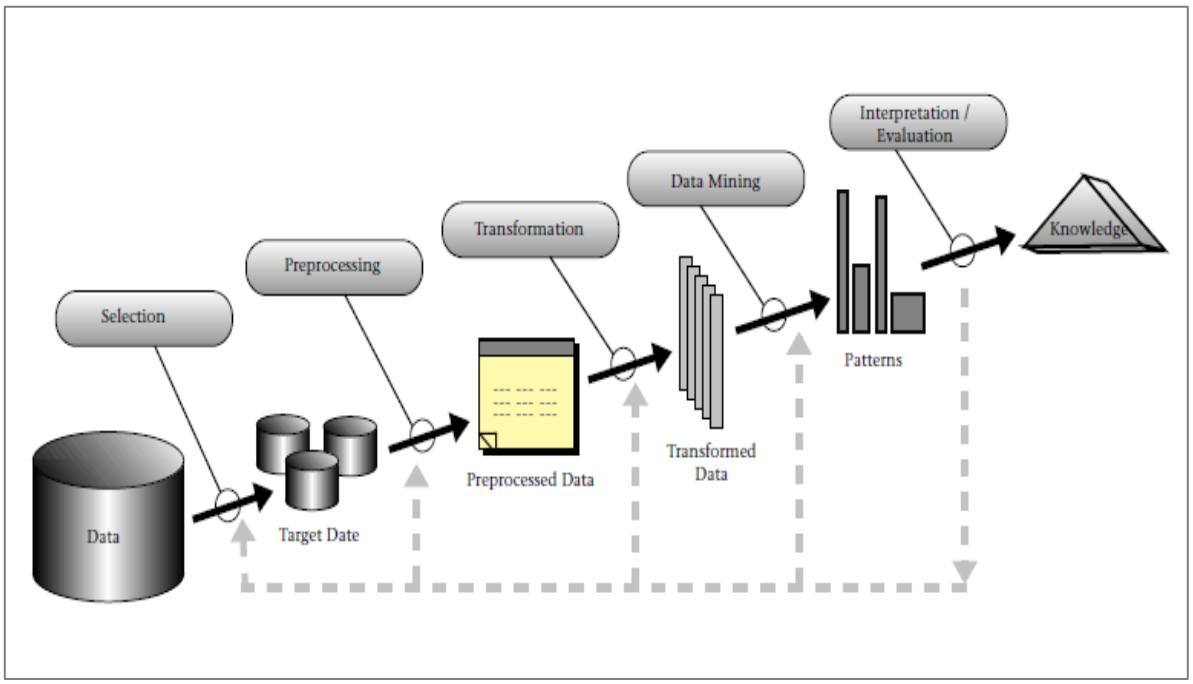


Figure 2.1: Overview of the KDD Process
(Source: Fayad et al., 1996)

Data Mining forms one of the steps in the KDD process. The goal of the data mining step is to identify patterns which can then be interpreted and allow for more informed decisions to be taken. The authors state that "*Data mining is a step in the KDD process that consists of applying data analysis and discovery algorithms that, under acceptable computational efficiency limitations, produce a particular enumeration of patterns (or models) over the data"*.

There are several frameworks that outline a process to deliver a successful data mining project summarised in table 2.1. The two most well-known frameworks are CRISP-DM and SEMMA. The *Cross Industry Process for Data Mining* or CRISP-DM is a commonly used process to complete a data mining project. The CRISP-DM (Shearer, 2000) process presents six phases of the data mining process, business understanding, data understanding, data preparation, modelling, evaluation and deployment. The arrows between different process

steps in Figure 2.2 highlight the iterative nature of a data mining project where insights and results from one step can inform previous or future steps in the process.

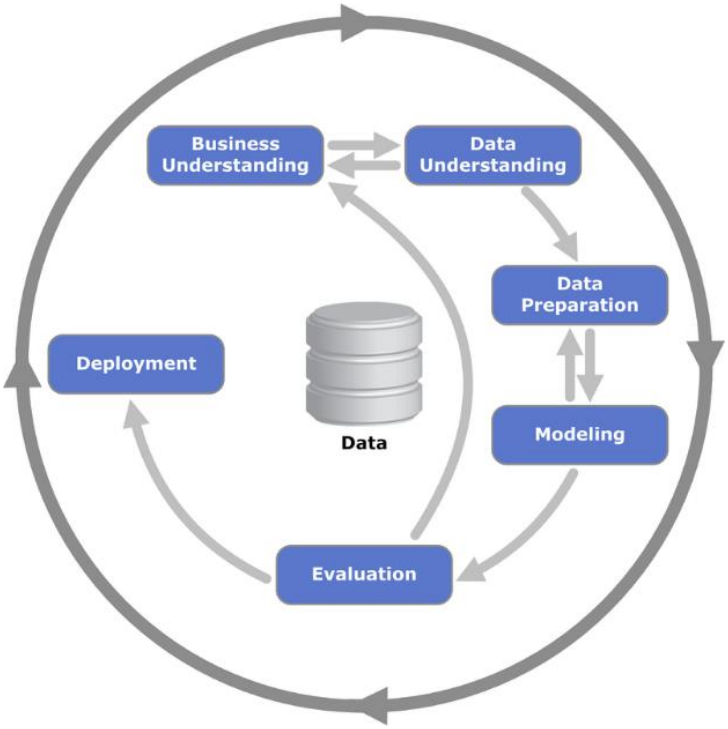


Figure 2.2: CRISP-DM Data Mining Process Model
(Source: Shearer, 2000)

The SEMMA process consisting of sample, explore, modify, model and assess is a list of sequential steps developed by the SAS Institute. It is often noted that the SEMMA process lacks business focus (Azevedo & Santos, 2008) in its process, unlike the CRISP-DM process which includes the business understanding phase as the first phase. Although gathering the domain and problem knowledge is not specifically identified as a phase in SEMMA, it is argued that it is not feasible to start a project without this understanding and therefore it is assumed that this forms part of the sample phase in SEMMA. Azevedo & Santos (2008, p. 5) state "*we can integrate the development of an understanding of the application domain, the relevant prior knowledge and the goals of the end-user, on the Sample stage of SEMMA, because the data cannot be sampled unless there exists a true understanding of all the presented aspects*". Table 2.1 below neatly summarises the comparison of the three methodologies discussed above.

KDD	SEMMA	CRISP-DM
Pre-KDD	---------	Business Understanding
Selection	Sample	Data Understanding
Pre-processing	Explore	
Transformation	Modify	Data preparation
Data mining	Model	Modelling
Interpretation/Evaluation	Assessment	Evaluation
Post KDD	---------	Deployment

Table 2.1: Comparison of KDD, SEMMA and CRISP-DM

Within the Data Mining step of the knowledge discovery process, there are three primary forms of learning algorithms namely supervised, unsupervised and reinforcement learning algorithms.

In supervised learning, the data provided to the model has known class labels which are the corresponding correct outcomes. For supervised learning tasks, the data is usually represented in a table similar to Table 2.2. Supervised learning algorithms create a function that models the data using historic data instances and the function is then applied to predict the outcome of previously unseen data. The function created on historic data is used to score new previously unseen data instances. Real-world examples of supervised learning include spam detection (Androutsopoulos, Koutsias, Chandrinos, Paliouras, & Spyropoulos, 2000), default prediction models in financial services (Atiya, 2001), cancer prediction in health services (Shipp, Ross, Tamayo, Weng, Kutok, Aguiar, & Ray, 2002) and voice recognition (Hinton, Deng, Yu, Dahl, Mohamed, Jaitly, & Kingsbury, 2012). The data mining algorithms used to perform supervised learning tasks include logistic regression (McCullagh, 1984), decision trees (Quinlan, 1986), random forests (Breiman, 2001), support vector machines (Cortes & Vapnik, 1995) and neural networks.

Standard Data Format					
Instance	Feature 1	Feature 2	…	Feature n	Class
1	XXX	X			Sale
2	XXX	X			No Sale
3	XXX	X			Sale
…					…

Table 2.2: Sample Dataset for Supervised Learning

Another form of data mining is reinforcement learning (Barto & Sutton, 1997). Reinforcement learning is learning what to do and mapping situations to necessary actions. In reinforcement learning, the learner is not told what to do but instead must learn what action yields the maximum reward. An example of reinforcement learning is teaching an agent how to play computer games such as Super Mario or Pac Man. An example of a reinforcement learning algorithm is Q-learning (Watkins, 1992). In Q-learning, the goal is to reach the state with the highest reward, so that if the learner arrives at the goal, it will remain there indefinitely. In reinforcement learning this type of goal is called an absorbing goal.

Unsupervised learning is applied where data instances are unlabelled. The dataset is typically similar to Table 2.2 above, however, the class label for prediction is not available. By applying these unsupervised algorithms, researchers hope to discover unknown, but useful, classes of items (Jain et al., 1999). Some of the most well researched unsupervised learning algorithms include clustering, anomaly detection and association rule learning.

Considerable research has been carried out on supervised learning techniques to predict classes including models such as decision trees and neural network approaches. More recent studies (Liu et al., 1998; Li et al., 2001) propose the use of unsupervised association rules for classification purposes by using a set of high-quality association rules to make the class predictions.

2.3 Association Rule Learning

Association rule learning, a form of dependency modelling[3], examines the dataset for relationships between variables or items. The classical application of Association Rule algorithms is within the context of retail store shopping transactions and the items within those transactions (Agrawal et al., 1993). The analysis of association rules in retail store databases is more commonly known as Market Basket Analysis. The general concept of association rule learning is to identify rules such as a customer who buys product A also buys product B with an identifiable confidence level. Another area where association rules have been employed is in medical research to identify high-risk patients (Obenshain, 2004) and the early identification of infection (Brossette, Sprague, Hardin, Waites, Jones, & Moser, 1998).

When applying association rule algorithms, the objective is completeness, the algorithm is required to find all interesting rules in the dataset. The difficulty with association rule mining is the size and complexity of the problem. The number of possible rules in the dataset increases exponentially with the number of items. The algorithms developed for association rule mining attempt to reduce this level of complexity and provide fast results from the models developed.

The idea of applying Association Rule Mining to Market Basket Analysis was introduced by (Agrawal et al., 1993). Formally, the problem of association rule mining is defined as: Let $I = \{i1, i2, \dots in\}$ be a set of n distinct literals called items. Let $D = \{t1, t2, \dots tm\}$ be a set of transactions in the database. Each transaction T is unique and contains a number of items from I.

An association rule is a conditional implication among itemsets, $X => Y$ where X, Y are items. In order to identify interesting rules in the dataset, there are two key metrics in measuring association rule mining results, the support and the confidence of the rule.

The support *supp(X)* is defined as the proportion of transactions in the dataset which contain the itemset X and reflects its statistical significance. In simple terms, the number of transactions which contain X in the transaction is divided by the total number of transactions.

[3] http://www2.cs.uregina.ca/~dbd/cs831/notes/kdd/2_tasks.html

The confidence of any rules identified is measured as the percentage of transactions containing Y which also contain X, divided by the number of transactions which contain X within the whole dataset. This identifies how often the identified combination occurs together. Confidence is the measure to monitor the individual strength of the association rules identified.

The goal of association rule mining is to find all association rules which exceed some user-defined minimum levels for both support and confidence.

To identify the association rules within a dataset using an association rule algorithm there are typically two steps.

1. The first step is to identify all combinations of itemsets that meet the user identified minimum support (minsupp) thresholds set. These itemsets are said to be large or frequent itemsets and those that do not meet the support level are said to be small or infrequent itemsets.

2. The second step is to measure the confidence of each rule and compare against the minimum confidence level chosen (minconf).

Once the rules that meet the minimum support threshold are identified the second step is rather straightforward (Agrawal et al., 1993). The algorithms for association rule learning focus predominately on the first sub-problem above and try to reduce the computationally expensive task of identifying all rules which are above the user-defined support level.

As the number of items increases, there is an exponentially growing number of itemsets which need to be assessed. For example, if |I| = m, the number of possible distinct itemsets is 2^m, which forms a lattice of subsets over I. Typically, only a very small number of the itemsets in this exponentially large subset will meet the minimum support levels set. Figure 2.3 (Hipp, Güntzer, & Nakhaeizadeh, 2000) provides an illustration of the itemsets that need to be assessed with 4 items.

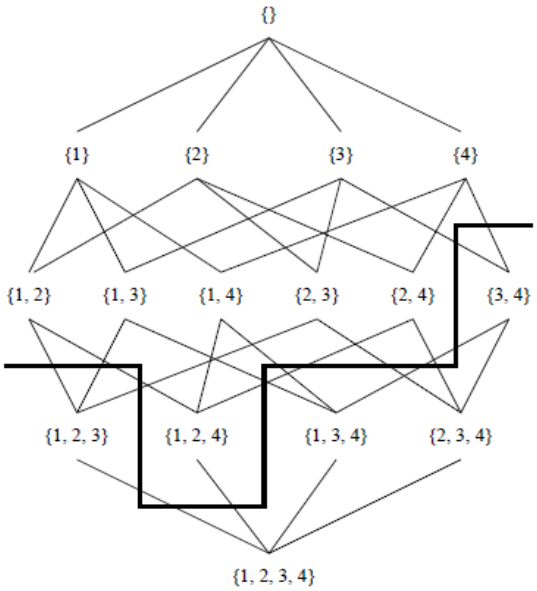


Figure 2.3: Lattice for I = {1,2,3,4}

(Source: Hipp et al., 2000)

The main problem in association rule mining is identifying itemsets which meet the user-defined minimum support level. When using these algorithms in practice with a large number of items, assessing each itemset is not possible as the size of the search space is too large.

To reduce the size of the search space the algorithms rely on the downward closure property (Agrawal & Srikant, 1994) which prevents the algorithm from counting itemsets which will not be frequent at the end. Employing this property significantly reduces the number of itemsets to be assessed.

There are four main types of association rule algorithms, each of which employs a different strategy for identifying itemsets that meet the minimum support level defined. The variances between the models are whether the algorithm employs breath first search or depth-first search and secondly whether the algorithm uses candidate generation or set intersecting to determine the support values of candidates. Within set intersections the algorithms use a tidlist. A TID is a unique transaction identifier for all transactions in the databases. For each item in *I* the relevant tidlist is a list of all transaction ids for transactions which contain the item. The use of tidlists is applied in the Partition and EClat algorithms.

2.3.1 AIS Algorithm

In Agrawal et al. (1993), the authors first introduced the idea of mining large datasets for association rules. The authors presented the AIS algorithm which generated new itemsets by extending out large itemsets found in the previous database pass with other items in the transactions, a step known as candidate generation. This resulted in a large number of itemsets being counted which would ultimately turn out not to meet the minimum support levels set. Houtsma and Swami (1995) subsequently presented an algorithm called SETM which introduced the idea of trying to solve the association rule problem using a relational database.

2.3.2 Apriori Algorithm

Agrawal et al. (1994), presented the Apriori and AprioriTID algorithms. The Apriori Algorithm uses a breath-first search and builds on previous algorithms through the application of the downward closure property to reduce the number of itemsets which need to be counted and therefore run more efficiently. In the paper the authors present the following lemma, "*The basic institution is that any subset of a largest itemset must be large. Therefore, the candidate itemsets having k items can be generated by joining large itemsets having k – 1 items, and deleting those that contain any subset that is not large. This procedure results in the generation of a much smaller number of candidate itemsets"* (Agrawal et al., 1994, p.4).

For example, if it is found the itemset {1,2,3} is small, then none of the itemsets which are extensions of {1,2,3} such as {1,2,3,4} or {1,2,3,5,7} need to be tested for minimum support. In practice, the Apriori algorithm prunes particular sets as it makes passes over the database and does not count any itemset in the next pass where a subset of the itemset did not meet the support level required in a previous pass. One of the criticisms of the Apriori algorithm is that the algorithm requires multiple passes over the database which can be computationally expensive.

The AprioriTid (Agrawal et al., 1994) aims to address the computationally expensive nature of the Apriori algorithm. AprioriTid encodes all the large itemsets in a transaction after the first pass to prevent having to pass over the database itself in subsequent passes. In subsequent passes, the level of transactions can be much smaller than the database, however,

for initial passes, the encoding of the transactions may be larger than the actual database. To overcome this issue, the authors propose a hybrid of Apriori and AprioriTid named Apriori Hybrid, which uses Apriori for earlier passes and AprioriTid for later passes.

The authors compared the performance of the two new algorithms with the previous algorithms AIS and SETM. The results showed that the performance gap, in favour of the two new algorithms, increased as the size of the problem increased, ranging from a factor of three for small problems to more than an order of magnitude for large problems.

The CBA algorithm for performing classification using association rules extends the Apriori algorithm to make class predictions.

2.3.3 Partition Algorithm

Savasere, Omiecinski, and Navathe (1995) present an alternative method for association rule mining known as the partition algorithm. The objective of the partition algorithm is to reduce the number of required passes over the database to identify large itemsets. Reducing the number of passes the algorithm needs to make over the database reduces the run time and reduces the impact on the underlying hardware system (Savasere et al., 1995).

The partition algorithm requires only two passes over the database. In the first pass of the database, the algorithm splits the database into a number of non-overlapping smaller partitions and then identifies all large itemsets within each of the smaller sets. The model ensures that the partition sizes are chosen to ensure there are no difficulties in relation to the main memory. In the second pass, these large itemsets are joined together, their actual count and support is calculated and those that meet the target support level are identified. The second step ensures that the itemsets which are found to be large in each partition i.e. locally supported are also supported globally on the full database.

Similar to the Apriori Algorithm the Partition Algorithm employs the downward closure property and prunes itemsets which are found not to be large from being considered for counting support.

In testing the model against previous algorithms, the authors used the same synthetic data as in Agrawal et al. (1994). The author's tests showed that the Partition Model outperformed the Apriori model by up to a factor of seven while also reducing the levels of CPU usage and I/O.

2.3.4 Frequent Pattern (FP) Growth Algorithm

Han, Pei, and Yin (2000) developed a new approach to identifying association rules moving away from an Apriori-like approach. Apriori algorithms use a generate and test approach which involves generating itemsets and then testing if they are frequent. Identifying frequent itemsets is the costliest element of Apriori-like algorithms. The authors note that applying the downward closure property (Agrawal et al., 1994) achieves good performance gain on previous algorithms but is still very costly in terms of performance in situations where there are a large number of frequent itemsets or the minimum support thresholds are low.

The FP Growth model proposes an alternative approach to identifying frequent itemsets which does not rely on candidate generation. The FP Growth model works in two steps:

1. The model converts the transactions in the database into a more compact data structure, a Frequent Pattern Tree (FP Tree) which is built using two passes of the database.

2. In the second step, the model then uses the FP tree constructed rather than the database to find frequent patterns.

The FP Tree is constructed using two passes over the database; in the first pass the support for each item in the database is calculated and infrequent items are pruned. Frequent items are sorted in a fixed order to ensure efficiency. Then in the second pass the FP tree is constructed and all transactions are mapped to a path on the tree and the counting is completed. As certain transactions may have items in common their paths may overlap, this is taken into account when building the FP Tree and therefore reduces the size of the data structure. Figure 2.4 (Tan, 2006) below outlines the process of creating an FP-tree for 10 transactions (TIDs). Where subsequent transactions follow similar paths the nodes of the tree will increase the count by 1.

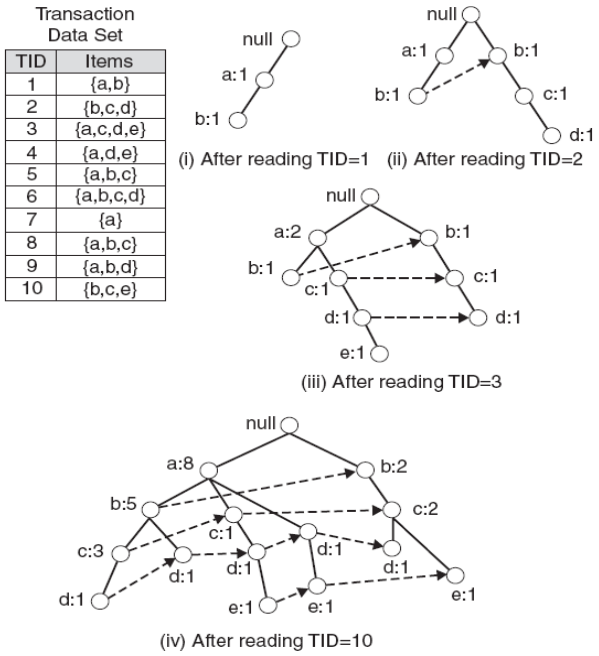


Figure 2.4: FP Tree for 10 Transactions Dataset

(Source: Tan, P. N., 2006)

Once the tree is created to identify frequent patterns *"the search technique employed in mining is a partitioning-based, divide and conquer method rather than an Apriori-like bottom up generation of frequent itemset combinations"* (Han et al., 2000, p.2).

In order to identify the frequent patterns, the authors create prefix path sub-trees for each item set. Each prefix path sub-tree is then processed recursively to extract the frequent itemsets. Based on the prefix path the model creates conditional FP Trees for each itemset.

The authors test the FP growth model against the candidate generation models and find that the model is about an order of magnitude faster than Apriori.

The CMAR algorithm for classification using association rules is an extension of the FP-growth algorithm for association rule learning.

2.3.5 Equivalent Class Transformation Algorithm

Zaki, Parthasarathy, Ogihara and Li (1997) present four new algorithms which only require one pass over the database. The algorithms presented by the authors differ to Apriori-like algorithms in that they traverse the prefix tree in depth-first order compared to breath-first search in the Apriori algorithm. The most important algorithm presented, EClat, relies on tid-lists as described above. Each transaction has a transaction id or tid, a tid-list is a list of all the transactions which contain a particular item. The Eclat model determines the support of any k-itemset by intersecting tid-lists of two of its (K-1) subsets. The authors state this as '*We partition L_k into equivalence classes based on their common K-1 length prefix, given as [a] = [b[k]|a[1:k-1] = b[1:k-1]}*'.

The authors propose that a vertical format for storing the transactional data is more applicable to association rule mining than a horizontal format. Under this method, the model only needs to make one pass of the database. Both Apriori and FP Growth use horizontal data format which starts with the transaction id and the itemsets within the transaction. The vertical format starts with the itemset and lists all transactions which contain that itemset. The authors state that a "*vertical format seems more appropriate for association mining since the support of a candidate k-itemset can be computed by simple tid-list intersections"* (Zaki et al., 1997, p.285). Figure 2.5 shows an example of a tid-list intersection.

