MSc Dissertation

UNIVERSITY OF THE
WITWATERSRAND,
JOHANNESBURG

Credit Scorecards: Can Machine learning methods enhance the current landscape?

By

Thapelo Sekanka (1105854)

Supervisor

Yogarani Chhana

A Dissertation submitted to the Faculty of Science, University of the Witwatersrand, in fulfilment of the requirements for the degree of Master of Science

February 27, 2022

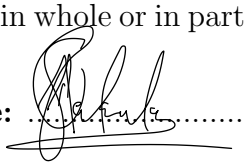
Contents

List of Figures

List of Tables

Declaration

I declare that this dissertation is my own work. Each significant contribution to, and quotation in, this dissertation from the work of other people has been attributed, cited and referenced. It is submitted to the Faculty of Science, University of the Witwatersrand, in fulfilment of the requirements for the degree of Master of Science. This dissertation has not, either in whole or in part, been submitted for a degree or diploma to any other university.

Signature: 

Date: February 27, 2022

Dedication

I would like dedicate this dissertation to my mother Catherine Louiza Kedibone Sekanka.

Acknowledgements

Throughout the writing of this dissertation I have received a great deal of support and assistance.

I would first like to thank my supervisor, Yogarani Chhana, whose expertise was invaluable throughout this journey. Your insightful feedback pushed me to sharpen my thinking and brought my work to a higher level.

I would also like to thank my family for the support and encouragement they gave me.

Abstract

Credit Scoring is a system that generates a credit score for a customer based on their financial behaviour. The higher the score, the more likely it is that the bank will approve the customer's request for credit. This study investigates a number of different methods that can be used for credit scoring and compares them. The study also assesses the stability of the scorecards by measuring the performance of the scorecard using a dataset that was not part of the development sample. The random forest regression scorecards provide the best relative performance out of all. The scorecards were implemented using a static as well as a dynamic approach. The dynamic approach provides better overall performance for most of the scorecards it allows the scorecards to improve by learning from the data.

List of Acronyms

ANN	Artificial Neural Network
AM	Account Management
AO	Account Origination
AUC	Area under the curve
BVT	Built, Validation and Test
CBOS	Cluster based over sampling
KS	Kolmogorov-Smirnov
LR	Logistic Regression
LDA	Linear Discriminant Analysis
ML	Machine Learning
OOT	Out of Time
PD	Probability of Default
PIT	Point in Time
PR	Probit Regression
POI	Poisson Regression
RFR	Random Forest Regression
ROC	Receiver of operating characteristics
SMOTE	Synthetic minority over sampling
SM	Statistical Method
SVM	Support Vector Machines
SV	Survival Modelling
IV	Information Value
WOE	Weight of Evidence

Chapter 1

Introduction

1.1 General Introduction

The word credit is derived from the Latin word "creditum" which means trust. Investopedia (2020) describes Credit as a concord between a borrower and a lender. The borrower receives money at the time of the agreement and the lender believes that the borrower will pay back the amount, including the cost (in the form of interest) charged by the lender. In this study, the lender is a Bank and the borrower is the bank's customer. The ability to pay back will vary amongst these customers. Abdou and Pointon (2011) and Siddiqi (2017) indicated that Banks employ a system that uses statistical methods to inform them about the borrower's financial behaviour and credit. The system generates a score, and the higher the score, the more likely it is that the bank will approve the customer's request for funding. This way of assigning a credit score to customers is called Credit Scoring. Anderson (2020) indicated that Credit Scoring was first commercialised in the 1950s by Fair Isaac and Company (FICO). Customers are assigned credit scores using a credit scorecard. A credit scorecard consists of partial scores for each customer characteristic which a customer may be classified. A customer is asked a series of questions and the answer for each question determines the partial score for the customer. Once the questions are answered by the customer, the partial scores are added, resulting in the customer's credit score.

Credit Scorecards are developed using historical data, and this process is called scorecard development. The assumption made is that the population will behave in a similar manner to the historical population. Historical credit payment information about clients is needed. Two main types of client data are available when an application comes to the credit provider. The first type is data that is stored by the credit provider (Banks). This can be the current customer base, and/or application data about prospective customers whose applications were not presently successful. The other type of client data is obtained from credit bureaus. Credit bureaus are institutions that collect customer credit data from various creditors. This

data gives a holistic view of the client and their credit behaviour with all credit providers. Dong, Lai and Yen (2010) explain that although there are many ways of developing credit scorecards, they need to answer the fundamental question being - "what is the probability that a customer will not be able to pay back?"

Although there are some overlaps, TransUnion (2020) describes that credit scorecards fall into two main categories, namely Account Origination (AO) and Account Management (AM). AO scorecards are developed for new credit accounts that are opened for both existing and first-time clients. The data used for AO scorecards comprises mainly of bureau information since the credit provider would not have data about all clients that may apply for new credit products and hence be scored through this scorecard. AM scorecards are developed exclusively for clients who have existing credit products. Only existing clients may be scored using this scorecard. AM scorecards can be developed with or without bureau data.

The underlying concept in credit scoring is the concept of odds. Odds are expressed as a ratio of the probability of one event to that of an alternative event as shown in Equation 1.1. The two events in credit scoring are an account being in default, meaning a payment was missed, and the account being up to date. Equation 1.1 shows that the odds are calculated by the probability of default (PD) divided by the probability of not default, and as seen in the equation this is (1 - PD).

$$Odds = \frac{No.\ of\ Bad\ Accounts}{No.\ of\ Good\ Accounts} = \frac{P(Default)}{P(Not\ default)} = \frac{PD}{1 - PD}\,. \tag{1.1}$$

Using historical data to calculate how each customer attribute contributes to the default odds of an applicant can be executed by pen and paper if there are a few customer variables, for example, if only Age and Income are used for this purpose. The final customer-specific odds are a product of each of the variable odds. FICO found that working with odds can be very tedious and prone to calculation errors. They looked at several statistical techniques to simplify credit scoring. This led to the scaling of odds using the natural logarithm and then the sum gave the customer score. The market reacted better to scores as compared to odds. This gave birth to the Logistic function in credit scoring.