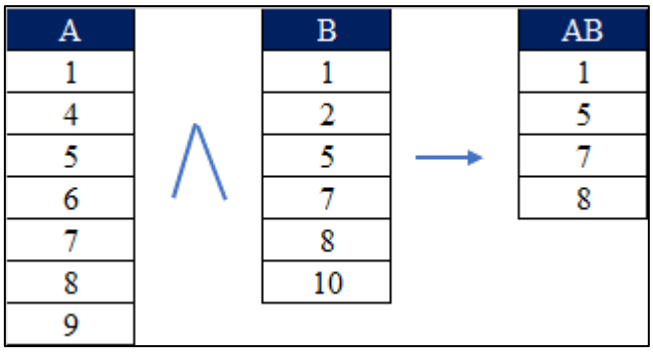


Figure 2.5: Example of tid-list intersection

The authors compare the new algorithms presented against Apriori and Partition (with 10 partitions). The authors state that EClat outperforms Apriori by a factor of 10 and Partition by a factor of 5. The authors also state the new models scale well as the transactions sizes

increase. The advantage of this algorithm is that the depth-first search can result in much faster results, however, the intermediate tid-lists may become too large for memory.

2.3.6 Conclusion Association Rule Learning Algorithms

Zheng, Kohavi and Mason (2001) performed the first evaluation and comparison of association rule learning algorithms on real-world datasets. In this experiment, the authors evaluated five of the state of the art association rule algorithms including Apriori and FP-growth. The authors evaluated performance on three real-world datasets and one artificial dataset using a range of minimum support values to test performance and scalability. For the artificial dataset, every algorithm outperformed Apriori by a significant margin for minimum support values less than 0.10%. FP-growth was one order of magnitude faster than Apriori when the minimum support was set to 0.02%. This evidence is consistent with the results of other previous experiments (Han & Pei, 2000; Zaki, 2000). The performance improvement of FPgrowth over Apriori increases as the minimum support decreases, indicating that FP-growth scales better than Apriori. For all of the real-world datasets, FP-growth is faster than Apriori, but the differences are not as large as on the artificial dataset. The reasoning proposed by Zheng et al. (2001) is that the artificial dataset has different characteristics to the real world datasets.

In Hipp et al. (2000), the authors compare a number of association rule algorithms on efficiency by carrying out several runtime experiments on synthetic data. The authors compare Apriori, DIC a variation of Apriori (Brin, Motwani, Ullman, & Tsur, 1997). Partition and Eclat. The authors state that the results of the experiments indicate that the runtime behaviour of the various algorithms is more similar than expected. Only in certain more extreme cases did the authors evidence varying performance. In Figure 2.6 one of the experiments on a more complex dataset shows Eclat and Partition performing better than Apriori, particularly at low minimum support levels.

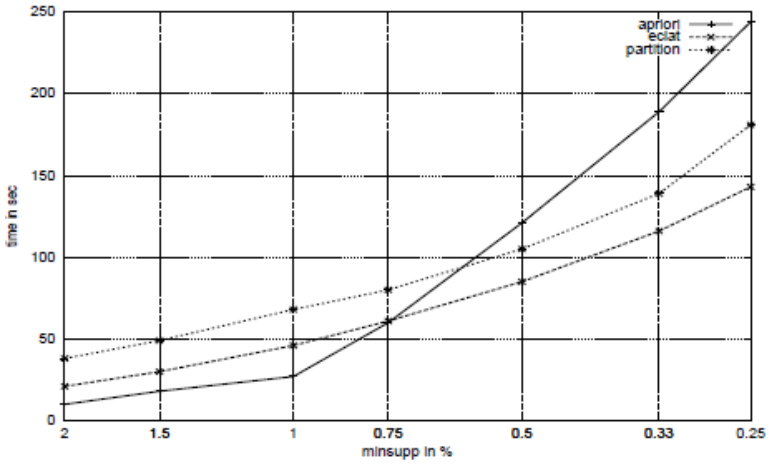


Figure 2.6: Comparison of association rule algorithms across varying support levels (Source: Hipp et al., 2000)

Heaton (2016), compared the performance of Apriori, Eclat and FP-growth across varying artificially created datasets. Two dataset characteristics were evaluated, maximum transaction size and frequent item density and the algorithms were tested under various conditions. The results demonstrate that Eclat and FP-Growth both handle increases in maximum transaction size and frequent itemset density considerably better than the Apriori algorithm, while FP-growth marginally outperformed Eclat. Figure 2.7 below shows the results of the tests for various frequent itemset densities. It shows that all three of the algorithms perform to a similar level up to roughly 70% at which point the performance of Apriori considerably deteriorates.

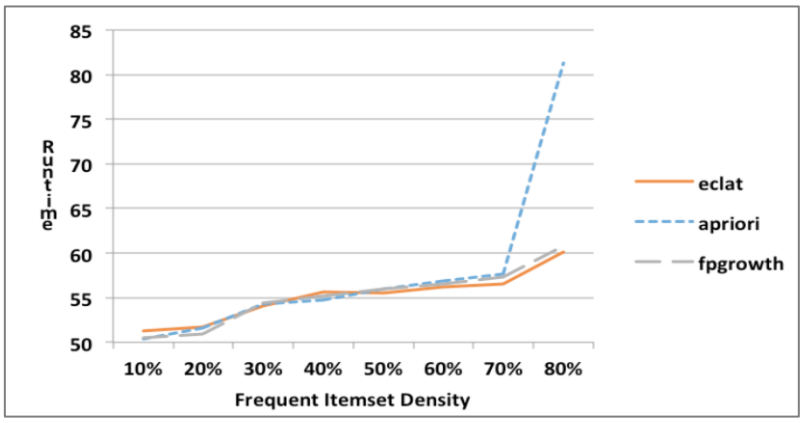
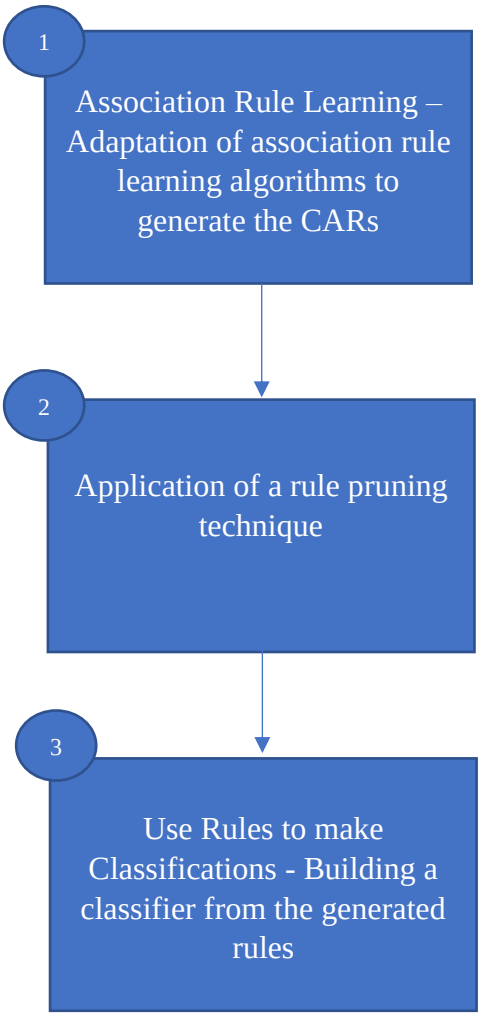


Figure 2.7: Comparison of association rule algorithms across varying itemset densities (Source: Heaton, 2016)

2.4 Extending Association Rules to make predictions

In data mining, making class predictions is typically associated with supervised learning. Supervised learning aims to create a function or set of rules to accurately classify labels and then the rules or function are applied to newly unseen data to make predictions. Association rule mining is an unsupervised learning approach that finds all rules in the database that satisfy some minimum support and minimum confidence constraints, for example Agrawal and Srikant (1994). Liu et al. (1998) propose a framework to combine these two techniques of data mining. The authors propose an integrated framework, called *associative classification*. The framework concentrates on a subset of association rules where the right-hand side of the rule is restricted to the class being predicted. This subset of rules is called class association rules.

The approach to associative classification involves three key steps:



2.4.1 Generating Interesting Rules

Liu et al. (1998) propose a new algorithm CBA with two parts, a rule generator CBA-RG and CBA-CB which uses the rules generated to build classifications. To use association rules for classification the underlying algorithms described in Section 2.3 above need to be adapted. The CBA-RG is an adaptation of the Apriori algorithm described in Section 2.3. Class association rules are a subset of all association rules where the right-hand-side is restricted to a distinct class label. Class association rules or itemsets take the form <condset,y> where y is a class label. The CBA-RG identifies frequent itemsets with high support and high confidence within the subset of rules.

Support is defined as $\frac{rulesupCount}{|D|}$ where *rulesupCount* is the total count of a particular itemset and *D* is the total database. Itemsets that satisfy a set minimum support level are deemed frequent and other remaining itemsets are deemed infrequent.

The confidence of the itemset is defined as $\frac{rulesupCount}{CondsupCount}$ where *CondsupCount* is the total count of the condset within *D*.

For example, if there is a rule {Age: 25-35, Location: Dublin} -> Sale. If the count of the condset {Age: 25-35, Location: Dublin} is 3 and the count of the itemset {Age: 25-35, Location: Dublin} -> Sale is 2 and the total database is 10. The support of the itemset is 2 / 10 or 20% and the confidence of the itemset is 2/3 or 66.67%.

The CBA-RG outputs all CARs that meet the minimum support and minimum confidence levels. For itemsets with the same condset the algorithm chooses the itemset with the highest confidence. The criticism of CBA-RG is that the algorithm outputs only a single high-confidence rule. This may lead to biased classifications that overfit the data. Verhein and Chawla (2007) challenge the ability of CBA to build an accurate classifier on imbalanced datasets.

Li et al. (2001) propose a new algorithm Classification Based on Multiple Association Rules (CMAR) to overcome the restrictions inherent in CBA. The authors propose the use of more rules to support the class prediction problem. The requirement to store many more rules,

however, has an impact on the combinatorial explosive nature of association rules as described in Section 2.3. If |I| = m, the number of possible distinct itemsets is 2^m. In CMAR, the authors propose the use of multiple high-quality rules to make a classification decision rather than restricting the decision to the rule with the highest confidence score as applied in the CBA algorithm.

For example, suppose there is a new database instance {Age: 25-35, Location: Dublin, Job: Accountant} and the Bank wants to determine whether this customer is likely to take out a mortgage for home purchase. The three rules with the highest confidence for this customer are as follows:

- Rule 1, {Location: Dublin} -> No Sale (Support 20%, Confidence 90%)
- Rule 2, {Age: 25-35} -> Sale (Support 30%, Confidence 87%)
- Rule 3, {Job: Accountant} -> Sale (Support 25%, Confidence 85%)

Using the CBA algorithm Rule 1 would be chosen for classification purposes given it is the rule with the highest confidence, however, the other two rules both have higher support and only marginally lower confidence. The objective of CMAR (Li et al., 2001) is to use a number of high-quality rules to make a more balanced decision to classification and increase prediction accuracy over CBA by reducing the levels of overfitting.

To find rules for classification, CMAR adopts a variant of the FP-growth method explained in Section 2.3. The FP-growth model is faster than the Apriori model particularly in situations where there are a large number of frequent itemsets or the minimum support thresholds are low (Hipp et al., 2000). One of the major advantages of the CMAR adaptation of FP-growth is that it is capable of identifying frequent itemsets and generating rules in one pass while traditional algorithms like Apriori and Apriori extensions such as CBA require two passes. In Apriori, first all items that pass the minimum support levels are identified and then in the second step, the confidence of frequent itemsets are calculated. In the CMAR algorithm, however, "*CMAR maintains the distribution of various class labels among data objects matching the pattern. This is done without any overhead in the procedure of counting (conditional) databases. Thus, once a frequent pattern (i.e., pattern passing support*

threshold) is found, rules about the pattern can be generated immediately" (Li et al., 2001, p. 4). This is a major advantage for CMAR over CBA in the rule generation phase.

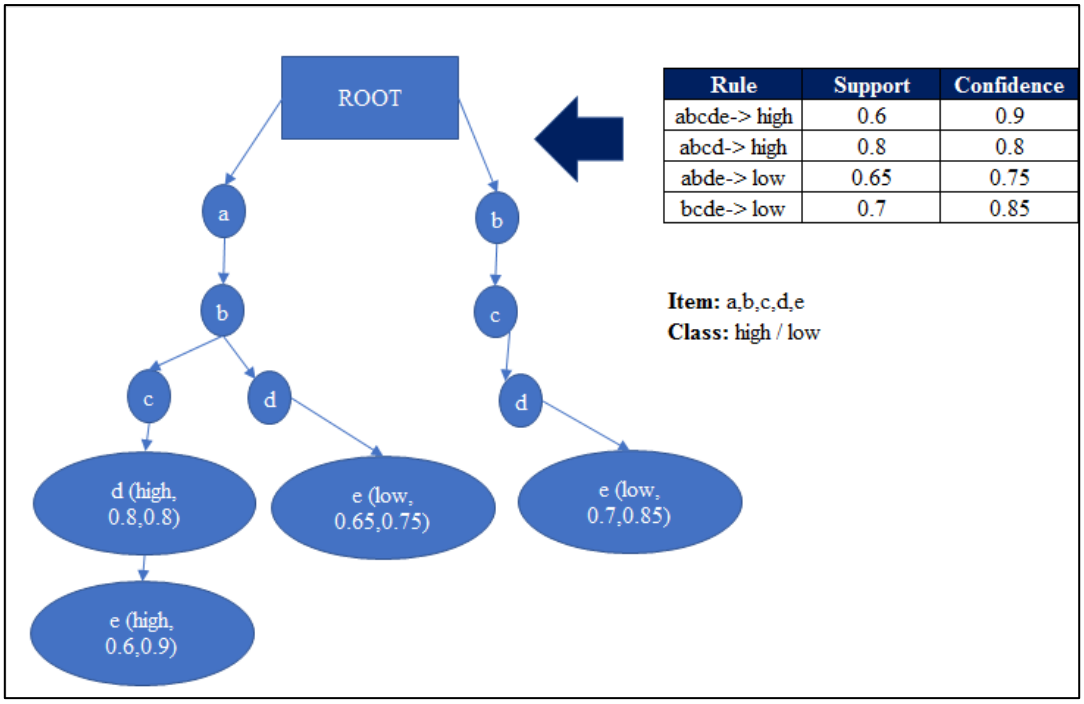
2.4.2 Data Storage CR-tree representation

Li et al. (2001) also propose a new approach to data storage and retrieval of a large number of association rules. The authors present a structure called a CR-tree which is a prefix tree structure. CR tree is a prefix tree structure that exploits sharing among rules. The main advantage of the CR-tree structure is that the representation of the data in this way means rules can be stored in a compressed way thus saving memory. The authors state that in their experiments about 50-60% of space can be saved by using a CR-tree structure to store the data.

Figure 2.8 below shows an example of the compression capability of CR-tree. A CR-tree has a root node. All of the values from the left-hand side of the association rules are sorted according to their frequency. The first rule abcd -> high is inserted in the tree, the class and the support and confidence of the rule are stored at the node. The next rule abcde -> high is then inserted but is simply an extension of the last rule where *e* is added as a new node, again the class, support and confidence are registered with the node. Storing each element of the left-hand side of the rules individually would require 17 cells while in this CR-tree representation just 11 cells are needed so a saving of 35% in this example.

2.4.3 Pruning

The number of rules generated from the Association Rule mining stage can be extremely large. In order to reduce the quantity of rules to a smaller number that are effective and efficient for classification purposes, a post pruning strategy is required. There are a number of pruning approaches employed across the state of the art models for classification using association rules. Across, the CBA and CMAR algorithms there are both similarities and differences to the pruning strategies employed.



Rule	Support	Confidence
abcde-> high	0.6	0.9
abcd-> high	0.8	0.8
abde-> low	0.65	0.75
bcde-> low	0.7	0.85

Figure 2.8: Example of the compression capability of CR-tree

Both the CBA and CMAR algorithm generate a global order of rules or a rule precedence. Given two rules *R1* and *R2*, *R1* ranks above *R2* as follows:

(1) Confidence R1 > Confidence R2
(2) Confidence R1 = Confidence R2 but Support R1 > Support R2
(3) Confidence R1 = Confidence R2 and Support R1 = Support R2 but R1 has fewer attributes on the left-hand side than R2.

The CMAR algorithm implements three steps in the pruning process. First, CMAR employs general to specific ordering. A rule R1 is said to be a general rule w.r.t R2, if the left-hand side of R2 is a subset of the left-hand side of R1. CMAR uses general and high-confidence rules to prune more specific and lower confidence rules. Given two rules, R1 and R2, where R1 is a general rule w.r.t R2. CMAR prunes R2 if R1 also has a higher rank than R2. More general rules are favourable to reduce overfitting and improve the ability for the model to generalize.

In the second pruning step, CMAR uses a statistical measure to further prune the rule set. In this step, CMAR selects only positively correlated rules. For each rule, R: P -> C, the

algorithm tests whether P is positively correlated with C using x^2 testing. Only rules that are positively correlated and above a certain statistical significance threshold are carried forward for use in classification. To perform the chi-square test, the rule R is tested against the whole database.

In the third pruning step, CMAR uses database coverage to prune rules. Database coverage ensures that each rule brought forward to the classification model can classify at least one training instance correctly. Database coverage also ranks the rules by precedence ensuring that the rules brought forward are those rules with the highest ranking among rules that can cover the instance. In association rule learning, a rule R covers an instance *d* if the attributes of the instance satisfy the condition of the rule. An example of rule coverage is outlined below using two example rules.

R1 (Age 45 – 55 = Yes), (Employed = No) -> No Sale
R2 (Age 25 – 35 = Yes), (Occupation = Mechanic) -> Sale

Name	Age	Employed	Occupation	Location	Class
Colin	25 - 35	Y	Mechanic	Dublin	Sale
John	45 - 55	N	Unemployed	Galway	No Sale

R1 covers the instance John
R2 covers the instance Colin

In this pruning step, CMAR retains more rules than CBA. In the CBA algorithm, once one rule covers an instance the instance is removed from the training dataset. In CMAR, in order for an instance to be removed, the instance must be covered by some threshold number of rules δ (in CBA δ = 1). Once the threshold is achieved all of these rules are brought forward for classification. The pruning approach is outlined in Figure 2.9. The CMAR approach increases the number of rules brought forward for classification purposes but should lead to less overfitting as a number of rules are used to make a collective decision when a new instance is to be classified. Li et al. (2001) state that their approach CMAR, using multiple rules for prediction, leads to higher average class accuracy than CBA and C4.5.

1. Input: rule set sorted according to interestingness measure, threshold σ
2. Set cover count of all training instances to 0.
3. For all rules *r* sorted order DO:
   (a) For all training instances *d* DO:
      i. If *r* covers *d* **Then**
      ii. mark d
      iii. **If** *r* classifies *d* correctly **Then**
      iv. mark *r*
   (b) **If** *r* marked **Then**
      i. Increase cover count of all marked *d*
      ii. **If** cover count of a marked *d* exceeds a threshold σ **Then**
      iii. remove *d* from the training set
   (c) delete all marks from each training instance *d*.

Figure 2.9: Pruning using database coverage.
(Source: The scheme is an adapted version from Li et al. (2001) and Liu et al. (1998))

As described above, CBA uses a simpler version of database coverage than CMAR for pruning where the threshold δ is set to one. This results in a smaller set of rules brought forward to be employed in the classification model. CBA also employs a number of additional obligatory and optional pruning steps. The additional obligatory pruning step completed in CBA is called default rate pruning and the optional pruning step is pruning based on the pessimistic error rate (Quinlan, 1993).

The CBA algorithm uses three steps to perform the default rate pruning process. Using the rule precedence set out above the rules are sorted based on highest precedence. Rules with the highest confidence are ranked at the top. Figure 2.10 below outlines the method applied where R is the set of sorted rules. Lines 2 – 12 below, select rules from R. For each rule r, go through D to find those cases covered by r (line 5). R is then marked if it correctly classifies a case d. d.id is the unique identification number of d. If r can correctly classify at least one case, it will be a potential rule for use in the classification stage. Those cases it covers are then removed from D (line 9). A default class is also selected (the majority class in the remaining data), which means if the algorithm stopped selecting more rules to include in the

classifier C this class will be the default class of C. A computation is then carried out to record the numbers of errors that are made by the current iteration of C. This is the combined sum of the total number of errors made by C and the errors made by default class across the total training data. When there is no rule or no training case left then the selection process is complete.

At this point, the model then checks back across the rules and errors logged. The first rule at which there is the least number of errors recorded on D is the cutoff point. All the rules after this rule can be pruned because they only produce more errors. The undiscarded rules together with the default class are the final list of rules for classification.

1. R = sort (R);
2. **for** each rule r ϵ R in sequence **do**
3. *temp* = $\emptyset$
4. **for** each case $d \epsilon D$ **do**
5. **if** d satisfies the conditions of r **then**
6. store d.id in temp and mark r if it correctly classifies d;
7. **if** r is market then
8. insert r at the end of C
9. delete all the cases with the ids in *temp* from D;
10. selecting a default class for the current C;
11. compute the total number of errors of C;
12. **end**
13. **end**
14. find the first rule p in C with the lowest total number of errors and drop all the rules after p in C;
15. Add the default class associated with p to the end of C and return C (our classifier).

Figure 2.10: CBA Pruning Process
(Source: Liu et al., 1998)

CBA also uses an optional pruning step during the mining for association rule phase, Section 2.4.1 above. The approach used is known as pessimistic pruning and is based on the

pessimistic error rate (PER) approach proposed in C.4.5 (Quinlan, 1993). In this approach, the training data is used to generate and prune rules. C4.5 examines each decision node and will replace the node with either the leaf or the most frequently used branch with fewer estimated errors.

In the case of classification using association rules, the PER is used to evaluate the expected classification error of the rules. For every rule identified through the mining process, during the pruning step the PER of the rule is compared with the PER value of general rules. The rule is pruned if the PER value is higher than the PER value of at least one of its general rules. This indicates the rule is less likely to be accurate than one of its general rules. The formal definition of PER pruning states, let R be rule set. A rule r_j can be pruned from R if there is a rule r_i in R, such that (i) r_j is a subset of r_i (ii) len(r_j) - len(r_i) = 1 and (iii) error(r_1) < error (r_j). Unlike database coverage described above, pessimistic pruning retains all rules and database instances until the end of the pruning. Therefore, each rule is always compared with all other available rules in the dataset. Liu et al. (1998), demonstrate that pruning using pessimistic pruning significantly reduces the number of rules with no impact on model accuracy.

2.4.4 Using rules to make classifications

The CBA approach to classification is much simpler when compared to the approach for pruning outlined above. In CBA, when a new instance needs to be classified the algorithm simply searches the pruned and ordered list of rules and picks out the top rule (highest confidence) that covers this instance. The predicted class of the new instance is the class from the top rule. As mentioned above, CBA also uses a default class. Where no rule covers the new instance seen CBA assigns the default case identified during training.

Unlike CBA, in the classification phase, CMAR selects a subset of high-quality rules for prediction rather than simply the top rule. The algorithm analyses a subset of high-quality rules that match the new unseen instance. The CMAR authors, state that such a simple selection process, as applied in CBA, using just one rule may affect the classification accuracy of the model through overfitting.

If all the rules that match a new instance have the same class, then CMAR simply assigns this class to the new instance. If the subset of rules has different class predictions a voting methodology is required to select the class for the new item. There are a number of weighted voting techniques including majority vote, simple linear weighting and inverse function. The CMAR algorithm uses a more complex weighting methodology based on weighted x^2. The authors propose a number of approaches using x^2.

If the rules are not consistent in terms of predicted class, CMAR divides the rules into groups which have the same class label. CMAR then compares the effects of these groups to identify the strongest group and uses the label from that group for prediction. In CMAR, the strength of a group is calculated using a Weighted Chi-Squared (WCS) measure.

This is done by first defining a *Maximum Chi-Squared* (MCS) value for each rule A -> c:

$$MCS = (\frac{(\min(sup(A), sup(c)) - \sup(A)\sup(c))}{N})^2 * N * e$$

Where:

1. sup(A) = support for antecedent.
2. sup(c) = support for consequent.
3. N = Number of records in test set.
4. e is calculated as follows:

$$e = \frac{1}{\sup(A)\sup(c)} + \frac{1}{\sup(A)\,N - \sup(c)} + \frac{1}{N - \sup(A)\sup(c)} + \frac{1}{(N - \sup(A))(N - \sup(c)}$$

For each group of rules, the Weighted Chi-Squared value is defined as:

WCS = The sum of (Chi-Squared * Chi-Squared)/(MCS)

The authors also tested a number of other approaches. A simpler approach is to take the strongest rule from each group, the rule with the highest x^2. However, this approach may

favour minority classes, see Figure 2.11 below for illustration. Another alternative proposed by the authors is to use the compound correlation of rules as the measure. For example, one option is to sum up values in a group as the measure of the group. However, this method suffers from the same problem that it may favour the minority class too much. The empirical results presented by the authors suggest that the weighted chi-squared measure is the best approach for identifying the class for new unseen instances.

For example, if there are two rules for mortgage sales R1: employed = no -> No Sale (support 30, confidence 60) and R2: Over 30 = yes -> Sale (support = 200, confidence 97.5%). The observed and contingency tables are presented below in Figure 2.11.

Observed Contingency Rule 1

R1	Sale	No Sale	total
employed = yes	438	32	470
employed = no	12	18	30
total	450	50	500

Observed Contingency Rule 2

R2	Sale	No Sale	total
Over 30 = yes	195	5	200
Over 30 = no	256	48	304
total	451	53	504

Expected Contingency Rule 1

R1	Sale	No Sale	total
employed = yes	423	47	470
employed = no	27	3	30
total	450	50	500

Expected Contingency Rule

R2	Sale	No Sale	total
Over 30 = yes	178	22	200
Over 30 = no	272	28	300
total	450	50	500

Figure 2.11: Chi-Squared rule choice illustration

In this example, the χ^2 value for rule 1 is 88.6 and 35 for rule 2. If the choice of rules was based on χ^2 values only Rule 1 would be chosen from the above rules. However, a closer look at Rule 2 shows it has both higher confidence and support. Weaknesses such as this investigated by Li et al. (2001), resulted in the development of the WCS measure.

2.5 The impact of class imbalance on Association Rules and the SPARCCC algorithm

Many real word problems face the issue where the dataset is imbalanced. Target class imbalance is described by Japkowicz (2000) as domains for which one class is represented by a large number of examples while the other class is represented by relatively few examples. Typical domains with imbalanced target classes include disease identification, fraud detection

and marketing response. In these examples, one of the classes typically covers the majority of instances and the second class is rarely evidenced in the data. In many cases, the smaller second class is the class of interest for the research. For example, in the fraud identification domain, only a small number of transactions will be fraudulent but these are the transactions that are most interesting for fraud identification purposes. In this research, the target class is whether the customer took out a mortgage loan or not. The volume of customers who took out a mortgage during the performance window is a low percentage of the total customer base (<2%), therefore the dataset for this research is largely imbalanced.

The problem of imbalanced datasets can also affect unsupervised learning approaches. Taking the classical retail store transactions example again, bread and milk, occur frequently and will have both high support and confidence. Other associations might be rare, for example, customers buying a *vacuum cleaner* and *washing machine,* although the items are likely to be bought together and therefore have high confidence they are not items that are frequently bought and therefore will have low support. This rule is then likely to be excluded from the final list of rules. To find this association the minimum threshold needs to be set very low, which will then cause the combinatorial explosion described in Section 2.3. This example is similar to rules that have sales = Yes in this research. Given these examples appear rarely it is likely that many rules with this value on the right-hand side of the rule will be excluded. This makes it difficult for the algorithm to predict the rare class accurately.

The risk with imbalanced datasets is that the model will focus on the majority class because it is evidenced more regularly in the dataset and the model will make poor predictions for the minority class of interest. Although the overall accuracy of the model will be high given the level of imbalance, a model that predicts the majority class for all examples will be correct in 98% of cases, however, these outputs provide no value. Weiss and Provost (2003) carried out a study over twenty-six datasets, the results showed that the error rate of minority class classification rules was 2-3 times that of the rules that identify majority class examples and minority-class examples are much less likely to be predicted than majority class examples. Many practitioners have observed that for extremely skewed class distributions the recall of the minority class is often 0, there are no classification rules generated for the minority class.

It is therefore important when dealing with imbalanced data to choose the right evaluation metrics, described in detail below in 2.5. One example is ROC analysis and the area under the

ROC curve (AUC) to assess classification performance of the models (Bradley 1997; Provost & Fawcett, 2001). The AUC measure does not place more emphasis on one class over the other, so it is not biased against the minority class. Other alternative measures such as balanced accuracy, the average of specificity and sensitivity, may be more applicable than simple accuracy at measuring the performance of algorithms on imbalanced datasets.

Given that class imbalance in real-world datasets is ubiquitous the class imbalance problem has been researched heavily in recent years in the machine learning domain (Chawla, Japkowicz, & Kotcz, 2004). A number of different approaches have been proposed and tested to improve prediction accuracy on imbalanced datasets. These methods include, among others, sampling approaches, cost-sensitive learning and learning only the rare case.

The most basic sampling approaches include under-sampling and over-sampling. Under-sampling removes majority class records, this can be done by using random under-sampling where a certain volume of records with the majority class are removed at random. Over-sampling is increasing the volume of the minority class. The simplest approach to over-sampling involves simply duplicating existing minority class records. A number of problems with oversampling have been identified through research on the problem of imbalanced datasets. Oversampling can cause overfitting as the new examples are typically exact matches of the existing cases in the rare class. Oversampling does not introduce any new data into the experiment and certain research has shown oversampling to be ineffective in resolving the imbalanced dataset problem (Ling & Li, 1998; Drummond & Holte, 2003).

There are also more advanced forms of sampling that combine oversampling and undersampling or introduce new data in a more advanced way. SMOTE is one such method of a more advanced sampling technique (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). While oversampling simply replicates existing data points, SMOTE, oversamples by introducing new data using statistical methods. Minority-class examples are generated by adding examples from the line segments that join the k minority-class nearest neighbours (SMOTE uses k=5). This causes additional generalisation, as opposed to the potential overfitting that may be caused by exactly replicating existing data in simple oversampling.

Another approach to dealing with imbalanced datasets is cost sensitive learning (Pazzani, Merz, Murphy, Ali, Hume, & Brunk 1994). This approach increases the value of correctly

identifying the minority class cases and reduces the value of correctly identifying the majority class records. This is typically done by assigning a greater cost to false negatives than to false positives. Areas where cost-sensitive learning are appropriate include medical diagnostics, fraud identification or terrorism prediction. In these cases, a false negative can lead to the loss of life in the examples of terrorism and medical diagnosis, while a false positive can lead to increased expenses in terms of investigation or testing. One difficulty with cost-sensitive learning models is getting access to the right misclassification cost ratio to apply for the given scenario.

SPARCCC

Verhein and Chawla (2007), challenge the existing *Associative Classifiers*, including CBA and CMAR described above, identifying classification performance concerns in imbalanced datasets. The authors state that association rule classifiers using the *support confidence* framework do not perform well with imbalanced datasets. Verhein and Chawla (2007, p. 1), propose a new measure Class Correlation Ratio, *"which measures the relative class correlation of a rule. A high CCR is desirable because it means the rule is more positively correlated with the class it predicts than the alternative(s)"*. The authors prove that confidence and support are biased towards the majority class in the context of CCR.

Verhein and Chawla (2007) propose a new algorithm SPARCCC using only rules that are statistically significant and positively correlated where the antecedent is more correlated with the class it predicts than other class(es). In comparison to the CBA and CMAR algorithms which require the user to define specific thresholds for support and confidence, the SPARCCC algorithm is parameter free, only using standard significance levels to prune rules.

The SPARCCC algorithm is built using significance and the authors propose a new metric Class Correlation ratio. The authors state "*We are interested in rules $X \rightarrow y$ that are statistically significant in the positively associated direction*" (Verhein and Chawla, 2007, p. 2). To test for significance the authors use Fisher's Exact Test (Equation 2.1) on contingency tables of the form in Figure 2.12. The authors select only rules that are below a certain P value which outputs rules that are statistically significant in the positively associated direction.

	X	¬X	Σrows
y	a	b	a + b
¬y	c	d	c +d
Σcols	a + c	b + d	n = a + b + c + d

Figure 2.12: Contingency tables for Fisher's Exact Test

$$p([a,b,c,d]) = \sum_{i=0}^{\min(b,c)} \frac{(a+b)!\,(c+d)!\,(a+c)!\,(b+d)!}{n!\,(a+i)!\,(b-i)!\,(c-i)!\,(d+i)!}$$

Equation 2.1

Correlation is the second element of the SPARCCC algorithm. The authors state *"We are interested in rules $X \rightarrow y$ where X is more positively correlated with y than it is with $\neg y$"* (Verhein and Chawla, 2007, p.2). For correlation, the authors propose a metric Class Correlation Ratio to measure how much more positively the rule is correlated with the class it predicts relative to other classes. The correlation definition is outlined in Equation 2.2.

$$c\hat{o}rr(X \rightarrow y) = \frac{\sup(X \cup y).|D|}{\sup(X).\sup(y)} = \frac{a.n}{(a+c).(a+b)}$$

Equation 2.2

Where $c\hat{o}rr\ (X \rightarrow y)$ is greater than 1 or less than 1 X and y are positively and negatively correlated respectively. In this approach, CCR is used to measure how correlated *X* is with *y* compared to $\neg y$. The CCR equation is described below in Equation 2.3.

$$CCR(X \rightarrow y) = \frac{c\hat{o}rr\ (X \rightarrow y)}{c\hat{o}rr\ (\mathrm{X} \rightarrow \neg \mathrm{y})} = \frac{a.(b+d)}{b.(a+c)}$$

Equation 2.3

The SPARCCC model only uses rules with $c\hat{o}rr > 1$ and CCR > 1, which ensures the rule is statistically significant in the positive association direction X -> Y, rather than in the opposite direction X-> ¬Y.

Using the CCR approach described above the authors prove that confidence and support are biased towards the majority class in imbalanced datasets in the context of CCR. This is a major concern in terms of using the CBA and CMAR algorithms on imbalanced datasets. The authors identify two areas where the confidence framework used in CBA and CMAR is biased towards the majority class. The first bias is where a highly confident rule predicting the majority class may, in fact, be more negatively correlated than the same rule predicting the other class(es), and the second concern is where a rule that is more positively correlated but predicts the minority class may have much lower confidence than the same rule predicting the other class(es).

To avoid the biases identified above, the SPARCCC algorithm does not rank with confidence, instead, the SPARCCC algorithm uses the CCR to identify and rank rules to be used for classification. The authors state that in balanced datasets the CCR ranking and the confidence ranking are comparable, however, in imbalanced datasets, the CCR measure outperforms the confidence measure for the bias reasons described above.

For search and pruning the authors propose the use of GLIMIT (Verhein & Chawla, 2006) as the underlying association rule learning algorithm but state that in line with previous experiments any of the alternative association rule learning algorithms such as Apriori and FP-Growth could be used.

The authors' experiments comparing SPARCC and CCR to CBA, C4.5 and alternative associative classifiers indicates comparable classification performance on balanced datasets but significantly improved classification performance on imbalanced datasets. These experiments point towards an inability for CBA and CMAR to achieve a high level of classification accuracy on imbalanced datasets.

2.6 Data Discretisation Approaches

For certain datamining algorithms, the dataset may need to be transformed to a binary format to be used by the model. Data discretisation is a data transformation approach where continuous data is converted to categorical data. There is a range of data discretisation techniques that can be applied. The main categories of discretisation are unsupervised discretisation approaches and supervised discretisation approaches. Unsupervised approaches are generally much simpler and do not take account of the class label. In unsupervised approaches, there are typically two options, one where the number of intervals is set and the second where the number of records per interval is set. These two methods are known as equal-width discretisation and equal-frequency discretisation. In the equal width approach, the user defines the number of intervals and the model simply divides the range of values into the user defined number of equal width intervals. In equal frequency discretisation the intervals are split such that each interval has the same number of records. An example of these discretisation approaches is provided below in Figure 2.13.

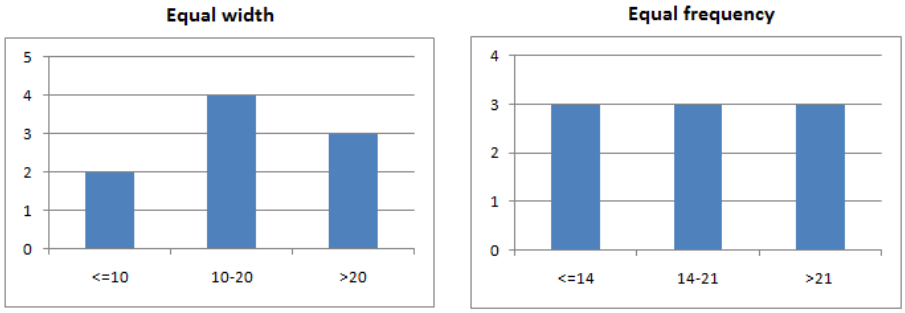


Figure 2.13: Example of unsupervised data discretisation techniques

Supervised discretisation approaches take account of the class label in making the interval cut off points. The objective is to identify the best split such that the majority of the values in a bin correspond to the same class label while also minimizing the information loss from transforming continuous variables into categorical variables. There are a number of supervised data discretisation approaches. Kerber (1992) and Liu and Setiono (1997) present supervised methods which discretise numeric data based on the χ^2 statistic.

2.7 Benchmark models

A number of benchmark models will be developed as part of this research to enable a comparison between traditional classification based models and the classification models developed using association rules. The benchmark models chosen for comparison purposes are 'rule' or 'tree' based models. The most popular implementations of such algorithms are 'CART' (Breiman, Friedman, Olshen, & Stone 1984) and 'C4.5' (Quinlan, 1993). The use of rule based models for comparison purposes is in line with existing research in this area where models performing classification using association rules are typically compared to existing rule based models (Li et al., 2001).

Conditional Inference Trees

The first benchmark model that will be implemented in this research is conditional inference trees (Hothorn, Hornik & Zeileis, 2006). Similar to traditional decision trees (Breiman et al., 1984) conditional inference trees recursively partition the data by performing a univariate split on the dependent variable. The difference between conditional inference trees and other decision tree methods is the choice of measure used to select variables at each node. Methods such as Breiman et al. (1984) use information measures such as the Gini coefficient or entropy and select the covariate showing the best split. Hothorn et al. (2006) claim that there are two main problems with these decision tree methods using an exhaustive search approach.

The first problem is that these methods have overfitting problems and secondly the models have a selection bias towards covariates with many missing values or covariates with many possible splits. To avoid overfitting, algorithms such as C4.5 can implement a pruning strategy after the tree is fully grown. Instead of applying a pruning approach conditional inference trees implement a unified framework for handling both selection bias and overfitting. In comparison to the traditional decision trees, conditional inference trees use a statistical test procedure in order to select variables instead of an information gain measure such as Gini coefficient. The statistical test procedure applied in conditional inference trees is based on permutation tests, a class of widely applicable non-parametric tests. Permutation tests randomly shuffle the data to get the correct distribution of a test statistic under a null hypothesis. To try to overcome the overfitting and selection bias issues of other decision tree

models the authors propose a new method that separates the variable selection from the splitting criteria. The approach used by Hothorn et al. (2006) follows three steps, in step 1 variable selection is made, in step 2 the methodology for splitting is chosen and step 3 is the recursive application of these first two steps. The authors state "*that recursive partitioning based on statistical criteria as introduced in this paper lead to regression models whose predictive performance is as good as the performance of optimally pruned trees by means of benchmark experiments*" (Hothorn et al., 2006, p. 2). When the tree is required to classify a new previously unseen record the new record is passed through the tree and the final node the record arrives at is used to classify the record, in this research project the classification is a binary result sale or no sale.

Random Forests

The second benchmark model that will be implemented in this research is random forests (Breiman, 2001). Random forests is an ensemble method using a large number of classification trees as opposed to one tree in conditional inference trees. When the forest is required to classify a new previously unseen record the new record is passed through each of the trees to arrive at many individual classification values. The algorithm then takes each of these individual results and applies a voting system to arrive at the final classification. The forest chooses the class with the most votes over all the trees in the forest.

Each tree is grown by first selecting a sample, if there are *N* cases in the training dataset, the sample will be *N* records selected at random with replacement. If there are *M* input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these *m* variables is used to split the node. The value of *m* is held constant during the forest growing. No pruning strategy is applied in building out the trees so each tree is grown to the largest extent possible. Within random forests, there are a small number of parameters that can be tuned including *m*, the number of features to select at each node and the total number of trees to build. The value for *m* is an important factor in producing a good classifier. The value for *m* affects the correlation between the trees and the strength of the individual trees the two factors identified by Breiman (2001) affecting the performance of the model. Increasing the value of *m* increases the correlation between trees, a negative impact on performance while increasing the strength of individual trees a positive

impact on performance. The objective is to identify a value of m in a range that minimises the errors produced by the model.

The out of bag (OOB) error is used to identify the range for *m* while also providing error estimation during model building. After the *N* records have been sampled without replacement to build an individual tree a third of the records are taken out of the sample. Using this approach records that have not been used to build the tree can be put down the tree to get an immediate classification. Using the OOB error rate to get a running unbiased estimate of the classification error generally means there is no requirement to use cross-validation approaches when using the random forest algorithm.

2.8 Model Validation Methods

A classification model is trained on historical data and then used to predict the result for previously unseen data. To measure if the model generalises well the performance on the training dataset must be validated. There are a number of approaches in data mining to validate the results of the model. The aim of the testing is to identify if the model is under or overfitting the data. This is the basic idea for a whole class of model evaluation methods including, for example, cross-validation (Friedman, Hastie, & Tibshirani, 2001).

Overfitting is evidenced when the performance of the model in training does not translate to new unseen data. Overfitting can occur for a number of reasons but generally means the model is learning concepts, noise or patterns in training that does not apply to new data. This essentially means that the model has been too tightly fitted to the specific data points in the training data, trying to model patterns in the data originating from noise. The performance of the model on new data is then worse than the performance in training.

Underfitting is evidenced when the performance of the model is poor both in training and in use with new data. This indicates the model being trained is too simple and cannot identify the patterns and relationships in the data. Underfitting may occur, for example, if a linear model was used to fit non-linear data.

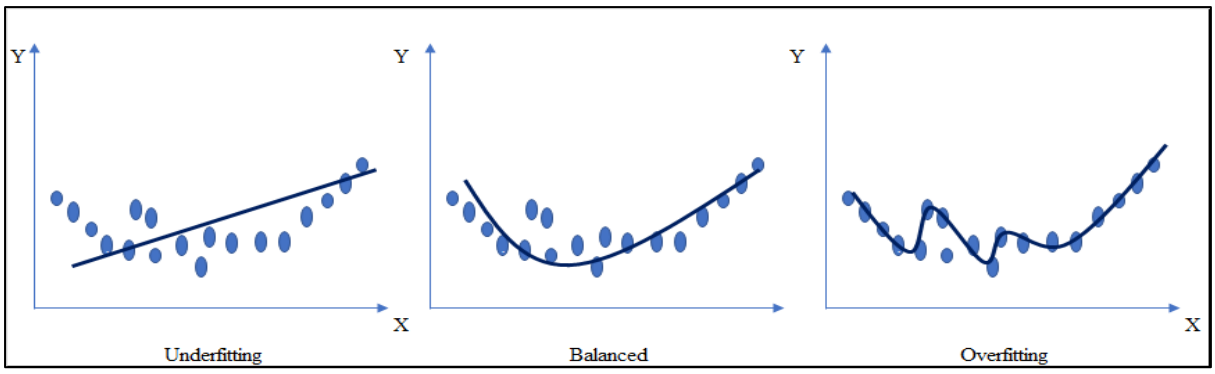


Figure 2.14: Graphical illustration of underfitting and overfitting

In order to judge if a model is overfitting or not, the model is tested on previously unseen data and the performance compared with the performance on the training data. Estimating this error can be done in several different ways.

Dataset Split

The most straightforward approach to estimating the generalised performance of a model is to partition the dataset into two or more partitions, a training set, a validation set and a test set. For example, the simplest method is to split the data into two subsets a training subset and a test subset. The model is developed on the training dataset and then tested to see how it performs on previously unseen data. This indicates how well the model generalises to new data instances. This approach is common and very respected but may not be appropriate where the dataset is small. Using this approach on small datasets is likely to result in poor performance and biased results. This method also holds out a large portion of the dataset that otherwise may have been used for training purposes. Other approaches to validation aim not to waste this large amount of valuable data in the training phase.

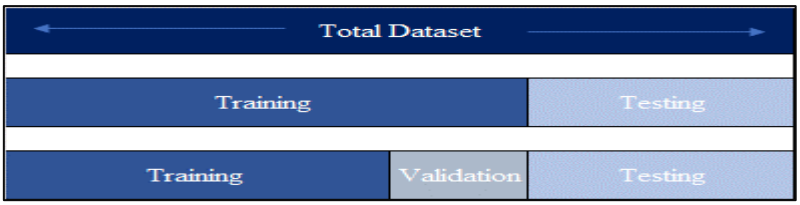


Figure 2.15: Illustration of the Dataset split data validation technique

K- fold Cross-Validation

K-fold cross-validation is one approach to account for the biases and data wastage identified in Dataset Split. In K-fold cross-validation the full dataset is used to train and test the model's generalization performance. The idea behind K-fold cross-validation is to split the dataset into K subsets and use each of the K-folds as a test set while training the model on the remaining K-1 subsets of the dataset. K models are developed and the performance metric used is then averaged across each of the K models. The more folds used for evaluation the smaller the bias but this also leads to higher variance across the folds (Geman, Bienenstock, & Doursat, 1992).