Table 1.1 illustrates an example of a scorecard. In this example, the total customer score is the sum of three partial scores from each of the variables. The minimum score is 150, for a customer who is under 18 years or over 60 years of age, does not own or rent a home and has no income. The maximum attainable score is 850 points and this score will be given to a customer who is between 35 and 60 years of age, owns a home and has an annual income of more than R750 001. This transparency contributed to logistic regression being the current industry standard. There is, however, a need to develop better models to address inefficiencies, prediction accuracy and cost-effectiveness. Baesens, Van Gestel, Viaene, Stepanova, Suykens and Vanthienen (2003) show that there is a wide range of algorithms that rival Logistic Regression within credit scoring.

Variable	Group	Partial Score
Age	[18;20]	100
Age	[21;35]	150
Age	[36;60]	200
Age	Other	50
Home	Rent	130
Home	Own	300
Home	Other	50
Income	<R250k	80
Income	[R250k; R500k)	110
Income	[500k; R750k)	250
Income	>= R750k	350
Income	No Income	50

Table 1.1: Scorecard example adapted from MathWorks (2020)

1.2 Regulation

In South Africa, the National Credit Act (2006) (NCA) states that when assessing credit applications, providers must use a clear mechanism, "statistical or otherwise". This section of the regulation is interpreted and adhered to by credit providers through developing credit scorecards that yield outputs like the example in Table 1.1. The "otherwise" description would refer to judgemental scorecards, where only experience is taken into account when assessing credit applications. According to the regulations, a judgemental scorecard is acceptable as long as the factors leading to the conclusion of an application are clear and openly communicated. The act further states that in the event of the refusal of credit, the client can request reasons as to why this decision was made. The bank must advise that client in writing of the main reasons for rejecting their credit application. Credit applications can be for a new credit agreement, an extension, or a renewal of existing credit. The need for banks to be able to provide these details makes it difficult for banks to use "black box" methods or methods that do not explicitly spell out the criteria for developing credit scorecards.

1.3 Aims and Objectives

This research aims to explore the possibility of a credit scoring framework that can potentially rival the current landscape. The framework has to abide by the industry standard while

providing better overall scorecards as required by the NCA. This will be achieved by firstly developing credit scorecards using Statistical Techniques, Machine learning Techniques and a Survival Analysis technique. The scorecard development process as outlined by Siddiqi (2017), has five critical steps. The steps are:

1. Determining the outcome variable.
2. Data selection, sampling period and sampling method.
3. Variable selection.
4. Variable grouping.
5. The modelling technique.

Scorecards will be developed under these guidelines for all modelling techniques investigated by this research.

The point in time (PIT) performance of each of the scorecards will then be compared. Point in time means "at a specific date". In this case, this date is immediately after the chosen sampling period, even though the final chosen scorecard usually depends on the business needs and capabilities.

The performance of the scorecards developed by the Built, Validation and Testing (BVT) samples will be assessed. Siddiqi (2017) indicates that there are a number of methods to compare credit scorecards. In this research, the scorecards will be compared using the Kolmogorov-Smirnov Statistic, Gini coefficient, confusion matrix, and the area under the Receiver Operating Characteristic curve (ROC). The methods used to compare the scorecards may be enriched as the research proceeds.

This research will also compare the out of time (OOT) performance. OOT means through a chosen period. In this case, OOT will be a year immediately after the scorecards have been developed. The predictive power of the scorecards will be monitored monthly for the specified period, using the above-mentioned statistics. This research will use historical data and the actual outcome will already be available for each of the monitored months as the data will be scored for each time period and the predicted outcome will be compared to the actual known outcome for the specific time period.

1.4 Outline of the Study

Chapter one introduced the reader to the concept of credit scoring. It reviewed the regulation around credit scoring in South Africa and outlined the aims and objectives of the study as well as how the scorecards will be evaluated.

Chapter two gives an overview of the procedure of developing a scorecard. The choice of the outcome variable, the data and sampling, variable selection and grouping, modelling techniques and evaluation methods of the final scorecards are discussed. Thereafter, the available literature is reviewed for each of the aspects.

Chapter three looks at the theoretical review of the methodology. This will give a background and underlying theory behind the methods and techniques investigated in this study.

Chapter four discusses and gives a more detailed overview of the data used.

The following chapters discuss and interpret the practical work undertaken in this study. Chapter five discusses sampling methods and variable preparation and selection, while chapter six discusses the results of the developed scorecard.

Chapter seven assesses the developed scorecard for OOT performance.

Chapter eight concludes the study and gives recommendations.

Chapter 2

Literature Review

2.1 Introduction

This chapter provides a literature review of the studies undertaken in addressing the topic of credit risk and their applications to credit scorecards. The goal of credit scorecards is to classify consumers of credit products according to their payment profile, either potential or existing. This is achieved by analysing the customer's past financial behaviour to predict their potential future behaviour. This study follows the credit scorecard development process as defined by Siddiqi (2017). The subsequent sections present a review of the literature within the key components of the credit scoring process in terms of the choice of the outcome variable, data selection, sampling, variable selection, modelling and scorecard evaluation, and choice of approach using Machine learning (ML) and Statistical methods (SM). Ij (2018) note that SM techniques are those that use the sampled data to draw inferences about the population and generate predictions, and that ML techniques aim to find patterns that could be generalised in the data as well as learn from the data to improve predictions. Michie et al. (1994) further note that models produced using ML techniques do not require extensive training like models produced using SM techniques. Predictions produced by ML models improve over a period of time, while predictions produced by SM models do not improve under these conditions.

2.2 Outcome Variable

The outcome variable describes the problem statement that the scorecard attempts to answer, and hence it must be well-defined. Accurately predicting the outcome variable is the main aim of developing credit scorecards. Hence, the definition of the outcome variable is very important to the model development process.

The outcome variable of a scorecard is also referred to as the Target variable or Response variable. This research will use outcome variable to refer to variable that the scorecard aims to predict. It can be binary or continuous but Siddiqi (2017) shows that credit scorecards are modelled on a binary outcome variable. Even though the output of a scorecard is not binary, the scorecard is used to predict an event or non-event. In this study, the event occurs when the customer has defaulted on their loan agreement, and the non-event is when their payments are up to date.

The Basel Accords refer to the recommendations on banking regulations issued by the Basel Committee on Banking Supervision (BCBS). In the Basel Accord (2004) a "default" occurs when the bank considers the customer not likely to pay back what they owe and/or when the customer is more than 90 days overdue on their loan instalments.