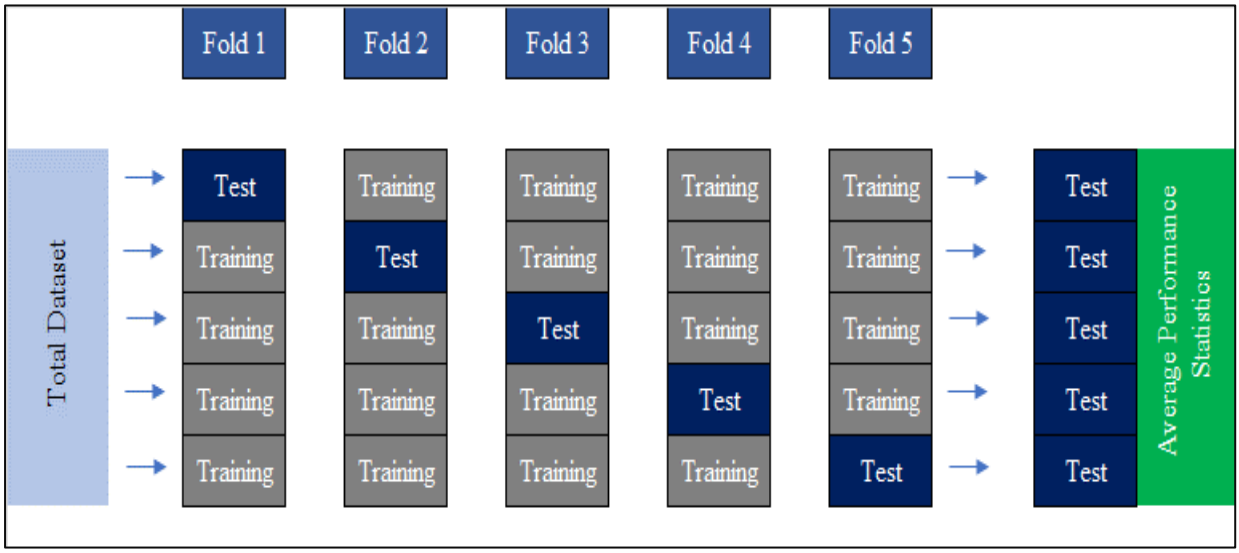


Figure 2.16: K-fold Cross-Validation illustration

Leave-one-out cross-validation

Leave-one-out cross-validation (LOOCV) is K-fold cross-validation taken to its logical extreme, with K equal to N, the number of data points in the set. That means that N separate times, the function approximator is trained on all the data except for one point and a prediction is made for that one point. As before, under K-fold cross-validation, the average of the relevant performance metrics used is computed and used to evaluate the model.

Bootstrap

The bootstrap approach is similar to K-Fold Cross-Validation, however, in this case, the training sets are sampled from the whole dataset with replacement. This introduces an amount of randomness into the sample data. The trained model is then used to predict the outcome of the data not chosen in the sampling process. This process is repeated many times and the average of the relevant performance metrics used is computed to evaluate the model.

Efron and Tibshirani (1997) propose the 0.632+ Bootstrap method to reduce the upward bias of leave-one-out bootstrap. The 0.632+ Bootstrap method takes into account the amount of overfitting in the model training. In order to calculate the overfitting term, the authors define the *non-information error-rate*, $\hat{\gamma}$, which is a measure of the classification error assuming that predictive variables and the class are independent. Given a classification learned from a dataset *D, φ (**x**; D)*, an estimation of γ can be obtained as follows:

$$\hat{\gamma} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\delta(c^i, \phi(x^{(j)}; D))$$

Equation 2.4

The relative overfitting rate is defined as:

$$\hat{R} = \frac{\widehat{Err^{(1)}} - \overline{err}}{\hat{\gamma} - \overline{err}}$$

Equation 2.5

The final 0.632+ estimator is calculated as follows:

$$\widehat{Err^{(.632+)}} = (1 - \frac{0.632}{1-0.368\hat{R}})\,\overline{err} + \frac{0.632}{1-0.368\hat{R}}\,\widehat{Err^{(1)}}$$

Equation 2.6

The idea of the bootstrap 0.632+ method is to obtain a trade-off between zero bootstrap and resubstitution that depends on how much the classifier overfits the training set.

2.9 Model Performance Metrics

This section describes some of the measures used to assess the quality of the class predictions made by a model. In this project, the class predictions will be a binary yes/no, where yes indicates the customer took a mortgage product and a no indicates the customer did not take a mortgage product. To calculate model performance a contingency table known as a confusion matrix is created. This section describes the confusion matrix and the relevant metrics that are available once the confusion matrix has been constructed.

Confusion Matrix

A confusion matrix is a table that documents the class predicted by the model and the actual correct class. Figure 2.17 below provides a sample confusion matrix for the prediction of whether a given animal is a cat or a dog. The prediction rows represent the class predicted by the model and the columns reflect the actual known class.

Confusion Matrix		Actual	
		Cat	Dog
Predicted	Cat	20	5
	Dog	3	50

Figure 2.17: Example of a confusion matrix

Figure 2.18 below provides a more generic version of a confusion matrix used to compare the results of the various models applied in this project. In this project, a 1 reflects a product sold or a positive and a 0 reflects no sale or a negative.

Confusion Matrix		Actual	
		0	1
Predicted	0	TN	FN
	1	FP	TP

Figure 2.18: Generic confusion matrix and key metrics

The confusion matrix produces four key outputs which can then be used to create appropriate performance metrics. The four outputs are True Negatives (TN), True Positives (TP), False Positives (FP) and False Negatives (FN).

True Positives – In our example, this implies the model correctly identified the data instance as a sale.

False Positives – In our example, this implies the model incorrectly identified the data instance as a sale when in fact no sale took place. This is defined as a Type I error.

True Negatives – In our example, this implies the model correctly identified the data instance as a no sale.

False Negatives – In our example, this implies the model incorrectly identified the data instance as no sale when in fact a sale took place. This is defined as a Type II error.

Once the confusion matrix is constructed this facilitates the calculation of a number of performance metrics. The simplest performance metric is accuracy which measures how many of the total instances the classifier scored correctly (Equation 2.7). As described above, when the dataset is highly imbalanced this performance metric may not be appropriate. Sensitivity, also known as the true positive rate (TPN) or recall, measures the proportion of positives (sale occurred) that have been correctly identified as a positive (Equation 2.8). Specificity or the true negative rate measures the proportion of negatives (no sale occurred) that have been correctly identified as a negative (Equation 2.9).

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

Equation 2.7

$$Sensitivity = Recall = \frac{TP}{TP + FN}$$

Equation 2.8

$$Specificity = \frac{TN}{FP + TN}$$

Equation 2.9

Other metrics include precision which measures the percentage of the instances predicted to be positive that are correctly predicted (Equation 2.10) and negative predictive value (NPV) measures the percentage of instances predicted to be negative and were correctly predicted as negative (Equation 2.11).

$$Precision = \frac{TP}{TP + FP}$$

Equation 2.10

$$Negative\ Predictive\ Value = \frac{TN}{TN + FN}$$

Equation 2.11

These metrics are often combined to create a balanced performance metric. Two such metrics are balanced accuracy and the F1-score. Balanced accuracy is particularly useful for imbalanced datasets. The F1-score is the harmonic mean of precision and recall where an F1-score reaches its best value at 1 (perfect precision and recall) and worst at 0.

$$Balanced\ Accuracry = \frac{Sensitivity + Specificity}{2}$$

Equation 2.12

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Equation 2.13

Figure 2.19 below provides an example confusion matrix aligned to this research project which is attempting to accurately classify mortgage sales. The metrics outlined above are then demonstrated using this example in Figure 2.20.

Confusion Matrix		Actual	
		0 (No Sale)	1 (Sale)
Predicted	0 (No Sale)	90	5
	1 (Sale)	7	80

Figure 2.19: Example mortgage sales confusion matrix

Metric	Formula	Score	Metric	Formula	Score
Accuracy	(90+80) / (90+80+7+5)	93%	Negative Predicted Value	90 / (90+5)	95%
Specificity	90/ (90+7)	93%	Balanced Accuracy	(Specificity + Sensitivity) / 2	93%
Sensitivity (recall)	80 / (80+5)	94%	F1 - Score	2 x (precision.recall)/ precision +recall)	93%
Precision	80 / (80+7)	92%			

Figure 2.20: Calculation example Key performance metrics

2.10 Conclusion

This chapter has summarised and evaluated the existing body of research in the key areas underpinning this research namely data mining methodologies, association rule learning and extending association rules to make class predictions.

Data Mining is a broad area of research endeavouring to identify patterns and draw conclusions from the analysis of data. Data Mining can be further segmented into three categories, supervised learning, unsupervised learning and reinforcement learning.

Association rule learning is a form of unsupervised learning where there is no target variable. In unsupervised learning, the algorithm is simply looking for patterns in the data as opposed to making class predictions. When implementing associations rule algorithms, the objective is completeness, the algorithm is required to find all interesting rules in the dataset above user defined thresholds for support and confidence. The majority of the research on association rules focuses on the speed of rule generation and reducing the computationally expensive nature of the rule generation process.

Classification using association rules is an extension of association rules where rules created using an association rule learning algorithm are extended to make class predictions. This chapter reviewed the seminal papers in this area describing the three-stage process. Step 1 is the adaptation of association rule learning algorithms to generate the classification association rules, Step 2 involves applying a rule pruning technique and Step 3 involves using a subset of high quality rules to make predictions.

The two seminal algorithms CBA and CMAR were discussed in detail. The CMAR research claims that the CBA algorithm overfits based on the simplified approach of using only individual rules to make predictions. CMAR proposes the use of multiple rules to make an individual prediction by using a voting system across the top-quality rules.

Verhein and Chawla (2007), claim that CBA and CMAR are not suitable for imbalanced datasets as they have a tendency to be biased towards the majority class. Given the imbalanced nature of the dataset used in this experiment potential approaches for dealing

with imbalanced datasets such as oversampling and undersampling have been evaluated and compared. The SPARCCC algorithm has been developed to deal with imbalanced datasets so the expectation is that this algorithm will compete best with the traditional classification algorithms in the experiments designed for this research.

Given the computationally expensive nature of Association rule learning, classification models using associations rules often require particular feature engineering approaches. One example described in detail above is the use of supervised and unsupervised data discretisation approaches to convert continuous attributes into discrete attributes.

In Section 2.7 the state of the art in rule based tree models including random forests and Conditional Inference Trees were presented and evaluated. The performance of these two benchmark models will be used for comparison purposes against the performance of the classification using association rule models CBA, CMAR and SPARCCC. How these experiments will be designed is outlined in Chapter 3.

Given the imbalanced nature of the dataset to be used in this experiment it is important that the performance metrics used for evaluation account for this bias. The simplest performance metric is accuracy which measures how many of the total instances the classifier scored correctly, however, when the dataset is highly imbalanced this performance metric may not be appropriate. Alternative performance metrics such as AUC, F1-score and Balanced Accuracy have been proposed as alternative metrics to give a more balanced performance assessment given the imbalance in the dataset.

3. DESIGN AND METHODOLOGY

3.1 Introduction

This chapter outlines the design and methodology of the experiments that will be carried out as part of this research. The aim of the design and methodology is to test whether classification using association rules can outperform traditional classification algorithms in predicting customer mortgage sales in an Irish retail banking context. This chapter is divided into five additional sections.

Section 3.2 presents the data that will be used for the experiments. This section outlines what customer data has been acquired to build the ABT modelling dataset. It describes how the data was acquired, the data discovery process, what data integration, filtering and transformation was carried out on the data, the process of feature engineering and ultimately the output delivered as an ABT for the purposes of analytical modelling. This section also highlights areas within the data where data quality issues were identified and how those issues were handled.

Section 3.3 presents the hardware, software and modelling algorithms used to develop the ABT and to implement the algorithms required to complete the necessary experiments.

Section 3.4 outlines the experiment design for the benchmark models that will be used for comparison purposes. In line with previous research on classification using association rules (Liu et al., 1998; Li et al., 2001) the benchmark models chosen are ‘rule’ or ‘tree’ based models. In this research, the two benchmark models that have been implemented are conditional inference trees and random forests (Hothorn et al., 2004; Breiman 2001). In Chapter 4 the results of these benchmark algorithms will be used for comparison purposes with the results derived from the association rule classification models.

Section 3.5 describes the design of the three association rule experiments that will be performed as part of this research. This section presents how each of the experiments will be set up and how the algorithms will be implemented.

The final section, Section 3.6, describes how the experiments will be evaluated with reference to Sections 2.7 and 2.8 above which describe in detail the model validation methods and model performance metrics.

3.2 Data sources and creation of the ABT

3.2.1 Data Acquisition and Integration

The data for this research project has been acquired from Bank of Ireland, Ireland's largest retail bank. The Bank provides a full suite of banking and insurance products to personal, business and corporate customers. The focus of this project is on personal retail banking customers and particularly the sale of mortgage loans to purchase residential property.

The Bank collects a large amount of data about its customers as they interact with the Bank on a daily basis. Figure 3.1 shows the types of data collected in the organisation. For example, the Bank collects data about customer transactions and spending habits, products held with the Bank, interactions with the Bank collected through customer relationship management (CRM), demographic data from application forms and data from customer complaints.

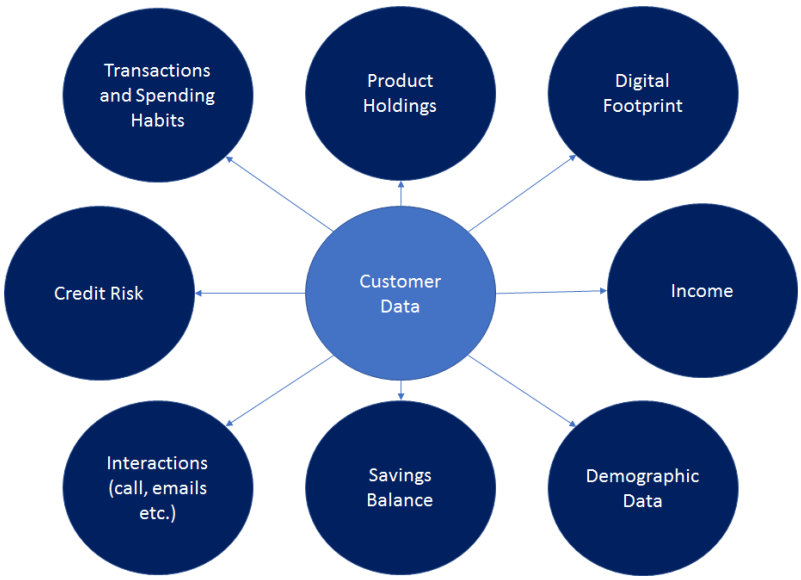


Figure 3.1: Types of data available in Retail Banks

Data is collected in a very siloed manner throughout Bank of Ireland with data predominately stored at a product level rather than a customer level. This is due to legacy systems which were originally set up independently to support only one product, for example, the loans system or the deposits system. Additional individual systems arose from the many acquisitions the Bank has made over its 235-year history. In order to overcome this problem of siloed data repositories the Bank has commenced a large-scale project to build an Enterprise Data Warehouse ('EDW'). This EDW project is collecting data from a number of structured data systems in what is described as an acquisition layer or staging area. The expectation is that over time this data will be integrated into an integrated data layer, however, this part of the EDW project is not mature at this point. Although the data is now in one environment it is not combined and cleansed across systems, there is no customer level view, data cleaning and standardisation has not been performed and there is limited master data management. This means that the data is not yet pre-processed to the point that it is easy to integrate datasets for analytical purposes. The siloed nature of the data requires a large investment of time to integrate the data and derive variables for model building purposes.

A large part of this research project has been invested in the preparation of this siloed data for analysis. This has involved the integration of a large number of these data sources together so it is possible to aggregate data at a customer level rather than the traditional account level view in these systems. Using the traditional banking data sources such as product holdings, balances and transactions c.200 derived features have been engineered for inclusion in the ABT. Each of these features needs to be hand crafted and the code written for each which is an expensive process from a timing perspective. Features such as current product holdings are reasonably straightforward to build, however, lagged features such as growth in deposit balance, or average month end current account balance for the last 12 months are more complex and require more development time. A sample of the features created from traditional banking systems can be seen in Table 3.1. The full list of features from these data sources can be seen in Appendix 1.

Feature Name	Feature Description
Per_Age	Age of the customer
Num_Open_Prod	Number of open products the customer holds
Num_Open_Sav_Prod	Number of open saving products the customer holds
Avg_Credit_Turnover_3Mths	Average of incoming payments into the customer's current account over a 3-month period

Table 3.1: Sample of features built from structured data

A smaller number of features have been developed from semi-structured and unstructured data sources. The Bank collects semi-structured JSON data from online user interactions on the Bank's website, mobile application, tablet application and desktop computer online banking application. This data is collected by putting JavaScript tags on the website or mobile application which collect data on user behaviour such as the pages the user has visited or user events such as buttons clicked. This data can be used to identify visits to the website for certain customers where, for example, customers have looked at certain webpages or started application forms and then abandoned the application process. This online behaviour may be useful to predict when a customer is likely to take out a product. The benefit of using this data in classification modelling is that the data is more dynamic. Customers tend to visit online applications very often compared to demographic data such as age or income which tends to be more static over time. A sample of features built from the online behavioural data sources can be seen in Table 3.2.

Feature Name	Feature Description
Num_Website_Visit_90	Number of visits to the website in last 90 days
Num_Mobile_Visit_90	Number of visits to the mobile app in last 90 days
Num_Mtg_Visit_10	Number of visits to the mortgage home page in the last 10 days
Num_Loan_Visit_10	Number of visits to the loan home page in the last 10 days

Table 3.2: Sample of features built from semi-structured data

Similarly, a small number of features have been abstracted from unstructured textual data. Customers often visit one of the Bank's branches to discuss their financial needs with a financial advisor. This conversation is typically known as a financial needs review. Here the

advisor asks the individual a series of questions to understand their financial life goals and aspirations together with shorter term objectives. The outcome of the conversation is a personalised financial plan for each individual providing the customer with support to achieve the goals identified. For traceability and risk purposes the advisor must document the responses to the questions asked throughout the conversation. The notes taken during these advisor conversations with customers tend to be hand typed by the Bank advisor into free form text fields in the Bank's CRM system. This data may be very valuable for predicting customer behaviour as the customer may give some indication of his life circumstances or intent to buy products in the near future. For example, the customer may explain that the reason they want to have a conversation about their finances is that they are getting married in six months. Where a customer is getting married this might trigger the need to buy a new home or the requirement for a personal loan to fund the wedding.

This data is not easily consumable for featuring engineering and requires considerable data manipulation in order to pre-process it for amalgamation in an ABT. In this research, historical customer conversations have been analysed and the features embedded in the ABT. In order to develop the features, the data must first be converted into a suitable representation. The first step was to parse the data from the notes into individual words. Certain data cleansing and filtering methods were then applied, for example, all words were converted to lowercase, all punctuation was removed, all numbers were removed and *stop words* were removed. The data was then represented in a bag of words representation which is the most convenient form to represent this data. In a bag of words representation, a dictionary of all the possible words that could occur in all text examples is generated and each text example is represented as a vector indicating the presence or absence of words from the dictionary.

Table 3.3 illustrates a simple bag of words representation for four examples and a dictionary of seven key words. The columns indicate the presence or absence of a word in a specific document. For example, example 1 contains the words abroad, farm, and married; while example 2 contains the words married and young. Figure 3.2 below visually represents in a word cloud certain words found in a sample of records from the dataset.

ID	abroad	account	finance	farm	married	short	young
Example 1	1	0	0	1	1	0	0
Example 2	0	1	1	0	1	0	1
Example 3	1	0	0	0	0	0	0
Example 4	0	0	0	0	1	0	1

Table 3.3: Example of data from text analysis

Figure 3.2: Word cloud of high frequency words from text analysis

A sample of features built from the unstructured notes fields can be seen in Table 3.4.

Feature Name	Feature Description
Number_Dependents	Number of dependents noted
Marital_Status	Up to date marital status
Home_Status	Whether the customer said they were renting, looking to buy, already have a home, own their own house etc.
Occupation	Up to date occupation status employed (what company), unemployed, carer etc.

Table 3.4: Sample of features built from unstructured data

The three sets of features from each of the structured, semi-structured and unstructured data sources are then amalgamated into one dataset. The features are then analysed for inclusion in the final modelling process.

3.2.2 Data Analysis

Once the dataset was developed the first review was carried out in relation to the extent of the imbalance in the dataset. The initial dataset has c.1.23m customer records and 3,078 responders, i.e. customers who took out a mortgage loan during the performance window. The figures for the initial raw dataset are outlined below in Figure 3.3 and indicate a response rate of 0.25%. This demonstrates a high level of imbalance in the dataset with the dataset dominated by customers who did not take out the product over the performance window.

Response Rate	Count of Customers	Rate
Responders	3,078	0.25%
Non-Responders	1,227,322	99.75%
Total Customer Base	1,230,400	100.0%

Figure 3.3: First view of imbalance in the dataset

This dataset includes all customers across a wide range of demographics, income bands, location etc. An assessment was carried out to identify certain cohorts of the customer base that could potentially be removed from the modelling dataset. The approach followed was to remove as many customers as possible from the total customer base but remove as few responders as possible. This approach should considerably increase the response rate of the remaining records. An example is identified in Figure 3.4 below; in this example, it can be seen that the majority of responders are in Segments 1, 4 and 6. Therefore a decision was taken to remove Segments 2, 3 and 5 from the modelling dataset.

By removing these three segments the base of customers reduces by 50%, from 1,230,400 622,713, however, only 3% of responders are lost with responders reducing from 3,078 to 2,981. The removal of these customers increases the natural response rate of the modelling dataset by nearly 100% from 0.25% to 0.48% as seen in Figure 3.5 below.

Segment	Non-Responders	Responders	Total	Response Rate	Response Rate / Total
Segment 1	325,883	1,751	327,634	0.53%	213.64%
Segment 2	247,998	14	248,012	0.01%	2.26%
Segment 3	219,792	17	219,809	0.01%	3.09%
Segment 4	200,229	1,019	201,248	0.51%	202.40%
Segment 5	139,800	66	139,866	0.05%	18.86%
Segment 6	93,620	211	93,831	0.22%	89.89%
Total	1,227,322	3,078	1,230,400	0.25%	100.00%

Figure 3.4: Response rates of particular customer segments

Response Rate	Count of Customers	Rate
Responders	2,981	0.5%
Non-Responders	619,732	99.5%
Total	622,713	100.0%

Figure 3.5: Dataset imbalance following first filter application

This exercise was repeated a number of times following a detailed data discovery and bivariate analysis between each of the independent variables and the target variable. Following this filtering exercise, the final dataset volumes and response rate are identified in Figure 3.6 below. The final modelling dataset contains 15% of the starting total customer base while maintaining 73% of responders.

Response Rate	Count of Customers	Rate
Responders	2,240	1.2%
Non-Responders	189,629	98.8%
Total	191,869	100.0%

Figure 3.6: Final dataset imbalance

Although the dataset remains highly imbalanced with a response rate of 1.2%, the response rate is nearly five times higher than the raw dataset of the total customer base seen in Figure 3.3 above.

An additional bi-variate analysis was carried out across a large number of the variables to identify variables with very little differentiation between the two classes i.e. redundant variables. For example, the majority of these customers have active current accounts with money coming in and out of their account so certain current account variables such as Tran_Active_3Mths was flagged as 1, the customer is an active customer, for nearly all of the customers in Figure 3.6. This variable is inherently redundant on that basis. This exercise was repeated across all of the variables in the ABT and redundant variables were removed.

3.3 Software

A number of software applications have been used to perform this research. This section is split into two parts. The first describes the software used for data acquisition, data integration and transformation to build the ABT, while the second part describes the software used to implement the algorithms in each of the three experiments.

Data Integration and Transformation (ABT)

The data described above has been acquired from multiple different data sources. The product and transactional data have been acquired from the Bank's databases including database technologies such as Teradata (EDW), Oracle and Microsoft SQL server. The data has been pulled together in Teradata, integrated and transformed using the SQL programming language. The semi-structured JSON data is stored in Google Big Query a product in the Google Cloud Platform suite. Again, SQL has been used to build certain features within Big Query. Once constructed, these features were then exported and imported into Teradata. As described above, the unstructured data from branch financial advice notes has been manipulated using R, an open source programming language, to parse the data and identify certain key words for inclusion as features in the ABT.

Implementation of the Models

The models used to perform the experiments in this research have been implemented in R and Java. R is an open source statistical programming language with a large variety of statistical and graphical techniques available through R packages. In this research, CBA has been

implemented in R using the arulesCBA R package. Unlike CBA, there is no R or Python implementation of CMAR and SPARCCC available. In order to implement these algorithms, the source code in Java has been obtained and applied directly for experiments two and three outlined in Section 3.5.2 and Section 3.5.3. The Java code has been sourced, compiled and adjusted to be implemented for use in experiments two and three. The Java code for SPARCC has been applied using the WEKA (Waikato Environment for Knowledge Analysis) software[4], a free software developed in Java that is licensed under the GNU General Public License. The software Apache Ant [5] has been used to build and compile the SPARCCC Java code. Ant is a Java based build tool created as part of the Apache open-source project.

3.4 Benchmark Models

3.4.1 Experiment 1 - Conditional Inference Trees

The conditional inference tree model will be implemented using the R statistical programming tool. Specifically, within R, the library used to run the model is the 'party' library using the 'ctree' model.

A number of sampling methodologies will be tested during the experiment including, oversampling, SMOTE and undersampling to determine which approach generates the best classification performance.

In implementing the conditional inference tree algorithm there are a number of parameters that can be tuned. The parameters for tuning are predominately focused on tree pruning to prevent overfitting. These parameters will be tested to identify the appropriate level of pruning to prevent the model overfitting on the training data. Examples of these parameters include minimum criterion and max depth. Minimum criterion is a parameter setting that determines a test statistic that must be exceeded for the tree to perform another split, the parameter is set to 1 minus the parameter setting so for example if testing a P-value of 0.05 the parameter would be set at mincriterion = 0.95. Max depth reflects the maximum number of splits the model can perform, for example, a parameter setting of maxdepth = 10, means the model cannot perform more than 10 tree splits.

[4] https://www.cs.waikato.ac.nz/ml/weka/
[5] http://ant.apache.org/

3.4.2 Experiment 2 - Random Forests

The random forests model will be implemented using the R statistical programming tool. Specifically, within R, the library used to run the model is the randomForest library. There are a smaller number of parameters that can be tuned when implementing the random forests model. The two key parameters are the number of trees to grow, parameter ntree in R and the number of independent variables randomly sampled as candidates at each split, parameter mtry in R. In order to help determine the right value for mtry, it is possible to use an independent algorithm to identify the optimum value of mtry which minimises the OOB error rate. In R the tuneRF library is used to estimate the optimal mtry value.

3.5 Classification Using Association Rule Models

The aim of the three experiments outlined below is to evaluate whether classification models using association rules can outperform the benchmark classification approaches discussed above in an Irish retail banking context. In this research, three experiments will be completed and the results of each experiment compared and contrasted.

Experiment set up

In order to run the model, the ABT will be split into two files, a train file and a test file. The data will be split 50% train and 50% test using stratified sampling. Where oversampling is applied the oversampling will be applied to the train dataset only. It is important to split the dataset before performing any oversampling otherwise there is a risk that similar or identical records synthetically created appear in the train and testing datasets causing the model to overfit and not generalise well beyond the dataset. A number of different sampling techniques will be applied including, oversampling, SMOTE and undersampling to determine which approach generates the best classification performance.

3.5.1 Experiment 3 - CBA

In experiment 3, the CBA association rule algorithm will be implemented to predict mortgage sales and the performance of the model compared with the performance of conditional

inference trees, CMAR, SPARCCC and random forests. The results will help to determine whether CBA should be used alongside traditional supervised classification methods to make predictions in an Irish retail banking context.

The CBA algorithm has been implemented in the R statistical programming language using the arulesCBA package[6]. The CBA algorithm requires the user to define the minimum support and confidence levels in the rule generation phase. This requires an amount of trial and error in order to identify a range of levels that produce the highest level of classification accuracy.

Experiment 3 will be implemented in line with Section 3.4.1 where a number of different sampling techniques will be applied including, oversampling, SMOTE and undersampling to determine which approach generates the best classification performance.

The results of the model are presented using a confusion matrix and the key evaluation metrics calculated using the same output.

3.5.2 Experiment 4 - CMAR

In experiment 4, the CMAR association rule algorithm is applied to predict mortgage sales and the performance of the model compared with the performance of conditional inference trees, CBA, SPARCCC and random forests.

As described in section 2.4.1, the CMAR algorithm uses a three-stage approach to classification using association rules. The first step is to generate the CARs using the support confidence framework. In this framework, the user must define the relevant support and confidence metrics. The second stage is to prune the rules generated and the third step is to use the rules generated to make classifications.

[6] Package 'arulesCBA' - CRAN.R-project.org

In order to implement the CMAR model, there is an amount of data pre-processing required. The dataset must be in a particular format. A small example of how the dataset must be set up is shown below in Table 3.5.

1 2 3 6

1 4 5 7

1 3 4 6

1 2 6

1 2 3 4 5 7

Table 3.5: Example data following data pre-processing for CMAR

Attribute numbers are ordered sequentially commencing with the number 1 including the class attributes which should follow on from the last attribute number as in the above example. In the example above, the last variable is the class variable and has been transformed from 0/1 to 6/7.

In order to transform the original ABT dataset into a format that could be used by CMAR, a discretisation and normalisation process needs to be applied. Discretisation is the process of converting a continuous variable into a number of sub ranges and then assigning an integer value to each of those subranges. Normalisation is the process of transforming nominal variables in a list of unique integer values. There are a number of approaches to class dependent discretisation as described in Section 2.6. The numerous approaches can generally be categorised according to three differentiating factors 1) supervised vs. unsupervised 2) bottom up vs. top down and 3) direct vs. in-direct. The approach used for CMAR discretisation in this research is a supervised, bottom up and direct method.

A sample output of a dataset pre and post discretisation and normalisation is presented below in Figure 3.7 and Figure 3.8.

This dataset has 4 attributes, colour, average size, age and the binary class.

red	25.6	56	1
green	33.3	1	1
green	2.5	23	0
blue	67.2	111	1
red	29	34	0
yellow	99.5	78	1
yellow	10.2	23	1
yellow	9.9	30	0
blue	67	47	0
red	41.8	99	1

Figure 3.7: Dataset prior to discretisation

This would be discretised and normalised as follows:

3	5	9	13
2	5	8	13
2	5	8	12
1	7	11	13
3	5	9	12
4	7	10	13
4	5	8	13
4	5	8	12
1	7	9	12
3	6	11	13

Figure 3.8: Dataset post discretisation

In this example, the dataset is now presented by 13 binary valued attributes where the class attributed is the last column 12/13 having replaced the original binary input 0/1. Figure 3.9 below shows the GUI for the discretisation and normalisation tool used[7]. The tool takes an input schema, the input data source and discretises and normalises the dataset for use in model development.

[7] https://cgi.csc.liv.ac.uk/~frans/KDD/Software/

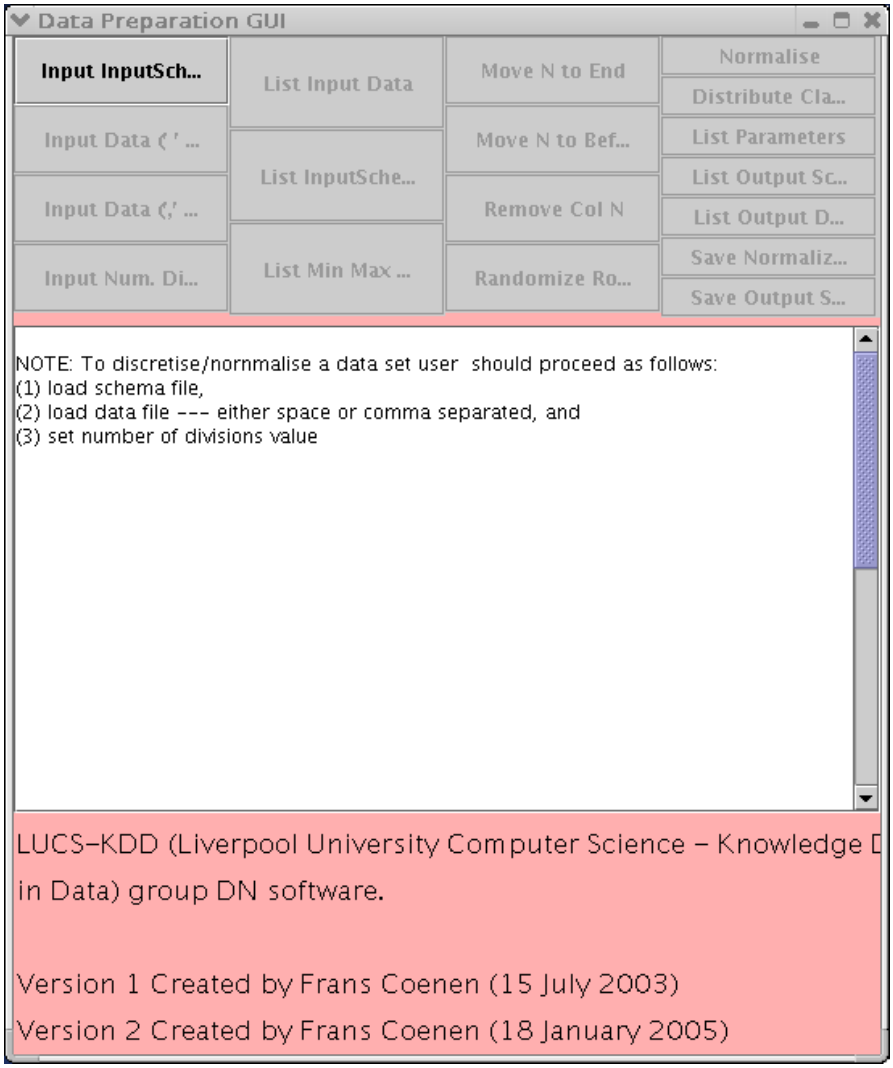


Figure 3.9: GUI for data discretisation and normalisation software

The algorithms applied for CMAR were sourced from the University of Liverpool "LUCS-KDD IMPLEMENTATIONS OF THE CMAR ALGORITHM"[8]. The two algorithms applied were the ClassCMAR_2file_App.java and the ClassCMAR_App10.java. The second algorithm ClassCMAR_App10.java includes 10-fold cross-validation while the first uses two distinct training and test datasets. The CMAR_2 algorithm requires five parameters to be completed, -F training filename, -T test set filename, -N number of classes, -S minimum support threshold and -C minimum confidence threshold. For example, the implementation of the algorithm with a support of 5% and confidence of 80% using the train and test data would be implemented as follows: java ClassCMAR_2file_App -Ftrain.txt -Ttest.txt -S5 -C80 -N2.

[8] https://cgi.csc.liv.ac.uk/~frans/KDD/Software/

The ClassCMAR_2file_App model outputs the accuracy and AUC. The ClassCMAR_App10 outputs the average accuracy and average AUC across the 10-fold cross-validation.

There are certain limitations in the implementation of CMAR compared with the other models. The Java software implementation described above does not provide the confusion matrix for the results instead it simply provides the metrics described above.

3.5.3 Experiment 5 - SPARCCC

In experiment 5, the SPARCCC association rule algorithm will be applied to predict mortgage sales and the performance of the model will be compared with the performance of conditional inference trees, CMAR, SPARCCC and random forests. Experiment 5, will look to evaluate whether the SPARCCC model performs better given the imbalanced nature of the dataset.

The SPARCCC algorithm has been sourced from Liu et al. 2010[9] where the code was made available following the publication. The algorithm has been compiled in Java and then loaded into the Weka software[10]. Weka is a Java based tool for running data mining algorithms which facilitated the loading of the SPARCCC algorithm. The software has been compiled by running the authors' build.xml file using the Apache Ant software. Figure 3.10 below shows the Weka GUI where data can be loaded, visualised and manipulated and Figure 3.11 shows the parameters of the SPARCCC algorithm that can be tuned in advance of running the algorithm.

[9] https://sites.google.com/site/weiliusite/
10 https://www.cs.waikato.ac.nz/ml/weka/downloading.html

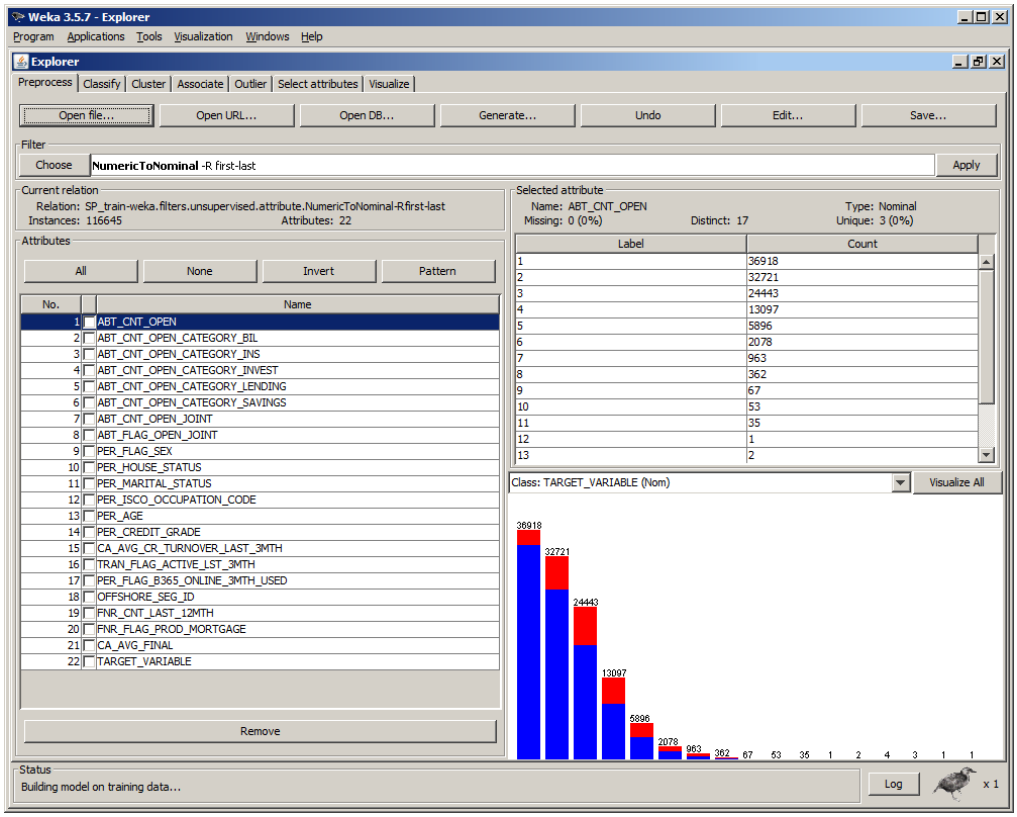


Figure 3.10: GUI for WEKA used to run SPARCCC

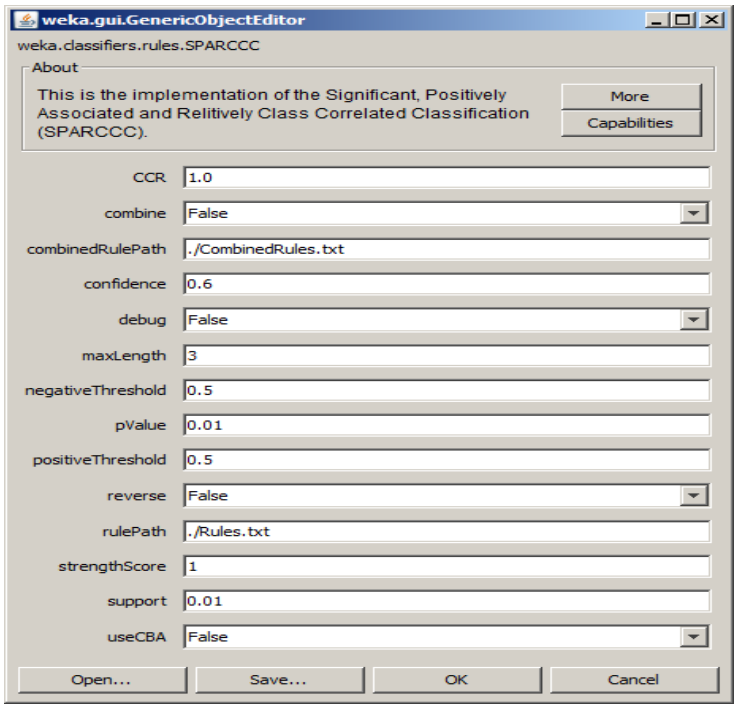


Figure 3.11: Parameter setting in WEKA for SPARCCC

Similar to CMAR the SPARCCC algorithm cannot handle numeric data, therefore, a discretisation process has been applied to variables such as customer age and transactional spend to be suitable for use in the model.

The experiment will be designed in line with Section 3.4.1 where a number of different sampling techniques will be applied including oversampling, SMOTE and undersampling to determine which approach generates the best classification performance.

3.6 Model Evaluation

In line with previous discussions on imbalanced datasets in Section 2.5 and model evaluation metrics in Section 2.9, the models will be evaluated using a number of measures. Given the dataset for use in this research is highly imbalanced the accuracy performance metric may not be appropriate. In these experiments, if the model predicted that all records belong to the majority class the accuracy of the model would be high but it would provide little value in differentiating between the two classes.

In this research the accuracy performance metric will be avoided and a number of alternative model evaluation metrics will be used for model evaluation. The sensitivity and specificity of the models will be evaluated and these metrics will also be combined to give more balanced performance metrics. Two such metrics are balanced accuracy and the F1-score. As described above, balanced accuracy is particularly useful for imbalanced datasets as it measures the ability of the model to correctly identify both classes and is less subject to the potential bias of the majority class. The F1-score is the harmonic mean of sensitivity and specificity where an F1-score reaches its best value at 1 (perfect sensitivity and specificity) and worst at 0. No one measure is suitable to assess the performance of any model, therefore it important to assess each of the models across a number of different performance metrics.

3.7 Conclusion

This chapter outlined the design and methodologies for the five experiments that will be carried out as part of this research and presented in Chapter 4.

The chapter has outlined what data will be collected and how the data will be acquired and integrated to build an ABT for model development. The imbalanced nature of the dataset has been presented together with a number of data filtering approaches that have been applied to somewhat address this imbalance. The various pieces of software required to extract and transform the structured, semi-structured and unstructured data sources have been presented.

The experiment design for each of the five experiments has been explained in detail, the two benchmark models and the three associative classifiers CBA, CMAR and SPARCCC. For each of the experiments the methodology, the key parameters that need to be tuned and the approach to evaluating the results has been clearly outlined.

Chapter 4 will detail the implementation of the five experiments designed in Chapter 3 and the results of each of the experiments.

4. IMPLEMENTATION AND RESULTS

4.1 Introduction

This chapter presents the implementation of the experiments designed in Chapter 3. The purpose of the experiments in this research is to test whether classification models developed using association rules can outperform traditional supervised classification models in predicting mortgage loan sales in an Irish retail banking context. Two benchmark models have been developed to support the evaluation of the three experiments using associative classifiers.

This chapter is divided into 4 sections, section 2 outlines the implementation approach and results for the two benchmark models, conditional inference trees and random forests, section 3 outlines the results of experiment 3 using CBA, section 4 outlines the results of experiment 4 using CMAR and section 5 outlines the results of experiment 5 using SPARCCC.

4.2 Benchmark models

Two traditional classifications models were implemented to provide benchmark classification performance metrics for comparison with the three association rule classification models. The two benchmark models implemented were conditional inference trees and random forests. A more detailed description of the algorithms can be found in Section 2.7. Both benchmark models were implemented using the R statistical programming language.

Experiment 1 - Conditional Inference Trees

Outlined below are the results of the conditional inference trees implementation. Stratified sampling was applied to split the dataset 50% training dataset and 50% test dataset. Simple oversampling was applied to the minority class to rebalance the dataset close to 80% zeros and 20% ones. The results presented below are the classification performance on the test dataset. Table 4.1 below presents the confusion matrix for the predictions made on the test dataset using the conditional inference trees. Table 4.2 presents the key evaluation metrics used to assess the performance of the model. The results give some indication of the ability of

the larger class to distort the accuracy metric. In this model, the total model accuracy is 93.1%., however, the results for recall and precision are much lower at 36.7% and 6.5% respectively. The F1-score which is the harmonic mean of recall and precision gives a balanced perspective of these two measures at 11.1%. The balanced accuracy at 65% compensates for the poorer performance in classifying the minority class.

	Actual Class	
Predicted Class	**1**	**0**
1	411	5,867
0	709	88,947

Table 4.1: Confusion Matrix for Conditional Inference Trees

Recall	36.7%
Precision	6.5%
F1-score	11.1%
Accuracy Class 1 (Recall)	36.7%
Accuracy Class 0	93.8%
Balanced Accuracy	65.3%
Accuracy	93.1%
AUC	72.7%

Table 4.2: Key Evaluation Metrics for Conditional Inference Trees

The ROC curve for the conditional inference tree model is shown below in Figure 4.1 and the model AUC calculated is 72.7%.

The confusion matrix shows the difficulty this model has dealing with the imbalanced data. The model has a high volume of false positives, 5,867 records classified as 1 when in fact the true value is zero. The model also misclassifies a high percentage of true value 1s as 0s.