2.3 Data selection and sampling

Historical data is needed for scorecard development. The data can be seen as a $n \times m$ matrix, with m individual customer features and n observations. Observations are customers observed at certain intervals, hence monthly payment data is collected and aggregated. The features are the variables in the data, for example, Age, Income and Employment type.

After the dataset has been obtained, a sample dataset and sample period for the data needs to be chosen. The sample dataset needs to be the best representation of the original data. Crone and Finlay (2012) note that historically the choice of sample is largely based on expert opinion. This is because real-world data is usually unbalanced across classes, and this affects the performance of credit scoring models. A dataset is said to be class unbalanced when the outcome variable has one of the outcomes ("good","bad") appearing significantly more than the other. For example, a dataset of 10 000 observations with 100 "bad" observations and 9 900 "good" observations would be classified as class unbalanced.

Junior, Nardini, Renso and de Macêdo (2019), Jing, Ling, Dun-Hu, and Jin (2017) and Crone and Finlay (2012) all agree that sampling technique can affect the overall performance. Jing et al. (2017) investigates the effect of five sampling strategies and also investigated not using

a sampling technique. The sampling strategies included one side selection sampling (OSS), random over-sampling (ROS), cluster-based oversampling (CBOS), random under-sampling (RUS), synthetic minority over-sampling technique (SMOTE). The techniques were compared using models developed using the Logistic regression (LR) technique. It is worth noting that there was no strategy that offered the best performance for all models.

Junior et al. (2019) suggested sampling using the k-Nearest Oracle Union (KNORA-U) dynamic selection technique combined with Balanced Random Forest, as this improved classification. Jing et al. (2017) found that re-sampling improves the performance of credit scoring, while noting that the SMOTE worked well with most models in their experiment. Chawla, Bowyer, Hall, and Kegelmeyer (2011) support the use of SMOTE in credit scoring based on their finding of a notable increase in accuracy across different modelling techniques, where the data was sampled by SMOTE. Siddiqi (2017) recommends that the sample size required to develop a scorecard is about 10 000 observations, but Crone and Finlay (2012) suggest that larger sample sizes are better.

The sample period is another debated topic in credit scoring. Tian, Xiao, Wang, Peng, Xing, Liao et al. (2017) investigated the influence of the sampling period on the final output and found that the choice of sampling period can assist in improving accuracy. Siddiqi (2017) and Tian et al. (2017) note that if the sample period is further back in history prior to the period for which the scorecard is being developed, there's a risk that the output might not be an accurate representation of the development landscape.

Jing et al. (2017) further recommended that the chosen sampling technique should be based on the evaluation criteria and models selected and this sentiment is shared by Zhu, Baesens, Backiel, and Vanden Broucke (2018). Zhu et al. (2018) also recommend different sampling strategies for different modelling techniques. Based on the above literature the SM sampling strategies that will be investigated by this study are CBOS and ROS, and the ML sampling strategy that will be investigated in this study is SMOTE.

2.4 Variable selection and Grouping

Variable selection is a process of choosing the variables to be used in developing the scorecard. This process helps to reduce the number of available variables by removing ones that would not be useful in scorecard development. The variables are compared to the outcome variable and based on whether there is a relationship between them, some variables will be chosen and some removed. For example, Age might have a direct relationship with the outcome variable like a high probability of default (PD) in the younger population and a lower PD in the older population. Another example is a customer's annual income having a direct relationship to the outcome variable. The figures below are visual representations of such

relationships.

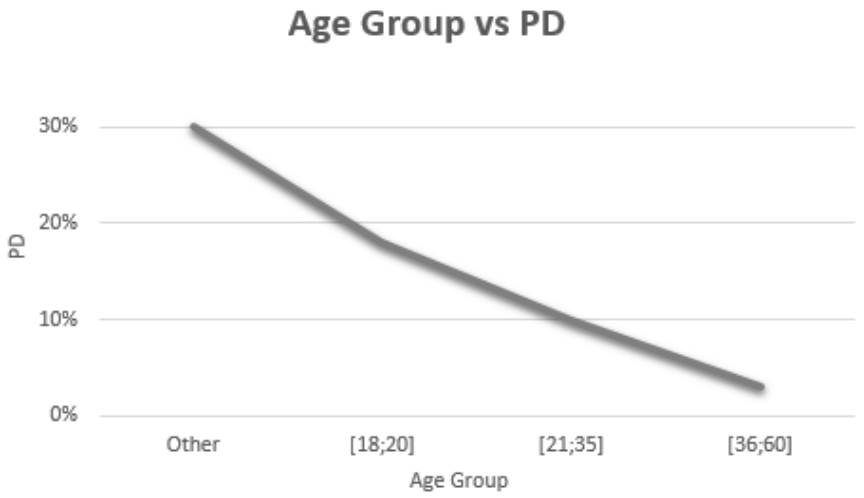


Figure 2.1: Age and Outcome Relationship

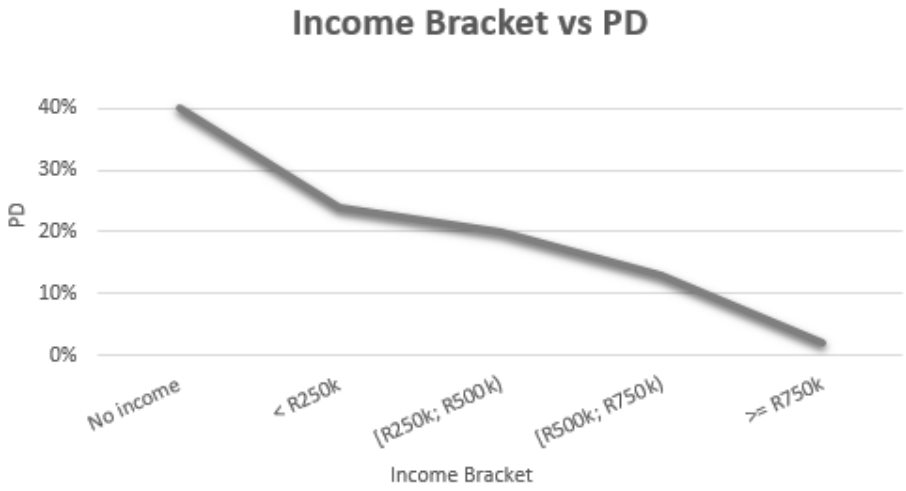


Figure 2.2: Income and Outcome Relationship

The common statistical technique to select variables is the use of the information value statistic (IV), which is an example of a filter feature selection method. Zeng (2013) and Siddiqi (2017) define IV as a value to quantify the predictive ability of an independent variable when compared to the outcome variable without considering other independent variables and show that IV is a useful measure to reduce the number of variables from big datasets with a high number of independent variables. Lund and Brotherton (2013) give guidelines on how to use and interpret the IV statistic. The underlying theory of IV allows

it to be classified as a statistical technique. Variables with a relatively higher IV than the rest of the variables will generally be preferred compared to those with a lower IV.