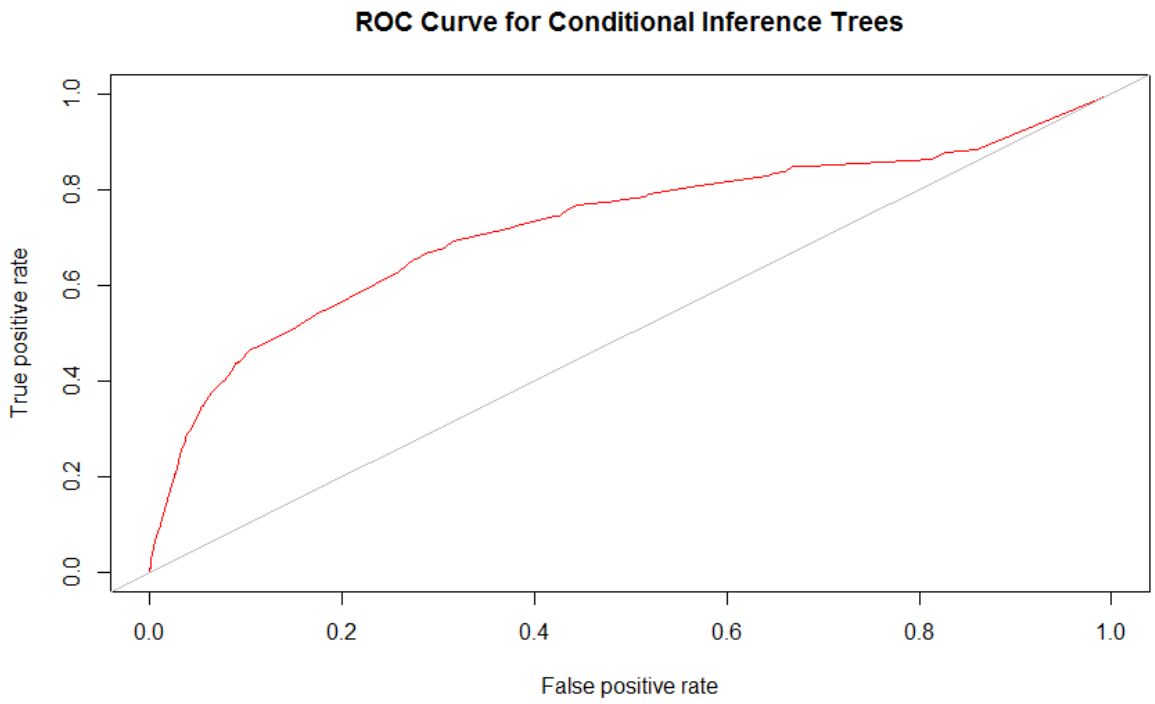


Figure 4.1: ROC for Conditional Inference Trees

Figure 4.2 below ranks the variables in terms of importance to the model. The most important variable in the model receives a score of 100 and all other variables are scored relative to this variable. Variables CA_AVG_CR_TURNOVER_LAST_3M, which is a proxy for an individual's salary, number of open products (ABT_CNT_OPEN) and the number of open savings products the customer has (ABT_CNT_OPEN_CATEGORY_SAVINGS) are the three most important variables. These variables all appear to be intuitive outcomes for a mortgage prediction model as customers with higher salaries and more savings are more likely to be in the market for a mortgage loan.

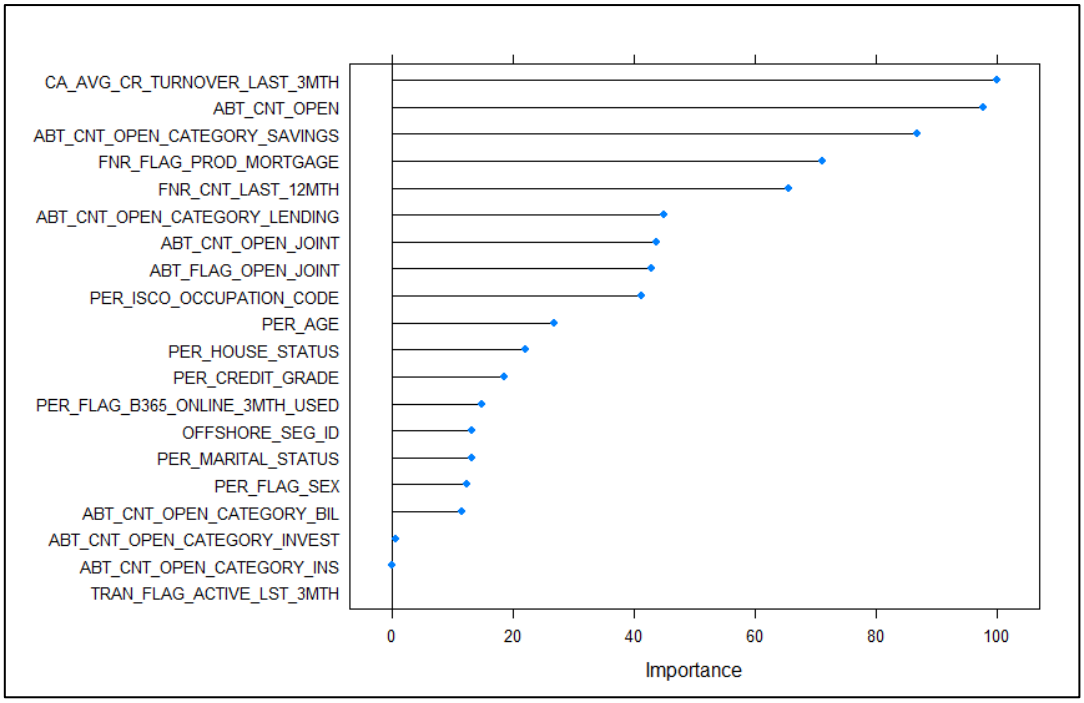


Figure 4.2: Variable importance for Conditional Inference Trees

Experiment 2 - Random Forests

The implementation of random forests is in line with the implementation approach for conditional inference trees above. The data has been split 50% for training purposes and 50% for test purposes and oversampling of the minority class has been applied to rebalance the training dataset closer to 20% in the minority class and 80% in the majority class. The results presented below are the classification performance on the test dataset. Table 4.3 below presents the confusion matrix for the predictions made on the test dataset using the random forests algorithm. Table 4.4 presents the key evaluation metrics used to assess the performance of the model. The performance of the random forest is similar to the performance of the conditional inference trees and has similar failings in terms of misclassification. Although the model accuracy is high at 96.1% the minority class metrics are considerably lower with a recall of 28.8% and precision of 9.9%. Similar to the conditional inference trees model these low values reflect a high number of false positives and a high percentage of the minority class incorrectly classified as no sale.

	Actual Class	
Predicted Class	**1**	**0**
1	323	2,942
0	797	91,872

Table 4.3: Confusion Matrix for Random Forests

Recall	28.8%
Precision	9.9%
F1-score	14.7%
Accuracy Class 1 (Recall)	28.8%
Accuracy Class 0	96.9%
Balanced Accuracy	62.9%
Accuracy	96.1%
AUC	80.5%

Table 4.4: Key Evaluation Metrics for Random Forests

Figure 4.3 below plots the ROC curve for both the conditional inference tree model and the random forest model. The blue line represents the random forest model and based on this chart indicates the classification performance of the random forest model is better than the conditional inference tree model. The random forest model has lower type 1 errors than the conditional inference trees resulting in a higher precision value but has a lower balanced accuracy due to lower accuracy on the minority class.

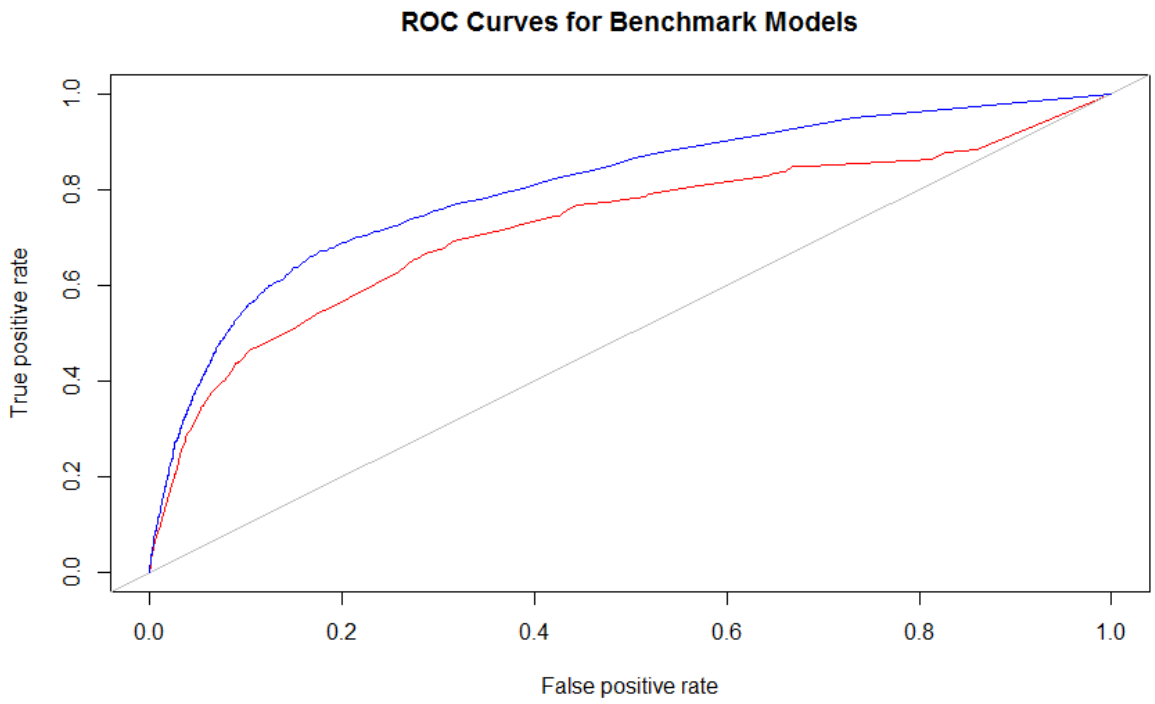


Figure 4.3: ROC comparison for Conditional Inference Trees and Random Forests

Figure 4.4 below ranks the variables in terms of importance to the random forest model. The figure shows the mean decrease in Gini coefficient. The mean decrease in Gini coefficient is a measure of how each variable contributes to the homogeneity of the nodes and leaves in the random forest model developed. Each time a particular variable is used to split a node, the Gini coefficient for the child nodes are calculated and compared to that of the original node. The Gini coefficient is a measure of homogeneity from 0 (homogeneous) to 1 (heterogeneous). The changes in Gini are summed for each variable and normalized at the end of the calculation. Variables that result in nodes with higher purity have a higher decrease in Gini coefficient. In Figure 4.4, the trend in variable importance is similar to that of the conditional inference tree model, however, the actual ranking of variables varies slightly. The CA_AVG_CR_TURNOVER_LAST_3M, which is a proxy for an individual's salary, is still the top variable in terms of importance, however, in the random forest model the second most important variable is age and the third most important variable is FNR_FLAG_PROD_MORTGAGE, which indicates if the customer came to speak to an advisor and indicated a long-term interest in getting a mortgage.

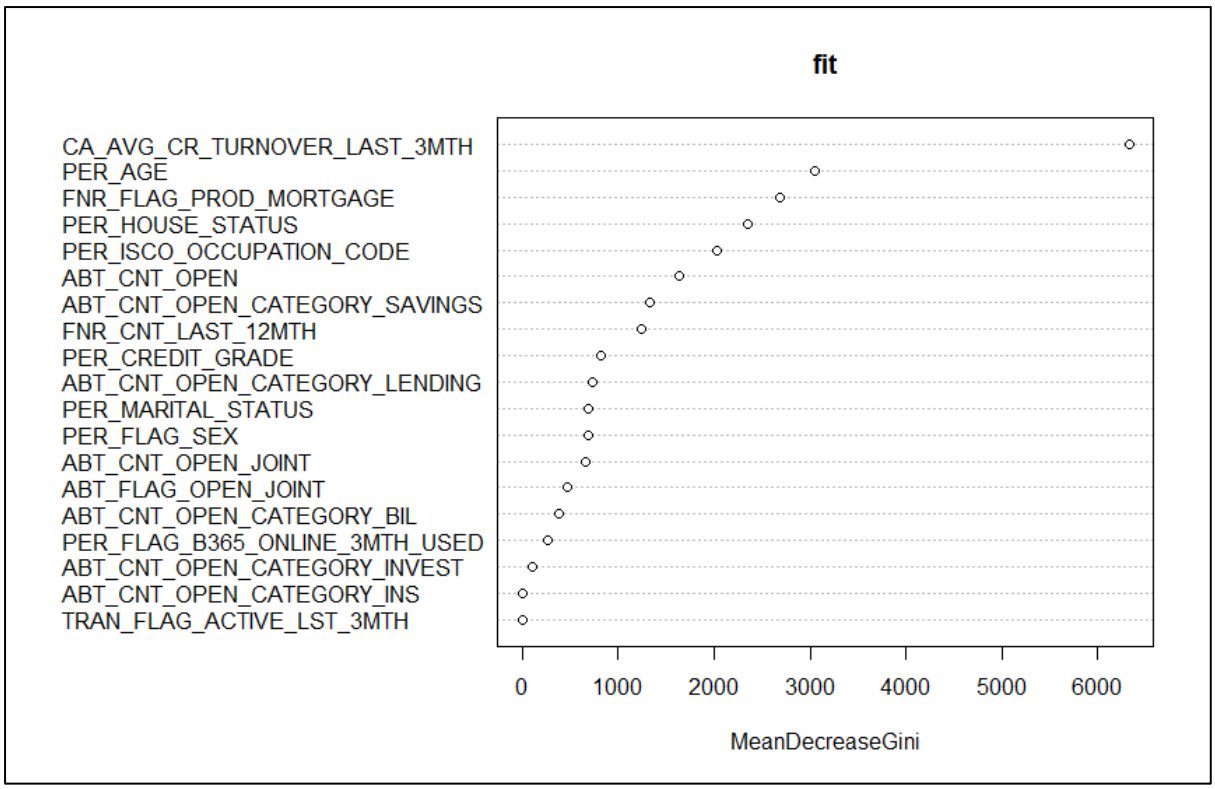


Figure 4.4: Variable Importance for Random Forests

A comparison of the performance of the two benchmark models in table 4.5 below shows that each of the two benchmark models has certain metrics where it outperforms the other model. For example, the conditional inference trees outperform the random forests on recall and balanced accuracy but is lower on F1-score and AUC.

Evaluation Metrics	CI Trees	Random Forests
Recall	36.7%	28.8%
Precision	6.5%	9.9%
F1-score	11.1%	14.7%
Accuracy Class 1 (Recall)	36.7%	28.8%
Accuracy Class 0	93.8%	96.9%
Balanced Accuracy	65.3%	62.9%
Accuracy	93.1%	96.1%
AUC	72.7%	80.5%

Table 4.5: Comparison between CI Trees and Random Forests across performance metrics

4.3 Experiment 3 - CBA Algorithm

In line with the implementation approach applied in the benchmark models, the dataset was first sampled 50% for the training dataset and 50% for the test dataset. The dataset was oversampled bringing the minority class close to 20% of records in the training dataset. Certain continuous variables such as age and income variables were discretised into categorical variables suitable for association rule mining.

The class association rules were created based on the training dataset and used to make predictions on the test dataset. In order to generate the rules, the minimum support level was set to 1% and the minimum confidence level set to 40%. A number of varying support and confidence levels were applied and tested to identify a range of support and confidence levels producing the highest level of classification accuracy. The parameter values for support and confidence identified above were chosen given they produced the optimal results. Figure 4.5 below shows a subset of rules where the right-hand side of the rule is the target variable minority class. The figure shows the left-hand side of the rule, the right-hand side and the support, confidence and lift metrics. The final column shows the count of appearances of each particular rule in the training dataset.

```
    lhs                                    rhs                    support confidence     lift count
[1] {PER_HOUSE_STATUS=C,
     FNR_CNT_LAST_12MTH=1,
     CA_AVG_DISC=[5645,1068771]}       => {TARGET_VARIABLE=1} 0.01117189  0.7048913 3.845599  1297
[2] {PER_HOUSE_STATUS=C,
     PER_FLAG_B365_ONLINE_3MTH_USED=1,
     FNR_CNT_LAST_12MTH=1,
     CA_AVG_DISC=[5645,1068771]}       => {TARGET_VARIABLE=1} 0.01075843  0.7008979 3.823813  1249
[3] {ABT_CNT_OPEN_CATEGORY_LENDING=1,
     FNR_CNT_LAST_12MTH=1,
     CA_AVG_DISC=[5645,1068771]}       => {TARGET_VARIABLE=1} 0.01034498  0.6668517 3.638071  1201
[4] {ABT_CNT_OPEN_CATEGORY_LENDING=1,
     PER_FLAG_B365_ONLINE_3MTH_USED=1,
     FNR_CNT_LAST_12MTH=1,
     CA_AVG_DISC=[5645,1068771]}       => {TARGET_VARIABLE=1} 0.01006934  0.6649602 3.627752  1169
[5] {PER_FLAG_SEX=1,
     FNR_CNT_LAST_12MTH=1,
     CA_AVG_DISC=[5645,1068771]}       => {TARGET_VARIABLE=1} 0.01126664  0.6443350 3.515229  1308
```

Figure 4.5: Top ranking rules for CBA implementation

After the classification association rules are built the rules can be sorted by a number of varying metrics including support, confidence and lift. Once the rules have been sorted according to one of these metrics they can be used to make predictions on the test dataset. In

this experiment, the rules have been sorted by both confidence and lift and the classification accuracy compared.

The results presented below are the classification performance on the test dataset. Table 4.6 below presents the confusion matrix for the predictions made on the test dataset using the CBA algorithm. Table 4.7 presents the key evaluation metrics used to assess the performance of the model. Although the CBA model performs better than the two benchmark models in classifying the minority class, the model predicts a high volume of 1s where in fact the actual value is 0. The high volume of type 1 errors is therefore reflected in a very low precision metric of 2.7%.

	Actual	
Predicted	**1**	**0**
1	687	24,383
0	433	70,431

Table 4.6: Confusion Matrix for CBA

Recall	61.3%
Precision	2.7%
F1-score	5.2%
Accuracy Class 1 (Recall)	61.3%
Accuracy Class 0	74.3%
Balanced Accuracy	67.8%
Accuracy	74.1%

Table 4.7: Key Evaluation Metrics for CBA

A comparison of CBA against the two benchmark models confirms the model evaluation above. The CBA algorithm has produced a high volume of type 1 errors or false positives. The balanced accuracy of the CBA model is the highest of the three models but the low precision value is a major concern about the model's predictive ability.

Evaluation Metrics	CI Trees	Random Forests	CBA
Recall	36.7%	28.8%	61.3%
Precision	6.5%	9.9%	2.7%
F1-score	11.1%	14.7%	5.2%
Accuracy Class 1 (Recall)	36.7%	28.8%	61.3%
Accuracy Class 0	93.8%	96.9%	74.3%
Balanced Accuracy	65.3%	62.9%	67.8%
Accuracy	93.1%	96.1%	74.1%
AUC	72.7%	80.5%	

Table 4.8: Comparison between CI Trees, Random Forests and CBA across performance metrics

4.4 Experiment 4 - CMAR

A number of various approaches were applied in the CMAR experiment, however, none of the methods returned positive results. For each of these approaches, the data was discretised and normalised in line with the methodology set out in Section 3.5.2.

The first approach applied was 10-fold cross-validation on the full dataset with no treatment for the imbalanced nature of the dataset. Here the model is trained on 90% of the data and tested on a holdout sample of 10%, and this is repeated 10 times. The CMAR algorithm was run with parameters of support 1% and confidence 40%. The results of each model are then aggregated together. This method did not work and the model did not predict any of the minority class in the final class predictions. The model predictions results in an AUC of 0.5, indicating the model was of no value.

The second approach applied was to undersample the majority class and then apply 10-fold cross-validation. In this undersampling approach, records with the majority class were removed at random from the dataset until the minority class represents 2.4%, a 50% reduction in the majority class. The CMAR algorithm was run with parameters of support 1% and confidence 40%. This model was not able to correctly predict any of the minority class values and the resulting AUC was 0.5.

In the third approach, in line with the implementation approach applied in the CBA experiment described above, the dataset was first sampled 50% for the training dataset and 50% for the test dataset. The dataset was oversampled bringing the minority class close to 20% of records in the training dataset. The CMAR algorithm was run with parameters of support 1% and confidence 40%. In this approach, the AUC reached 0.54 but this is still a model with very poor performance. Figure 4.6 below presents the top-ranking rules from the CMAR implementation.

(#) Ante	->	Cons	confidence %	(Sup. Rule, Sup. Ante, Sup. Cons.)
(1) {70 71 20 79 50}	->	{91}	100.00%	, (2053.0, 2053.0, 94795.0)
(2) {1 76 67}	->	{91}	100.00%	, (1823.0, 1823.0, 94795.0)
(3) {19 1 76 67}	->	{91}	100.00%	, (1823.0, 1823.0, 94795.0)
(4) {25 1 76 67}	->	{91}	100.00%	, (1823.0, 1823.0, 94795.0)
(5) {19 25 1 76 67}	->	{91}	100.00%	, (1823.0, 1823.0, 94795.0)
(6) {70 38 47 79 50}	->	{91}	100.00%	, (1820.0, 1820.0, 94795.0)
(7) {71 38 47 79 50}	->	{91}	100.00%	, (1764.0, 1764.0, 94795.0)
(8) {70 71 58 79 50}	->	{91}	100.00%	, (1668.0, 1668.0, 94795.0)
(9) {55 81}	->	{91}	100.00%	, (1633.0, 1633.0, 94795.0)
(10) {71 55 81}	->	{91}	100.00%	, (1611.0, 1611.0, 94795.0)
(11) {38 55 81}	->	{91}	100.00%	, (1602.0, 1602.0, 94795.0)
(12) {71 38 69 81}	->	{91}	100.00%	, (1596.0, 1596.0, 94795.0)
(13) {71 38 55 81}	->	{91}	100.00%	, (1580.0, 1580.0, 94795.0)
(14) {71 38 20 79 50}	->	{91}	100.00%	, (1493.0, 1493.0, 94795.0)
(15) {70 25 47 58 50}	->	{91}	100.00%	, (1448.0, 1448.0, 94795.0)
(16) {71 25 47 2 50}	->	{91}	100.00%	, (1448.0, 1448.0, 94795.0)
(17) {71 38 19 69 81}	->	{91}	100.00%	, (1444.0, 1444.0, 94795.0)
(18) {70 25 47 2 50}	->	{91}	100.00%	, (1426.0, 1426.0, 94795.0)
(19) {38 69 82}	->	{91}	100.00%	, (1423.0, 1423.0, 94795.0)
(20) {70 71 2 80 50}	->	{91}	100.00%	, (1398.0, 1398.0, 94795.0)

Figure 4.6: Top ranking rules for CMAR implementation

```
----------Average Accuracy = 82.02SD Accuracy          = 0.11Average AUC value = 0.5339Ave. # Freq. Sets      = 1569128.0
Generation time = 31264.66 seconds (521.08 mins)
```

Figure 4.7: Experiment Results CMAR

An alternative approach would have been to apply oversampling during the cross-validation implementation. This would involve oversampling only in the 90% sampled for model training and the remaining 10% test remains unbalanced. If the cross-validation approach is applied to the full dataset where oversampling has already been applied there is a risk of overfitting as synthetically created records could appear in both the test and training datasets. However, there were certain limitations in the implementation of this algorithm. The

implementation did not allow oversampling the 90% training sample while leaving the 10% test sample untreated.

4.5 Experiment 5 - SPARCCC

In line with the implementation approach applied in the experiments described above, the dataset was first sampled 50% for the training dataset and 50% for the test dataset. The dataset was oversampled bringing the minority class close to 20% of records in the training dataset. The SPARCCC algorithm does not accept numeric values so variables were converted to nominal values or discretised into a number of bins.

The class association rules were created based on the training dataset and then used to make predictions on the test dataset. In order to generate the rules, the minimum support level was set to 1% and the CCR, described in Section 2.5, set to a value of 1. A number of varying support and CCR levels were applied and tested to identify a range of support and confidence levels producing the highest level of classification accuracy.

A subset of the rules generated during the model training phase is presented below in Figure 4.8. In addition to the outputs from the previous rule generation models, the SPARCCC CCR value can be seen, the furthermost right value. For example, the variable FNR FNR_CNT_LAST_12MTH=2 -> 1 generates a CCR value of 8.8, the highest in this subset of rules where the right-hand side of the rule is the minority class of the target variable.

Rules	Predict Value	Confidence	Support	Strength Score	Sensitivity	Specificity	F-measure	pValue	CCR
ABT_CNT_OPEN=5 -> 1	TARGET_VARIABLE=1	40%	2%	1.12	10.77	96.11	19.37	0	2.77
ABT_CNT_OPEN=6 -> 1	TARGET_VARIABLE=1	44%	1%	1.4	4.42	98.62	8.46	0	3.19
ABT_CNT_OPEN_CATEGORY_SAVINGS=3 -> 1	TARGET_VARIABLE=1	44%	1%	1.4	7.27	97.73	13.53	0	3.2
ABT_CNT_OPEN_JOINT=2 -> 1	TARGET_VARIABLE=1	44%	1%	1.39	7.18	97.74	13.37	0	3.17
OFFSHORE_SEG_ID=218 -> 1	TARGET_VARIABLE=1	40%	1%	1.11	2.74	99.01	5.33	0	2.75
FNR_CNT_LAST_12MTH=1 -> 1	TARGET_VARIABLE=1	43%	6%	1.35	29.96	90.38	45	0	3.11
FNR_CNT_LAST_12MTH=2 -> 1	TARGET_VARIABLE=1	68%	1%	6.01	3.2	99.64	6.19	0	8.8
FNR_FLAG_PROD_MORTGAGE=1 -> 1	TARGET_VARIABLE=1	61%	6%	3.97	30.36	95.31	46.05	0	6.47

Figure 4.8: Subset of the rules from SPARCCC training

The results presented below are the classification performance on the test dataset. Table 4.9 below presents the confusion matrix for the predictions made on the test dataset using the SPARCCC algorithm. Table 4.10 presents the key evaluation metrics used to assess the performance of the model. Similar to the results in CBA, the SPARCCC model generates a large number of false positives with 10,822 incorrectly classified values. The number of false positives in the SPARCCC model is still significantly lower than the number of false positives generated by CBA which is reflected in a higher precision and F1-score. The SPARCCC model performs worse than the CBA model in classifying the minority class, this is reflected in a lower recall value for SPARCCC of 44.4% vs. 61.3% for CBA.

	Actual	
Predicted	**1**	**0**
1	480	10,822
0	602	84,030

Table 4.9: Confusion Matrix for SPARCCC

Recall	44.4%
Precision	4.2%
F1–score	7.8%
Accuracy Class 1 (Recall)	44.4%
Accuracy Class 0	88.6%
Balanced Accuracy	66.5%
Accuracy	88.1%
AUC	66.5%

Table 4.10: Key Evaluation Metrics for SPARCCC

A comparison of the models in shown Table 4.11.

Evaluation Metrics	CI Trees	Random Forests	CBA	SPARCCCC
Recall	36.7%	28.8%	61.3%	44.4%
Precision	6.5%	9.9%	2.7%	4.2%
F1–score	11.1%	14.7%	5.2%	7.8%
Accuracy Class 1 (Recall)	36.7%	28.8%	61.3%	44.4%
Accuracy Class 0	93.8%	96.9%	74.3%	88.6%
Balanced Accuracy	65.3%	62.9%	67.8%	66.5%
Accuracy	93.1%	96.1%	74.1%	88.1%
AUC	72.7%	80.5%		66.5%

Table 4.11: Comparison of performance metric across all models

4.6 Conclusion

Chapter 4 has explained in detail how each of the five experiments was implemented and the results of each experiment. For each of the associative classifiers a sample of the top high-quality rules used for classification has been presented. The confusion matrix, performance metrics, variable importance and ROC curve have been provided where applicable.

Chapter 4 provided an initial comparison of the models across the key performance metrics and some insights into the positive and negative outcomes evidenced.

In Chapter 5 the models are evaluated in more detail and an assessment is completed as to how the results compare to the existing body of research. Chapter 5 also details how the results of the experiments support real-world use cases.

5. EVALUATION

5.1 Introduction

This chapter evaluates the results of the experiments carried out in Chapter 4. The objective of this research is to extend the existing body of research on classification using association rules through the completion of a number of experiments and compare the results of the experiments with the existing research in the field. This research is unique as the experiments have been performed on real-world Irish retail banking mortgage data. This chapter is divided into three further sections.

Section 5.2 evaluates in more detail the experiments carried out in Chapter 4, the strengths and weaknesses of the experiments and outlines which model would ultimately be chosen for use in a production environment. This section compares the results of the experiments implemented with previous literature in this area and analyses whether the results of the experiments in this research align or not with the existing body of research.

Section 5.3 outlines how these results support real-world experiments and what role associative classifiers could play in supporting traditional classification models. This section describes additional real-world use cases, for example, supporting data processing activities under the new data regulation GDPR.

Section 5.4 gives a brief evaluation of the various software packages used throughout this research, detailing some of the positives and negatives of each.

5.2 Evaluation of Experiments

This section evaluates in more detail the experiments carried out in Chapter 4. The existing research in this area suggests that given an imbalanced dataset such as the dataset in this research experiment, the SPARCCC algorithm should outperform CBA and CMAR and the SPARCCC algorithm should compare favourably with traditional tree-based supervised learning classification algorithms.

Evaluation Metrics	CI Trees	Random Forests	CBA	SPARCCCC	CMAR
Recall	36.7%	28.8%	61.3%	44.4%	
Precision	6.5%	9.9%	2.7%	4.2%	
F1–score	11.1%	14.7%	5.2%	7.8%	
Accuracy Class 1 (Recall)	36.7%	28.8%	61.3%	44.4%	
Accuracy Class 0	93.8%	96.9%	74.3%	88.6%	
Balanced Accuracy	65.3%	62.9%	67.8%	66.5%	
Accuracy	93.1%	96.1%	74.1%	88.1%	
AUC	72.7%	80.5%		66.5%	53%

Figure 5.1: Top performing model across key performance metrics
(Highlighted in blue is the top performing algorithm for the particular metric)

On review of the implementation results of the five experiments carried out in Chapter 4, Figure 5.1 above, the models that would be implemented in this real-world scenario would be the two benchmark models. In particular, the random forests model performed the best of the two benchmark models and would be the model used in production, in this scenario for use in marketing activities to retail banking customers. Choosing the random forests as the top performing model is based on assessing the performance of the models across a number of balanced performance measures such as AUC, balanced accuracy and F1-score rather than focusing on any single performance evaluation measure. The results of these experiments signify that in this real-world mortgage loan sales prediction experiment *associative classifier* models have not been able to perform at least as well as or better than traditional supervised learning classification models.

The poor performance in the CBA and SPARCCC algorithms is predominately evidenced by the large amount of false positive predictions. In the CBA and SPARCCC experiments, the models have predicted a large volume of records to be a sale when in fact no sale occurred leading to very low precision values. The rules generated from the associative classifiers are not able to adequately identify the relevant patterns in the data.

For CBA, the known weakness with this model as discussed in 2.4, is that the CBA algorithm uses only one high quality rule to make class predictions ranked by rule confidence. The algorithm ignores other rules that may also be high quality but have slightly slower

confidence than the top rule. This single rule approach is a simplified approach and is likely to result in high levels of misclassification as the single rule is used to classify all records that are covered by that rule.

In Li et al. (2001), the authors developed CMAR and tested the results against CBA and C4.5. The authors' results state that CMAR outperforms both C4.5 and CBA in terms of accuracy on 13 or 50% of the 26 UCI[11] datasets used for the research. CMAR is expected to outperform CBA as it uses voting systems across a number of top quality rules rather than simply picking only the top ranked rule. In this research, however, CMAR performed the worst of the models across the five experiments. The explanation for this performance can be attributed to the rule ranking methodology as part of the CMAR implementation.

A review of the CMAR implementation approach shows a bias towards rules with high support as well as high confidence. In the implementation of CBA, although the model is simplistic in that it ranks based on confidence only, rules with low support can be still be ranked high in terms of the ordering of rules. This means for CBA, rules which cover the minority class that have a low support level can still rank highly in the rules ordering process. For CMAR, however, the rule ranking procedure, uses both support and confidence. Ranking rules by using support biases rules towards the majority class given they appear more often in the dataset. This results in the top-ranking rules being dominated by the majority class. This rule ranking implementation resulted in a model similar to defaulting all predictions to the majority class, in this research that classification is 'no sale'.

This rule ranking bias is outlined in Figure 5.2 and Figure 5.3 below. Figure 5.2 below shows the top priority rules from the CMAR implementation. In line with the discretisation approach applied for CMAR described above, Cons {91}, reflects the majority class and Cons {92} reflects the minority class. Figure 5.2 shows that the top-ranking rules all have the consequent {91}, the majority class. Figure 5.3 shows where the minority class rules begin to appear in the rule rank ordering, the first Cons {92} rule appears at rule ranking number 6,990. For context, the CMAR algorithm has created 14,968 high quality rules. This means the first 6,989 rules all default to the majority class. This rule ranking bias is the reason for the poor performance of the CMAR algorithm implementation. The CMAR implementation

[11] https://archive.ics.uci.edu/ml/datasets.html

in this research does not have the flexibility to change the rule ranking process to be driven only by confidence and this is, therefore, a limitation of this research.

(#) Ante	->	Cons	confidence %	(Sup. Rule, Sup. Ante, Sup. Cons.)
(1) {70 71 20 79 50}	->	{91}	100.00%	, (2053.0, 2053.0, 94795.0)
(2) {1 76 67}	->	{91}	100.00%	, (1823.0, 1823.0, 94795.0)
(3) {19 1 76 67}	->	{91}	100.00%	, (1823.0, 1823.0, 94795.0)
(4) {25 1 76 67}	->	{91}	100.00%	, (1823.0, 1823.0, 94795.0)
(5) {19 25 1 76 67}	->	{91}	100.00%	, (1823.0, 1823.0, 94795.0)
(6) {70 38 47 79 50}	->	{91}	100.00%	, (1820.0, 1820.0, 94795.0)
(7) {71 38 47 79 50}	->	{91}	100.00%	, (1764.0, 1764.0, 94795.0)
(8) {70 71 58 79 50}	->	{91}	100.00%	, (1668.0, 1668.0, 94795.0)
(9) {55 81}	->	{91}	100.00%	, (1633.0, 1633.0, 94795.0)
(10) {71 55 81}	->	{91}	100.00%	, (1611.0, 1611.0, 94795.0)
(11) {38 55 81}	->	{91}	100.00%	, (1602.0, 1602.0, 94795.0)
(12) {71 38 69 81}	->	{91}	100.00%	, (1596.0, 1596.0, 94795.0)
(13) {71 38 55 81}	->	{91}	100.00%	, (1580.0, 1580.0, 94795.0)
(14) {71 38 20 79 50}	->	{91}	100.00%	, (1493.0, 1493.0, 94795.0)
(15) {70 25 47 58 50}	->	{91}	100.00%	, (1448.0, 1448.0, 94795.0)
(16) {71 25 47 2 50}	->	{91}	100.00%	, (1448.0, 1448.0, 94795.0)
(17) {71 38 19 69 81}	->	{91}	100.00%	, (1444.0, 1444.0, 94795.0)
(18) {70 25 47 2 50}	->	{91}	100.00%	, (1426.0, 1426.0, 94795.0)
(19) {38 69 82}	->	{91}	100.00%	, (1423.0, 1423.0, 94795.0)
(20) {70 71 2 80 50}	->	{91}	100.00%	, (1398.0, 1398.0, 94795.0)

Figure 5.2: Top ranking rules from CMAR training

(#) Ante	->	Cons	confidence %	(Sup. Rule, Sup. Ante, Sup. Cons.)
(6990) {51 72 7 30}	->	{92}	100.00%	, (106.0, 106.0, 34170.0)
(7549) {26 72 85 62}	->	{92}	100.00%	, (94.0, 94.0, 34170.0)
(8139) {52 84 28 41}	->	{92}	100.00%	, (81.0, 81.0, 34170.0)
(8186) {76 72 27 7}	->	{92}	100.00%	, (80.0, 80.0, 34170.0)
(8531) {52 89 40 65}	->	{92}	100.00%	, (73.0, 73.0, 34170.0)
(8532) {77 72 85 62}	->	{92}	100.00%	, (73.0, 73.0, 34170.0)
(8762) {52 90 69 5}	->	{92}	100.00%	, (69.0, 69.0, 34170.0)
(8823) {76 87 53 5}	->	{92}	100.00%	, (68.0, 68.0, 34170.0)
(8926) {90 67 30}	->	{92}	100.00%	, (66.0, 66.0, 34170.0)
(8941) {20 72 67 7}	->	{92}	100.00%	, (66.0, 66.0, 34170.0)
(8942) {72 90 67 7}	->	{92}	100.00%	, (66.0, 66.0, 34170.0)
(8943) {48 72 28 22}	->	{92}	100.00%	, (66.0, 66.0, 34170.0)
(9052) {77 89 9}	->	{92}	100.00%	, (64.0, 64.0, 34170.0)
(9053) {77 29 9}	->	{92}	100.00%	, (64.0, 64.0, 34170.0)
(9054) {58 39 74}	->	{92}	100.00%	, (64.0, 64.0, 34170.0)
(9077) {72 59 21 29}	->	{92}	100.00%	, (64.0, 64.0, 34170.0)
(9078) {70 77 72 9}	->	{92}	100.00%	, (64.0, 64.0, 34170.0)
(9079) {70 58 39 74}	->	{92}	100.00%	, (64.0, 64.0, 34170.0)
(9080) {19 58 39 74}	->	{92}	100.00%	, (64.0, 64.0, 34170.0)
(9139) {48 63 86 7}	->	{92}	100.00%	, (63.0, 63.0, 34170.0)

Figure 5.3: Subset of minority class rules from CMAR implementation

The results of CBA and CMAR give weight to the concerns raised in Verhein and Chawla (2007) that these models have difficulty dealing with imbalanced datasets. Even following oversampling approaches in the training dataset, the model performance is still poor particularly for certain performance metrics such as precision. In Verhein and Chawla (2007), the authors set out to demonstrate that existing associative classifiers, for example, CBA and CMAR perform poorly on imbalanced data and that their newly developed algorithm SPARCCC is a more suitable model to accurately classify the minority class on such imbalanced datasets.

The authors performed two sets of tests for their classifier SPARCCC. The first test, on balanced datasets, compares the performance of the SPARCCC algorithm with other classifiers CBA, CMAR and C4.5 and the results show little difference in the accuracy of the predictions. The authors state that the benefit of SPARCC for balanced datasets is that the SPARCCC model uses a much smaller search space so is, therefore, less computationally intensive.

The second test focuses on imbalanced datasets where the authors state that the SPARCCC model was 45.8% better than CBA and 26.1% better than CCCS, Complement Class Support, (Arunasalam & Chawla, 2006), another associative classification model. For performance measurement, the authors have used True Positive Rate or recall. In choosing True Positive Rate, the authors recognise that accuracy is not a valid performance metric and instead have decided to use True Positive Rate as their evaluation metric of choice. The concern with the evaluation approach used by Verhein and Chawla (2007) is the use of a single performance metric for evaluation purposes. One metric by itself may not provide the full picture. This is a potential weakness in the results presented by Verhein and Chawla (2007). In this research, if the chosen metric for evaluation was simply recall the final model chosen as the top performing model would have been CBA. Instead in this research evaluation has been carried out across a range of metrics to give a more balanced and comprehensive evaluation.

The expectation when designing the experiments, given the claims made in Verhein and Chawla (2007), was that the SPARCCC algorithm would be the top performing associative classifier and would challenge the performance of the tree based supervised learning approaches. What is concerning about the results in Verhein and Chawla (2007), is the authors removed the performance of the traditional decision tree method C4.5 when

presenting their results on the imbalanced dataset and did not provide any basis behind this decision.

In these experiments, the SPARCCC algorithm did perform the best when comparing the three associative classifiers across a range of balanced measures, balanced accuracy, AUC and F1–score. However, the SPARCCC model is not able to match the performance of the traditional classification approaches. One possible explanation for this is the real-world nature of this dataset compared to the datasets used in Verhein and Chawla (2007). In order to generate imbalanced datasets, the authors undersampled existing UCI datasets and left the majority class intact. The authors undersampled the minority class until the minority class accounted for 10% of total records in the dataset. The risk with this approach is that by removing records useful information may be discarded and undersampling an existing balanced data is not the same as applications on real-world imbalanced datasets.

5.3 How these results support real-world experiments

The purpose of this research was to test whether models performing classification using association rules, i.e. models converting unsupervised learning rules into class predictions, could outperform a number of benchmark supervised learning algorithms. In this research the *associative classifiers* did not perform as well as the traditional supervised learning approaches. This research supports the existing claims that associative classifiers do not perform well on imbalanced real-world datasets.

The association rules generated during the rule generation phase give intuitive insights into how certain variables interact with the target variable. The association rules generated could be used as part of the data discovery phase of traditional classification modelling to provide insight into the feature engineering process or to identify variables for consideration as interaction terms. Association rules may also be useful to explain relationships to business users where a more ‘black box’ model has been implemented. For example, if the model chosen for a particular use case is a neural network, this model does not easily allow intuitive rules or relationships to be pulled out for explanation to the business user. In this case association rules could be generated to supplement the neural network and help explain to business users some interesting relationships in the data.

As described in Chapter 1 there is new data protection regulation GDPR due to go live on May 25th, 2018. As part of the new regulation there is increasing oversight being placed on automated decision making. One example of automated decision making in a banking context includes real time credit decisions i.e. should we give this person a loan or not. In order to perform these decisions banks often use sophisticated data mining algorithms. The requirements under the new regulation allow the customer to request the underlying rules behind an automated decision if the decision has a material legal effect. This may have impacts for organisations using black box algorithms such as neural networks to carry out the credit scoring. There may be a role under GDPR for association rule learning to complement more sophisticated classification algorithms. Where the customer asks for the rules behind the decision the bank may be able to provide the association rules as an alternative approach as association rules are intuitive and can be easily described to an end customer.

Additional research would be required to test the experiments on Irish retail banking balanced datasets, for example, predicting product sales for small business customers tends to be more balanced. In Verhein and Chawla (2007), the authors compare the performance of SPARCCC, CBA, CMAR and C4.5 and the results show little difference in the accuracy of the predictions.

5.4 Software Evaluation

Experiments 1, 2 and 3 were built in the R statistical software. R is an open source software requiring the user to program their code using the R language. R has a wide variety of statistical and graphical techniques available through R packages to implement, visualise and evaluate models. Once the dataset was created the experiment could be designed in R and the various parameters of each model tuned to yield the highest of level of predictive accuracy. R also has certain limitations. For certain packages the documentation is poor, making it hard to decipher exactly what the various parameters are and what values they accept. Certain R packages can create memory issues, consuming all memory very quickly causing the software to stop functioning and the model needs to be re-run or re-designed to use less memory.

In summary, although R requires the user to code up the implementation of each algorithm and necessary visualisations, it provides the most comprehensive library of statistical techniques.

Experiment 4, CMAR, was run using the command prompt. The Java based models[12] were compiled and run through lines of codes in the command prompt, for example, java ClassCMAR_2file_App -Ftrain.txt -Ttest.txt -S5 -C80 -N2. This software had a number of limitations in terms of flexibility. There was limited scope to tune the model in terms of the rule rank order process, there was limited scope to tune parameters and there was no capability to visualise the model performance.

In Experiment 5, SPARCCC, the Java based models were loaded into WEKA. The ability to run the model from WEKA provided significantly more flexibility than running the source code directly. In WEKA, similar to R, the experiment could be designed easily and there was a certain level of flexibility to tune the parameters. WEKA also provides some data visualisation components for data discovery.

For data acquisition and data manipulation the majority of the effort was spent using SQL to structure the data into the ABT. This exercise was carried out using Teradata Studio an interface into the back-end Teradata database. This software proved to be a suitable tool for this exercise.

5.5 Conclusion

Chapter 5 details the results of the five experiments completed and evaluates the results in more detail. The results of the experiments conclude that the benchmark traditional classification models have outperformed the associative classifier models. The random forests model has performed the best following a comprehensive performance assessment across a number of balanced accuracy metrics such as AUC, F1-score and balanced accuracy.

The Chapter then presents the reasons for the poor performance of the associative classifiers. The associative classifier algorithms produced a high volume of type 1 errors or false

[12] https://cgi.csc.liv.ac.uk/~frans/KDD/Software/

positives. This indicates that the rules are not sophisticated enough to pick up certain patterns in the training data. The results of the CBA and CMAR experiments substantiate the claims made by Verhein and Chawla (2007) that these classifiers do not perform well on imbalanced datasets as they are biased towards the majority class.

The SPARCCC algorithm was designed to perform well on imbalanced datasets. The results of the experiment show that the SPARCCC algorithm outperforms CBA and CMAR but does not perform as well as the benchmark algorithms conditional inference trees and random forests.