Siddiqi (2017) suggests that the recommended number of variables required to develop a scorecard is five to fifteen variables. Classical statistical methods such as IV and Chi-Square test of independence, as well as machine learning methods such as k-means clustering, can be utilised for choosing variables to be modelled. The Chi-square test of independence tests the for the relation between two variables and k-means clustering for variables partitions the variable space into k non-overlapping clusters such that variables within in a cluster are similar to each other. The commonality among these methods is that they take two types of relationships into account and rank the optimal one. Firstly, the relationship between the variable and the outcome variable and secondly, the relationship between the variable and the other independent variables that can be used. For example, the correlation coefficient can be used to assess the relationship between Age and Income. If the two variables have a high correlation coefficient then it means that there is a strong linear relationship between them and hence shows dependence. The statement is not necessarily true in the opposite direction, though. Variables that are non-linearly dependent, can still give a correlation coefficient that is low, hence the correlation coefficient alone is not sufficient to detect full dependency as shown by Benesty et al. (2009).

Cateni, Colla, and Vannucci (2017) highlight three main types of variable selection methods, namely, Filter, Wrapper and Embedded methods. Filter selection methods use a statistical or machine learning technique to evaluate the relationship between each input variable and outcome variable. In Wrapper selection methods, a number of models are developed using random combinations of variables to choose the best performing model. Wrapper and Filter methods can be combined to form Embedded methods. This research will focus on these filter methods. Cai, Zhang and He (2010) proposed Multi-Cluster Feature Selection (MCFS) for unsupervised feature selection. This method can also be classified as a filter method. With this method, the independent variables are compared amongst themselves and not with the outcome variable. Alfat, Rizkinia, Sari and Romano (2019) showed the usage of MCFS in credit scoring by showing how variables that may have been discarded by traditional methods can be useful.

Both ML and SM algorithms can use categorical and continuous variables as inputs. The key difference is the variable transformations. Credit scoring uses grouped variables, as seen in Table 1.1. Each continuous variable will be grouped by choosing intervals and each categorical variable will be grouped by placing similar types of categories together. Referring to Table 1.1, Age would be an example of a continuous variable with intervals whereas Home is a categorical variable.

The subgroups in each variable will be internally homogeneous and externally heterogeneous. This process is referred to as "variable binning". There are supervised variable grouping methods where the variable groups are set to optimise a predetermined statistic. These grouping methods are referred to as Statistical methods. On the other hand, there are

unsupervised variable grouping methods, where the variable is the input for a machine learning algorithm and the algorithm determines the optimal bins. This study investigates and uses the weight of evidence (WOE) measure to group all variables. WOE is a supervised statistical method. This method bins the variables internally based on the similarity within the data points. The inputs for this method are the actual data and the bad rate. A bad rate is the number of defaults divided by the total number of observations. Sidiqqi (2017) has shown that WOE is an efficient way to analyse grouped data for credit scorecards.

2.5 Modelling

Credit scorecards are used to predict the probability of default, which is presented as a score. A modelling algorithm is employed to estimate the scores for each subgroup in each variable, and hence, all the variables need to have been internally grouped before the modelling procedure commences. Both Machine Learning methods and Statistical methods have modelling algorithms that can be used to develop credit scorecards.

The modelling algorithms proposed to be investigated by this research are LR, probit regression (PR), poisson regression (POI), and linear discriminant analysis (LDA) from the statistical methods and support vector machines (SVM), artificial neural network (ANN) and random forest regression (RFR) from the Machine Learning Methods. Additionally, an exponential distribution (parametric model) used to estimate the survival curve for survival modelling (SV) as suggested by Man (2014) and Chimedza and Marimo (2017) will also be explored by this research.

Abdou and Pointon (2011) reviewed over 200 published papers that involve credit scoring. Although there is no single method that is ideal for all types of data, ML have been shown to outperform SM. This study compares the performance of SM and ML Techniques for credit scorecards, and also assesses them over time. The predictive ability of the models developed will be monitored, reported and compared monthly over a year to assess how the scorecards perform the further away from the development time. All methods predict a probability which is an element of $\{0, 1\}$, then the probability is converted into a score. Based on business rules, the score or probability is translated into $y \in 0, 1$. For this study, the predicted outcome will be assigned the value of 1 if the probability of default is greater or equal to 0.5 and 0 otherwise.

2.5.1 Statistical methods (SM)

Ij (2018) note that SM techniques are those that use the sampled data to draw inferences about the population and generate predictions. Michie et al. (1994) further note that models developed by SM techniques require extensive training and that predictions produced by SM models do not improve under when implemented. Anderson (2020) indicates that the first credit scorecards were developed using logistic regression (LR) and the literature shows that LR is a popular technique in credit scoring. As Abdou and Pointon (2011) outlined, some of the celebrated benefits of LR include long-term dependencies, even though a drawback is that there are assumptions needed about the data, as shown by Siddiqi (2017) and Altland (2012). These are some of reasons that there is a need for improved scorecards.

Probit regression (PR) is suggested as an alternative to Logistic regression. The PR method is based on the Normal distribution. Ren, Hou, and Li (2013) found that the PR based credit scoring models can be effective in risk evaluation. Chasco, Aroca, and Anselin (2019) concluded that PR models are related to Logit models and hence can be useful in credit scoring. Hussain, Khan, Rehman, Shams, and Khattak (2019) found that the Probit based model performed better than Logistic regression and Discriminant analysis for their specific credit scoring model.

Henley (1995) and Karlis and Rahmouni (2007) have shown that the application of the Poisson distribution in Credit Scoring by developing credit scoring models that competitively performed against Logit based models.

Henley (1995) indicated that LDA is one of the earliest statistical techniques used to separate distinct populations. Hooman, Marthandan, Yusoff, Omid and Karamizadeh (2016) applied LDA in credit scoring and found that LDA outperformed LR. Hooman et al. (2016) further showed that there were novelties in using a hybrid that combined LDA and LR. Casin (2018) also proposed a hybrid version of LDA and called it Categorical multiblock linear discriminant analysis.

Bücker et al. (2019) explain that the underlying theory behind statistical models is transparent and hence acceptable in credit scoring applications. The regulatory requirements that credit application decisions be transparent, as documented in the National Credit Act (2006) make methods like LR more attractive to credit providers, as indicated by Lessmann, Baesens, Seow and Thomas (2015).

2.5.2 Machine learning techniques

Ij (2018) notes ML techniques are techniques that aim to find patterns that could be generalised in the data as well as learn from the data to improve predictions. Michie et

al. (1994) further note that models produced using ML techniques do not require extensive training. Predictions produced by ML models improve over a period of time. Machine learning techniques used in credit scoring applications include but are not limited to Support Vector Machines (SVM) utilised by Van Gestel, Baesens, Garcia and Van Dijcke (2003) and Nehrebecka (2018), Artificial Neural Networks (ANN) as investigated by West (2000) and Pacelli and Azzollini (2011), as well as Random Forest Regression introduced by Breiman (2001). Dong et al. (2010) indicate that although many Machine Learning techniques have been investigated in the context of credit scoring, they have not been implemented in developing operational credit scorecards. This is mainly due to the lack of transparency of the machine learning methods as shown by Abdou and Pointon (2011), West (2000) and Baesens, Van Gestel, Viaene, Stepanova, Suykens and Vanthienen (2003).

Pacelli and Azzollini (2011) concluded that ANNs can be an alternative technique for credit scoring. West (2000) and Abdou and Pointon (2011) investigated various modelling techniques and found that ANNs were amongst the top performers. Even though ANNs have high accuracy when compared to other techniques, Doori and Beyrouti (2014) outline the importance of choosing the correct activation and error functions. This is a disadvantage since SMs do not have such complexities, since there are no arbitrary inputs to in SMs.

Van Gestel et al. (2003) acknowledged the applications of SVMs in credit scoring and reported better results when experimenting with a version of SVM named least square support vector machines (LS-SVMs). Nehrebecka (2018) compared logistic regression to SVMs in credit scoring and found that there were significant differences in the respective performances of the developed models, as SVMs outperformed logistic regression models.

Random Forests were introduced by Breiman (2001), who showed that the Random Forest theory could be applied in regression. Mercadier and Lardy (2019) and Saitoh (2016) agree that random forest regression can improve the accuracy of credit scoring models.

Ghodselahi and Amirmadhi (2011) proposed a hybrid model, which had better results when compared to traditional methods. However, the long-term quality of the resultant scorecard has not been assessed. Hand and Henley (1997) note that on a theoretical basis, Logistic regression has the upper hand when compared to linear regression. Doori and Beyrouti (2014), Nehrebecka (2018) and Yao and Chen (2019) provide a different view, showing that Machine Learning Methods like Support Vector Machines and Artificial Neural Networks, also perform better than classical statistical methods. Wang, Hao, Ma and Jiang (2011) proposed an ensemble learning model for credit scorecards. Ensemble learning methods are methods that use more than one modelling technique to produce predictions.

2.5.3 Survival Modelling

Survival analysis is a statistical data analysis method used to predict the time until the occurrence of an event, often referred to as a failure time. This can be modified to predict the probability of default, and hence scorecard development. The "event" in the context of credit scoring and this research is "if the instalments are late by at least a 90 days". For modelling survival data, different models can be applied.

This research will apply the PD (Probability of Default) estimation framework as proposed by Man (2014) to develop a scorecard. Chimedza and Marimo (2017) also show that SV can contribute to the prediction of bank loans. A PD model and a credit scorecard are related, even though a PD model is not necessarily a scorecard. A PD model's very nature is to output a probability, whereas a scorecard outputs a score. Siddiqi (2017) gives a formula that can convert PD's into scores. This formula will be used for all methods

2.6 Scorecard Evaluation

Siddiqi (2017), Doori and Beyrouti (2014), and Al Ghayab, Li, Abdulla, Diykh and Wan (2016) note that it is not ideal to use only one criterion to evaluate the performance of scorecards. Abdou and Pointon (2011) suggest criteria like the estimated misclassification cost criterion and the receiver operating characteristic (ROC) curve to rank the credit scorecards. All the scorecards investigated by this research will be measured using the Kolmogorov-Smirnov (KS) statistic, Gini coefficient, the confusion matrix and the area under the receiver operating characteristic (ROC) curve.

The KS statistic looks at the maximum difference between the cumulative distributions of bad and good clients. The Gini coefficient measures the scorecard's ability to separate the scored population. The Confusion matrix outputs a matrix where each row shows values in the predicted class and the columns show the actual values. It is important because, amongst others, values like accuracy and specificity can be directly calculated for the matrix. The area under the receiver operating characteristic curve (ROC) is a graph that plots the distribution of true positives against the distribution of false positives.

2.7 Summary

This study attempts to contribute to credit scoring in two ways. Firstly, by demonstrating that even though classical credit scoring has been traditionally a classical statistical problem, it is also possible to develop credit scorecards using machine learning techniques.

This study investigates all the steps involved in credit scoring, and not only the modelling aspect. Attempts are being made to introduce machine learning techniques into the credit scoring steps of data sampling and variable selection where previously only classical statistical methods have been used.

Finally, the study assesses the impact of time on the "robustness" of the scorecards developed using the different techniques.

Chapter 3

Methodology and Theoretical Review

3.1 Introduction

This chapter gives a theoretical view of the methodology used in this study. Recall from the literature review the steps involved in credit scoring looked at the data, the sampling methods, variable selection methods, and grouping, modelling and then evaluation of the scorecards. The type of data needed for developing a scorecard has been discussed in the previous chapter. Figure 3.1 shows the credit scoring process from data to the evaluation of the scorecard.