Alternative uses of association rules were presented in this chapter. For example, association rules may be useful as part of the data discovery phase to identify hidden relationships which could highlight areas of consideration as part of the feature engineering process. Association rules could also be used to present patterns in the data where a 'black box' modelling approach has been applied. The use of association rules to support 'black box' algorithms may have a very valuable role under GDRP where organisations will be required to be able to explain the rules behind 'automated decision making'.

6. CONCLUSION

6.1 Introduction

This chapter concludes the dissertation and outlines how the research has achieved the previously stated goals. The research objectives are reiterated together with the results against the stated objectives. Any limitations identified as part of the experiments and evaluation are clearly stated. How this research extends the existing body of research in classification using association rules is presented. Finally, ideas and areas of interest for future work and research are highlighted.

6.2 Research Definition and Research Overview

The objective of this research was to assess whether association rule algorithms could produce statistically better classifications of mortgage sales than traditional classification algorithms in an Irish retail banking context. The state of the art research in the area of association learning and classification using association rules was reviewed and this research was utilised to design and implement five experiments to test the performance of *associative classifiers* with two traditional classification models. The experiments involved the implementation of three associative classifier models to test their accuracy in predicting Irish retail mortgage sales and comparing their performance to two benchmark models, random forests and conditional inference trees.

The objectives achieved by this research were:

- The state of the art literature was reviewed across knowledge discovery in databases, association rule learning, extending association rules to make class predictions and model evaluation methods.

- Five experiments were designed to assess whether associative classifier models could outperform traditional classification algorithms in predicting Irish retail mortgage sales.

- An Analytics Base Table was constructed using structured, semi-structured and unstructured data sources. The ABT developed formed the basis for model development.

- Five algorithmic experiments including two benchmark algorithms and three associative classifiers were designed and implemented.

- The five experiments were evaluated and compared to determine their strengths and limitations in predicting Irish mortgage sales.

6.3 Experimentation, Evaluation and Results

In order to achieve the objectives of this research a number of experiments were performed using various traditional supervised classification models and associative classifiers. Five experiments were performed to evaluate the performance of associative classifiers in predicting mortgage sales in an Irish retail banking context. The five experiments performed included two benchmark experiments for comparison purposes and three associative classifiers, CBA, CMAR and SPARCCC. The two benchmark models implemented were conditional inference trees and random forests.

The results of the five experiments were comprehensively evaluated across a number of accuracy measures. Given the imbalanced nature of the dataset, the straightforward accuracy measure was avoided in favour of more balanced metrics including AUC, F1-score and balanced accuracy. The results presented from this research show that the benchmark models performed better than the associative classifiers. The benchmark models provided the highest accuracy when taking a comprehensive view of all the various model performance metrics. Of the two benchmark models the random forest model performed best and would therefore be the chosen model for implementation into a production environment.

There are a number of limitations to the research:

- The CMAR algorithm implementation in this research was very inflexible in relation to the options for rule ranking. This is reflected in the poor results in this experiment

(Experiment 4). With more time the source code could be altered to remove the support ranking condition from the rule ranking process which is biased towards rules with the majority class as the consequent.

- The data used in the experiments is not exhaustive in terms of the data available on Bank of Ireland customers. Certain data, for example, granular spending behaviour at the store level and certain interaction data such as inbound call data was not used in this research. This data has not been included in the ABT as part of this research due to time restrictions. Perhaps additional data sources could improve the model accuracy in each of the experiments.

- Implementing certain experiments using very low support levels, for example, 0.1%, proved too computationally intensive from a memory usage perspective in R. Lower support levels can introduce high quality rules for the minority class. This restriction may have contributed to the bias towards the majority class.

6.4 Contributions to Body of Knowledge and Achievements

The experiments and results of this research have increased the body of knowledge on classification using association rules. The majority of the existing body of research on association rule learning focused on association rule learning as an unsupervised learning technique. Significant research has been carried out in the area of association rule learning primarily focusing on ways to improve the speed of rule generation and reduce the computationally expensive nature of generating rules (Han, Pei, & Yin, 2000).

Extending association rules to make predictions has received less research attention. Chapter 2 above outlines some of the seminal papers in the area of classification using association rules. Research such as Liu et al. (1998) and Li et al. (2001) have extended traditional association rule learning algorithms such as Apriori (Agrawal et al., 1994) and FP-growth to make class predictions. Verhein and Chawla (2007), claimed that existing models such as CBA and CMAR did not perform well on imbalanced datasets so the authors developed SPARCCC to deal with imbalanced datasets. However, in order to generate imbalanced

datasets in the SPARCCC experiments the authors under sampled existing UCI datasets and left the majority class intact.

This research has added to the existing body of research on associative classifiers by carrying out five experiments on a real world imbalanced dataset which is novel when compared to the previously reviewed state of the art research.

In this research, the associative classifiers performed poorly compared to the traditional classification models adding to claims in the existing body of research that associative classifiers perform poorly on imbalanced datasets.

6.5 Future Work and Research

This research focused on an imbalanced dataset where the minority class represented 1.2% of the total records in the dataset. Similar experiments could be carried out on more balanced datasets. In Verhein and Chawla (2007), the authors compare the performance of SPARCCC, CBA, CMAR and C4.5 and state that the results in terms of prediction accuracy were very similar. These claims could be tested on real world Irish retail banking datasets such as business customer datasets.

6.6 Conclusion

Chapter 6 concludes the research and experiments carried out to evaluate whether models performing class predictions using association rules could outperform traditional classification models in terms of classification accuracy. This chapter provided an overview of the research performed, outlined the experiments implemented and the results achieved. The evaluation of the research in this chapter also included certain limitations identified. Finally, chapter 6 presented some additional areas for future work to build on the experiments carried out as part of this research.

Bibliography

Agrawal, R., Imieliński, T., & Swami, A. (1993, June). Mining association rules between sets of items in large databases. In *Acm sigmod record* (Vol. 22, No. 2, pp. 207-216). ACM.

Agrawal, R., & Srikant, R. (1994, September). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB* (Vol. 1215, pp. 487-499).

Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., & Verkamo, A. I. (1996). Fast discovery of association rules. *Advances in knowledge discovery and data mining*, *12*(1), 307-328.

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., Paliouras, G., & Spyropoulos, C. D. (2000). An evaluation of naive bayesian anti-spam filtering. *arXiv preprint cs/0006013*.

Atiya, A. F. (2001). Bankruptcy prediction for credit risk using neural networks: A survey and new results. *IEEE Transactions on neural networks*, *12*(4), 929-935.

Aze Azevedo, A. I. R. L., & Santos, M. F. (2008). KDD, SEMMA and CRISP-DM: a parallel overview. *IADS-DM*.vedo, A. I. R. L., & Santos, M. F. (2008). KDD, SEMMA and CRISP-DM: a parallel overview. *IADS-DM*.

Barto, A. G., & Sutton, R. S. (1997). Reinforcement learning in artificial intelligence. In *Advances in Psychology* (Vol. 121, pp. 358-386). North-Holland.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern recognition*, *30*(7), 1145-1159.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5-32.

Brin, S., Motwani, R., Ullman, J. D., & Tsur, S. (1997). Dynamic itemset counting and implication rules for market basket data. *Acm Sigmod Record*, *26*(2), 255-264.

Brossette, S. E., Sprague, A. P., Hardin, J. M., Waites, K. B., Jones, W. T., & Moser, S. A. (1998). Association rules and data mining in hospital infection control and public health surveillance. *Journal of the American medical informatics association*, *5*(4), 373-381.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3), 15.

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Special issue on learning from imbalanced datasets. *ACM Sigkdd Explorations Newsletter*, *6*(1), 1-6.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321-357.

Drummond, C., & Holte, R. C. (2003, August). C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II* (Vol. 11, pp. 1-8). Washington DC: Citeseer.

Efron, B., & Tibshirani, R. (1997). Improvements on cross-validation: the 632+ bootstrap method. *Journal of the American Statistical Association*, *92*(438), 548-560.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*, *17*(3), 37.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1, pp. 337-387). New York: Springer series in statistics.

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural computation*, *4*(1), 1-58.

Han, J., Pei, J., & Yin, Y. (2000, May). Mining frequent patterns without candidate generation. In *ACM sigmod record* (Vol. 29, No. 2, pp. 1-12). ACM.

Hothorn, T., Hornik, K., & Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical statistics*, *15*(3), 651-674.

Heaton, J. (2016, March). Comparing dataset characteristics that favor the Apriori, Eclat or FP-Growth frequent itemset mining algorithms. In *SoutheastCon, 2016* (pp. 1-7). IEEE.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82-97.

Hipp, J., Güntzer, U., & Nakhaeizadeh, G. (2000). Algorithms for association rule mining—a general survey and comparison. *ACM sigkdd explorations newsletter*, *2*(1), 58-64.

Houtsma, M., & Swami, A. (1995, March). Set-oriented mining for association rules in relational databases. In *Data Engineering, 1995. Proceedings of the Eleventh International Conference on* (pp. 25-33). IEEE.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, *31*(3), 264-323.

Japkowicz, N. (2000, June). The class imbalance problem: Significance and strategies. In *Proc. of the Int'l Conf. on Artificial Intelligence*.

Kamakura, W. A., Wedel, M., De Rosa, F., & Mazzon, J. A. (2003). Cross-selling through database marketing: A mixed data factor analyzer for data augmentation and prediction. *International Journal of Research in marketing*, *20*(1), 45-65.

Kerber, R. (1992, July). Chimerge: Discretization of numeric attributes. In *Proceedings of the tenth national conference on Artificial intelligence* (pp. 123-128). Aaai Press.

Li, W., Han, J., & Pei, J. (2001). CMAR: Accurate and efficient classification based on multiple class-association rules. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on* (pp. 369-376). IEEE.

Ling, C. X., & Li, C. (1998, August). Data mining for direct marketing: Problems and solutions. In *KDD* (Vol. 98, pp. 73-79).

Liu, H., & Setiono, R. (1997). Feature selection via discretization. *IEEE Transactions on knowledge and Data Engineering*, *9*(4), 642-645.

Ma, B. L. W. H. Y., & Liu, B. (1998, August). Integrating classification and association rule mining. In *Proceedings of the fourth international conference on knowledge discovery and data mining*.

McCullagh, P. (1984). Generalized linear models. *European Journal of Operational Research*, *16*(3), 285-292.

Obenshain, M. K. (2004). Application of data mining techniques to healthcare data. *Infection Control & Hospital Epidemiology*, *25*(8), 690-695.

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., & Brunk, C. (1994). Reducing misclassification costs. In *Machine Learning Proceedings 1994* (pp. 217-225).

Provost, F., & Fawcett, T. (2001). Robust classification for imprecise environments. *Machine learning*, *42*(3), 203-231.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, *1*(1), 81-106.

Quinlan, J. R. (1993). C4. 5: Programming for machine learning. *Morgan Kauffmann*, *38*, 48.

Savasere, A., Omiecinski, E. R., & Navathe, S. B. (1995). *An efficient algorithm for mining association rules in large databases*. Georgia Institute of Technology.

Shearer, C. (2000). The CRISP-DM model: the new blueprint for data mining. *Journal of data warehousing*, *5*(4), 13-22.

Shipp, M. A., Ross, K. N., Tamayo, P., Weng, A. P., Kutok, J. L., Aguiar, R. C., ... & Ray, T. S. (2002). Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. *Nature medicine*, *8*(1), 68.

Tan, P. N. (2006). *Introduction to data mining*. Pearson Education India.

Verhein, F., & Chawla, S. (2006, December). Geometrically inspired itemset mining. In *Data Mining, 2006. ICDM'06. Sixth International Conference on* (pp. 655-666). IEEE.

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine learning*, *8*(3-4), 279-292.

Weiss, G. M., & Provost, F. (2003). Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, *19*, 315-354.

Xie, Y., Li, X., Ngai, E. W. T., & Ying, W. (2009). Customer churn prediction using improved balanced random forests. *Expert Systems with Applications*, *36*(3), 5445-5449.

Zaki, M. J., Parthasarathy, S., Ogihara, M., & Li, W. (1997, August). New Algorithms for Fast Discovery of Association Rules. In *KDD* (Vol. 97, pp. 283-286).

Zaki, M. J. (2000, August). Generating non-redundant association rules. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 34-43). ACM.

Zheng, Z., Kohavi, R., & Mason, L. (2001, August). Real world performance of association rule algorithms. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 401-406). ACM.

Appendix 1

Full list of structured variables in the ABT

PER_AGE	CLIENT_TYPE
PER_ISCO_OCCUPATION_CODE	PER_CREDIT_GRADE
PER_B365_REGISTERED	PER_B365_ACTIVE
PER_FLAG_CONSENT	PER_FLAG_DECEASED
PER_FLAG_PREMIER	PER_FLAG_PRIVATE
PER_HOUSE_STATUS	ABT_CNT_CLOSED
PER_FLAG_SEX	ABT_CNT_OPEN_JOINT
PER_FLAG_B365_ONLINE_6MTHS_USED	ABT_CNT_OPEN_CATEGORY_INVEST
PER_FLAG_GRADUATE	ABT_CNT_OPEN_CATEGORY_CC
PER_ROI_RESIDENT	ABT_CNT_CLOSED_CATEGORY_INVEST
ABT_FLAG_OPEN_JOINT	ABT_CNT_CLOSED_CATEGORY_CC
ABT_CNT_TOTAL_JOINT	ABT_CNT_TOTAL_CATEGORY_INVEST
ABT_CNT_OPEN_CATEGORY_LOANS	ABT_CNT_TOTAL_CATEGORY_CC
ABT_CNT_OPEN_CATEGORY_BIL	BRANCH_LAST_AC_OPENED
ABT_CNT_CLOSED_CATEGORY_LOANS	FNR_CNT_LAST_6MTHS
ABT_CNT_CLOSED_CATEGORY_BIL	FNR_FLAG_PROD_SAVINGS
ABT_CNT_TOTAL_CATEGORY_LOANS	FNR_FLAG_PROD_BIL
ABT_CNT_TOTAL_CATEGORY_BIL	CA_AVG_CR_TURNOVER_LAST_3MTH
ABT_DATE_AC_FIRST_OPENED	TRAN_FLAG_ACTIVE_LST_3MTH
FNR_CNT_LAST_12MTHS	OD_PERM_UTILISED
FNR_FLAG_PROD_INVEST	TRAN_FLAG_ACTIVE_LST_12MTH
FNR_FLAG_PROD_LOANS	OD_FLAG_PERM_LIMIT
CA_AVG_CR_TURNOVER_LAST_12MTH	CNT_ATM_DEBIT_12MTH
ABT_CNT_OPEN	CNT_ATM_CREDIT_12MTH
ABT_CNT_TOTAL	CNT_DD_DEBIT_12MTH
ABT_CNT_OPEN_CATEGORY_SAVINGS	CNT_DD_CREDIT_12MTH
ABT_CNT_OPEN_CATEGORY_MTG	CNT_SO_DEBIT_12MTH
ABT_CNT_CLOSED_CATEGORY_SAVINGS	CNT_SO_CREDIT_12MTH

ABT_CNT_CLOSED_CATEGORY_MTG	CNT_LODGE_DEBIT_12MTH
ABT_CNT_TOTAL_CATEGORY_SAVINGS	CNT_LODGE_CREDIT_12MTH
ABT_CNT_TOTAL_CATEGORY_MTG	CNT_FX_DEBIT_12MTH
BRANCH_FIRST_AC_OPENED	CNT_FX_CREDIT_12MTH
FNR_CNT_LAST_3MTHS	CNT_SO_DEBIT_3MTH
FNR_FLAG_PROD_MTG	CNT_SO_CREDIT_3MTH
FNR_FLAG_PROD_CC	CNT_LODGE_DEBIT_3MTH
CA_AVG_CR_TURNOVER_LAST_1MTH	CNT_LODGE_CREDIT_3MTH
TRAN_FLAG_ACTIVE_LST_1MTH	CNT_FX_DEBIT_3MTH
OD_PERM_LIMIT	CNT_FX_CREDIT_3MTH
CNT_SO_DEBIT_1MTH	CNT_365_DEBIT_1MTH
CNT_SO_CREDIT_1MTH	CNT_365_DEBIT_3MTH
CNT_LODGE_DEBIT_1MTH	CNT_365_DEBIT_MTH
CNT_LODGE_CREDIT_1MTH	CNT_365_DEBIT_12MTH
AMOUNT_FX_DEBIT_1MTH	AMOUNT_FX_CREDIT_1MTH
AMOUNT_SO_DEBIT_1MTH	AMOUNT_365_DEBIT_1MTH
AMOUNT_SO_CREDIT_1MTH	AMOUNT_365_DEBIT_3MTH
AMOUNT_LODGE_DEBIT_1MTH	AMOUNT_365_DEBIT_MTH
AMOUNT_LODGE_CREDIT_1MTH	AMOUNT_365_DEBIT_12MTH
AMOUNT_FX_DEBIT_1MTH	AMOUNT_FX_CREDIT_1MTH
AMOUNT_ATM_DEBIT_12MTH	AMOUNT_DD_DEBIT_6MTH
AMOUNT_ATM_CREDIT_12MTH	AMOUNT_DD_CREDIT_6MTH
AMOUNT_DD_DEBIT_12MTH	AMOUNT_SO_DEBIT_6MTH
AMOUNT_DD_CREDIT_12MTH	AMOUNT_SO_CREDIT_6MTH
AMOUNT_SO_DEBIT_12MTH	AMOUNT_LODGE_DEBIT_6MTH
AMOUNT_SO_CREDIT_12MTH	AMOUNT_LODGE_CREDIT_6MTH
AMOUNT_LODGE_DEBIT_12MTH	AMOUNT_LODGE_DEBIT_3MTH
AMOUNT_LODGE_CREDIT_12MTH	AMOUNT_LODGE_CREDIT_3MTH
AMOUNT_FX_DEBIT_12MTH	AMOUNT_FX_DEBIT_3MTH
AMOUNT_FX_CREDIT_12MTH	AMOUNT_FX_CREDIT_3MTH
AMOUNT_DD_DEBIT_1MTH	AMOUNT_DD_CREDIT_1MTH
TOTAL_CNT_FX_1MTH	TOTAL_CNT_FX_1MTH
TOTAL_CNT_SO_1MTH	TOTAL_CNT_365_1MTH
TOTAL_CNT_SO_1MTH	TOTAL_CNT_365_3MTH
TOTAL_CNT_LODGE_1MTH	TOTAL_CNT_365_MTH
TOTAL_CNT_LODGE_1MTH	TOTAL_CNT_365_12MTH
TOTAL_CNT_FX_1MTH	TOTAL_CNT_FX_1MTH
TOTAL_CNT_ATM_12MTH	TOTAL_CNT_DD_6MTH
TOTAL_CNT_ATM_12MTH	TOTAL_CNT_DD_6MTH
TOTAL_CNT_DD_12MTH	TOTAL_CNT_SO_6MTH

TOTAL_CNT_DD_12MTH	TOTAL_CNT_SO_6MTH
TOTAL_CNT_SO_12MTH	TOTAL_CNT_LODGE_6MTH
TOTAL_CNT_SO_12MTH	TOTAL_CNT_LODGE_6MTH
TOTAL_CNT_LODGE_12MTH	TOTAL_CNT_LODGE_3MTH
TOTAL_CNT_LODGE_12MTH	TOTAL_CNT_LODGE_3MTH
TOTAL_CNT_FX_12MTH	TOTAL_CNT_FX_3MTH
TOTAL_CNT_FX_12MTH	TOTAL_CNT_FX_3MTH
TOTAL_CNT_DD_1MTH	CNT_FX_CREDIT_6MTH
PER_NUM_DEPENDENTS	CNT_ATM_DEBIT_3MTH
PER_MARITAL_STATUS	CNT_ATM_CREDIT_3MTH
PER_FLAG_B365_ONLINE_3MTHS_USED	CNT_DD_DEBIT_3MTH
PER_FLAG_STUDENT	CNT_DD_CREDIT_3MTH
PER_STAFF	CNT_ATM_DEBIT_1MTH
ABT_FLAG_OPEN_DORMANT	CNT_ATM_CREDIT_1MTH
ABT_CNT_CLOSED_JOINT	CNT_DD_DEBIT_1MTH
ABT_CNT_OPEN_CATEGORY_INS	CNT_DD_CREDIT_1MTH
ABT_CNT_OPEN_CATEGORY_BIF	CNT_365_CREDIT_1MTH
ABT_CNT_CLOSED_CATEGORY_INS	CNT_365_CREDIT_3MTH
ABT_CNT_CLOSED_CATEGORY_BIF	CNT_365_CREDIT_MTH
ABT_CNT_TOTAL_CATEGORY_INS	CNT_365_CREDIT_12MTH
ABT_CNT_TOTAL_CATEGORY_BIF	AMOUNT_ATM_DEBIT_6MTH
ABT_DATE_AC_LAST_OPENED	AMOUNT_365_CREDIT_1MTH
FNR_CNT_LAST_9MTHS	AMOUNT_365_CREDIT_3MTH
FNR_FLAG_PROD_INS	AMOUNT_365_CREDIT_MTH
FNR_FLAG_PROD_BIF	AMOUNT_365_CREDIT_12MTH
CA_AVG_CR_TURNOVER_LAST_6MTH	AMOUNT_ATM_CREDIT_6MTH
TRAN_FLAG_ACTIVE_LST_9MTH	AMOUNT_FX_DEBIT_6MTH
OD_PCT_PERM_UTILISED	AMOUNT_FX_CREDIT_6MTH
CNT_ATM_DEBIT_6MTH	AMOUNT_ATM_DEBIT_3MTH
CNT_ATM_CREDIT_6MTH	AMOUNT_ATM_CREDIT_3MTH
CNT_DD_DEBIT_6MTH	AMOUNT_DD_DEBIT_3MTH
CNT_DD_CREDIT_6MTH	AMOUNT_DD_CREDIT_3MTH
CNT_SO_DEBIT_6MTH	AMOUNT_ATM_DEBIT_1MTH
CNT_SO_CREDIT_6MTH	AMOUNT_SO_DEBIT_3MTH
CNT_LODGE_DEBIT_6MTH	AMOUNT_SO_CREDIT_3MTH
CNT_LODGE_CREDIT_6MTH	AMOUNT_ATM_CREDIT_1MTH
CNT_FX_DEBIT_6MTH	TOTAL_CNT_DD_1MTH
TOTAL_CNT_ATM_3MTH	TOTAL_CNT_ATM_6MTH
TOTAL_CNT_ATM_3MTH	TOTAL_CNT_365_1MTH
TOTAL_CNT_DD_3MTH	TOTAL_CNT_365_3MTH
TOTAL_CNT_DD_3MTH	TOTAL_CNT_365_MTH

TOTAL_CNT_ATM_1MTH	TOTAL_CNT_365_12MTH
TOTAL_CNT_SO_3MTH	TOTAL_CNT_ATM_6MTH
TOTAL_CNT_SO_3MTH	TOTAL_CNT_FX_6MTH
TOTAL_CNT_ATM_1MTH	TOTAL_CNT_FX_6MTH