Figure 3.1: Complete process end to end

3.2 Data

Figure 3.2: Data

The most important component of credit scorecard development is the data. The data should have the outcome variable, which is what the scorecard aims to predict and the variables that are used to generate the predictions. More details about the particular dataset utilised in this research is given in Chapter 4. Figure 3.2 shows that data is the first step of the scorecard development process.

3.3 Sampling

Figure 3.3: Sampling

The chosen sample needs to be representative of the population in order to avoid bias and sampling errors. This section looks at the different sampling methods used in this study, that were selected in the literature review, namely: Random oversampling (ROS), Synthetic Minority Oversampling Technique (SMOTE) and Cluster-based oversampling (CBOS).

3.3.1 Random oversampling (ROS)

ROS combines two known sampling techniques, random sampling and over sampling. Random sampling is a sampling technique where each sampling unit has an equal chance of being selected. The probability of selecting any element from a population of N elements is then $\frac{1}{N}$. Oversampling is a technique that is used when there is not enough data for modelling, but it can also be used to adjust the class distribution of a dataset when dealing with class imbalanced data. A class imbalanced dataset is one where the outcome variable with, say, two classes ("good" and "bad"), has one class appearing significantly more than the other. For instance, having significantly more "good" than "bad". Figure 3.4 below shows the impact of oversampling on the population dataset. First, the class that is under-represented is duplicated once thereafter a random sample is chosen. This procedure has been shown by Jing et al. (2017) to increase the probability of producing samples that adequately represent the population.

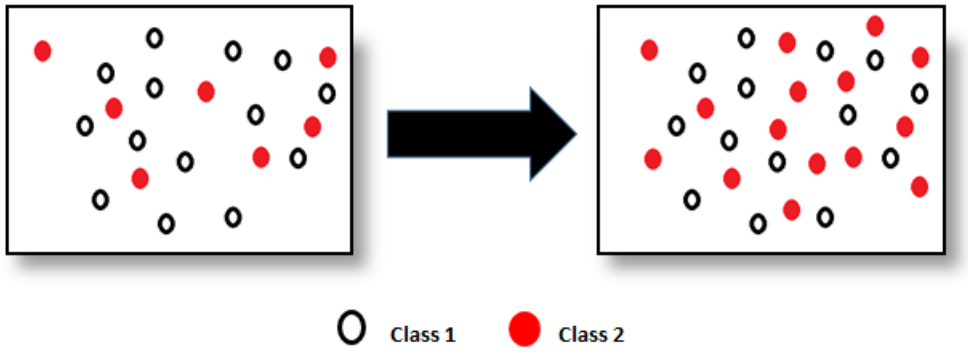


Figure 3.4: Imbalanced Dataset example adapted from Ali et al. (2019)

This study found that another way of applying ROS is to first separate the population dataset by class into two smaller datasets, the minority class and the majority class, then choosing a random sample with replacement from each dataset and combining them, but intentionally choosing the proportion of the minority class. The result is a sample with the desired class proportion. For example, if the population has 10 000 customers and only 100 of them are classified as "bad", implying a bad rate of 1%, ROS can be applied to a sample of 500 customers by randomly sampling 50 "bad" customers and 450 "good" ones. The overall bad rate for this hypothetical sample is now 10%. ROS has a risk of overfitting the data. The risk is that only very similar customers may end up in the sample, giving results that may be distorted.

3.3.2 Synthetic Minority Oversampling Technique (SMOTE)

The issue of overfitting can be addressed by SMOTE, which is a variant of oversampling that oversamples the data by creating synthetic examples of the minority class. The minority class is the under-represented class. According to Chawla et al. (2011), the t minority nearest neighbours are chosen at random based on the amount of oversampling required. Then, along the line segments connecting the nearest neighbours in the t minority classes, synthetic examples are generated.

Figure 3.5 shows how the SMOTE sample are created. In the figure, the orange squares are the "defaults" and are clearly fewer than the blue dots. Then using the SMOTE, the middle part of the figure is produced, and the green squares are the synthetic samples. The last part of the figure shows the complete sample with the synthetic samples.

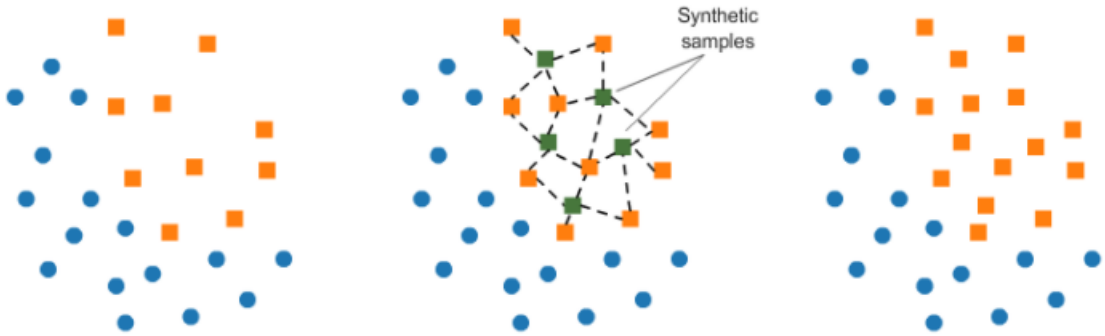


Figure 3.5: Smote algorithm adapted from Das (2019)

3.3.3 Cluster-based oversampling (CBOS)

CBOS combines two methods, namely clustering and oversampling. There are no prescribed methods to be used, and this study combines KNORA-U and ROS. ROS was chosen as it has been shown by Jing et al. (2017) to improve scorecard performance, and KNORA-U was selected because it was shown by Junior et al. (2019) to improve classification in credit scoring data. This technique considers the population as one big cluster, and this cluster is then divided into smaller clusters, and the minority class within each cluster is then oversampled. The cluster sizes are arbitrary, and are chosen at random. Since the minority class is oversampled, it is important that every cluster contains at least one observation from the minority class since the oversampled examples will be based on this class in each cluster.

Figure 3.6 shows the CBOS technique in action. The first part of the figure shows the sample with red squares as the defaults and the triangles the good observations. In the middle part,

the data is divided into two clusters, and in each of the clusters, the defaults are oversampled. The last part of the figure shows the output of the sampling procedure. and it can be seen that the oversampled observations are not concentrated in any single part of the sample.

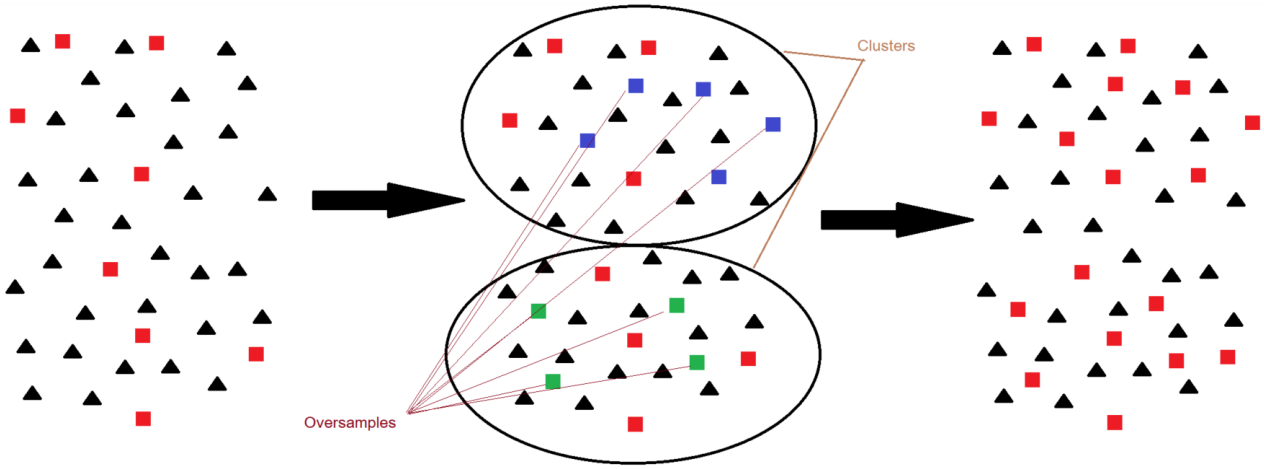


Figure 3.6: Cluster based oversampling adapted from Pérez-Ortiz et al. (2015)

3.4 Variable selection and Grouping

Figure 3.7: Variable selection and grouping

Siddiqi (2017) recommends that a scorecard should have between five and fifteen variables, thereby presenting a challenge of selecting only the "best" variables for the scorecard. The complexity of choosing variables can be compounded by the number of variables that are initially available. This can be solved by the concept of variable selection, where variables are ranked and selected based on a predetermined criteria, for instance, WOE, IV, correlation coefficient, and Chi-square test of independence, all applied one after the other. This section discusses the different variable selection methods that are used in this study.

3.4.1 Weight of Evidence (WOE)

The weight of evidence (WOE) evaluates the predictive power of a variable compared to the outcome. This measure is a useful tool for binning or grouping variables. The calculation of the WOE measure is relatively intuitive for categorical variables because these variables are, in a way, already grouped, as compared to continuous variables, which need to be grouped into intervals first. Groups with similar WOE are grouped together as this indicates equal risk. WOE converts the probability of a default occurring into a linear scale and has a linear relationship with the logistic function:

$$WOE_i = ln\left(\frac{TotalGoods_i}{TotalBads_i}\right), \tag{3.1}$$

where "Total Bads" is defined as the total number of bad observations in a group i in a given variable and "Total Goods" is defined as the total number of good observations in the same group. The only assumption for this measure is that for each of the groups within a variable, there are observations comprising both "goods" and "bads".

3.4.2 Information Value

Information Value (IV) focuses more on the distribution of the variable. IV is more commonly used for selecting variables when developing LR scorecards as shown by Siddiqi (2017). An IV of 0 implies that the variables offer no statistical explanation of the outcome variable. There is no upper limit for IV, but the higher the IV, the better the probability of the variable being a good variable for the scorecard. In a selected dataset, the variable with the highest IV will be selected first, and this will be the benchmark. If one is required to develop a scorecard comprising of p variables, then the top p variables ranked by IV will be selected. IV is calculated by the following formula:

$$IV = \sum_{i=1}^{g}\left((\%Goods_i - \%Bads_i) * ln\left(\frac{\%Goods_i}{\%Bads_i}\right)\right) = \sum_{i=1}^{g}((\%Goods_i - \%Bads_i) * WOE_i), \tag{3.2}$$

where the variable has g groups.

3.4.2.1 WOE and IV example

The calculation of WOE and IV for the Age variable from Figure 2.1, is shown in Table 3.1. The variable has four groups, and hence $g = 4$. It can be concluded that bins in the variable are optimal since the WOE is monotonic. If this was not the case, the variable would need to be regrouped. This section shows the calculation of IV and WOE for the Age variable in Figure 2.1. IV is calculated for the individual variable and WOE is calculated for the bins in the variable. The WOE is monotonic from the lower intervals to the higher ones. The overall IV for this variable is 0.64120. This will be compared to IV's from other variables.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Age Group	**Count**	**Bin %**	**# Goods**	**# Bads**	**Bad Rate**	**Good %**	**Bad %**	**WOE**	**Bin IV**
Other	75	2.99%	52	23	30.67%	2.29%	9.39%	1.40876	0.09992
[18;20]	600	23.89%	492	108	18.00%	21.71%	44.08%	0.70817	0.15841
[21;35]	836	33.29%	752	84	10.05%	33.19%	34.29%	0.03259	0.00036
[36;60]	1 000	39.83%	970	30	3.00%	42.81%	12.24%	-1.25159	0.38251
Total	2511	100%	2266	245		100%	100%		0.64120

Table 3.1: WOE and IV example

Some of the column values for each row are calculated by:

$$BadRate\,(5) = \frac{\#\,Bad\,(4)}{Count\,(1)}\,, \tag{3.3}$$

$$Good\,\%\,(6) = \frac{\#\,Good\,(3)}{\sum \#\,Good\,(3)}\,, \tag{3.4}$$

$$Bad\,\%\,(7) = \frac{\#\,Bad(4)}{\sum \#\,Bad\,(4)}\,, \tag{3.5}$$

$$WOE\,(8) = ln\left(\frac{Good\,\%\,(6)}{Bad\,\%\,(7)}\right)\,, \tag{3.6}$$

$$IV\,(9) = WOE\,(8) * (Good\,\%\,(6) - Bad\,\%\,(7))\,, \tag{3.7}$$

and the total row for 9 is calculated by summing column 9.

3.4.3 Pearson Correlation Coefficient

References to the correlation coefficient in this study mean the Pearson Correlation Coefficient. This measures the similarity, or linear association, between two variables. The variables can either be continuous or categorical. The correlation coefficient takes on scaled values that range from -1 to 1. The association between the examined variables gets stronger as the correlation coefficient approaches the value of 1 in absolute. The association gets weaker as the coefficient approaches 0, and the value of 0 shows that there is no linear association between the tested variables. Table 3.2 below shows the interpretation of the Pearson correlation coefficient.

	Direction of association	
Strength of association	**Positive**	**Negative**
Weak	0.1 to 0.3	-0.1 to -0.3
Medium	0.3 to 0.5	-0.3 to -0.5
Strong	0.5 to 1.0	-0.5 to -1.0

Table 3.2: Pearson Correlation Interpretation adapted from Ratnasari et al. (2016)

When the correlation coefficient is employed to select variables to use in a scorecard, the correlation coefficient between each of the variables (the examined variables) and the outcome variable is calculated and ranked. The selection of variables can be carried out by performing hypothesis testing on the correlation coefficients. Siddiqi (2017) shows that the desirable variables will have the lowest absolute correlation coefficient to the outcome variable. When there is strong association, it means that the examined variable might be sufficient to explain the outcome variable and hence there won't be a need to develop the scorecard. Variables that are highly correlated to the outcome variable will also lead to overfitting of the scorecard. Overfitting is a type of modelling error that occurs when the predicted function is too similar and fits only the underlying data. The Pearson correlation coefficient ρ between two variables can be calculated as:

$$\rho_{XY} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}\,, \tag{3.8}$$

where Y and X are variables in the dataset. Equation 3.8 can be expanded to:

$$\rho_{XY} = \frac{n(\sum_{i=1}^{n} X_i Y_i) - (\sum_{i=1}^{n} X_i)(\sum_{i=1}^{n} Y_i)}{\sqrt{n\sum_{i=1}^{n} X_i^2 - (\sum_{i=1}^{n} X_i)^2}\sqrt{n\sum_{i=1}^{n} Y_i^2 - (\sum_{i=1}^{n} Y_i)^2}}\,, \tag{3.9}$$

even though Y_i in Equation 3.9 can be any variable in the dataset, when selecting variables initially, it will be the outcome variable. Then, the correlation coefficient is also calculated between the variables as part of variable selection. If two variables are highly correlated, then it is enough to consider only one of them for the modelling procedure. This is because highly correlated variables will offer a similar contribution to the scorecard. The Chi-square test of independence and k-means clustering shown in the sections below are used to decide which variable should be chosen.

3.4.3.1 Pearson Correlation Coefficient calculation

The example below demonstrates the calculation of the Pearson correlation coefficient between age and income. Data from only ten customers is used for the practicality of the example. Since all the variables are grouped, the WOE for each group is used as a representation for that group as seen in Table 3.3:

Customer	**Age**	**Age WOE**	**Income**	**Inc WOE**	**(Inc WOE)(Age WOE)**	**(Age WOE)^2**	**(Inc WOE)^2**
1	[18;20]	-0.70817	[R250k; R500k)	0.05613	-0.03975	0.50150	0.00315
2	[36;60]	1.25159	[R500k; R750k)	0.57079	0.71440	1.56647	0.32580
3	[21;35]	-0.03259	<R250k	-0.17749	0.00578	0.00106	0.03150
4	[21;35]	-0.03259	[R500k; R750k)	0.57079	-0.01860	0.00106	0.32580
5	[21;35]	-0.03259	>= R750k	2.56165	-0.08349	0.00106	6.56207
6	[36;60]	1.25159	[R250k; R500k)	0.05613	0.07025	1.56647	0.00315
7	[36;60]	1.25159	>= R750k	2.56165	3.20613	1.56647	6.56207
8	[21;35]	-0.03259	[R500k; R750k)	0.57079	-0.01860	0.00106	0.32580
9	[36;60]	1.25159	<R250k	-0.17749	-0.22214	1.56647	0.03150
10	[36;60]	1.25159	[R500k; R750k)	0.57079	0.71440	1.56647	0.32580
Total		**5.41939**		**7.16376**	**4.32836**	**8.33808**	**14.49666**

Table 3.3: Correlation Calculation

Based on Table 3.3, the coefficient can be computed as :

$$\rho_{AgeIncome} = \frac{10(\sum_{i=1}^{10}(Age\,WOE)_i(Income\,WOE)_i) - (\sum_{i=1}^{10}(Age\,WOE)_i)(\sum_{i=1}^{10}(Income\,WOE)_i)}{\sqrt{n\sum_{i=1}^{10}(Age\,WOE)_i^2 - (\sum_{i=1}^{10}(Age\,WOE)_i)^2}\sqrt{10\sum_{i=1}^{10}(Income\,WOE)_i^2 - (\sum_{i=1}^{10}(Income\,WOE)_i)^2}}\,, \tag{3.10}$$

$$\rho_{AgeIncome} = \frac{10(4.32836) - (5.41939)(7.16376)}{\sqrt{54.01101}\sqrt{93.64654}} = \frac{4.46039}{(7.34921)(9.67715)} = 0.06271\,. \quad (3.11)$$

The Pearson correlation coefficient for the above data is 0.06271, which suggests that there is a small or no linear association between the income and age of an individual in this dataset. Hence, it can be concluded that both variables can be included in the development of the scorecard.

3.4.4 Chi-square test of independence

This is a statistical test applied to categorical or internally binned continuous variables. When a continuous variable like Age is binned into age intervals, then the age intervals and the outcome variable are tested for independence. The data is displayed in a two-way frequency table where each column represents a category for age intervals and each row shows a category for the outcome variable. The rows and columns can be interchanged and the conclusion will be consistent. The null hypothesis will be that Age and the outcome are not related. The alternative hypothesis is that there exists a dependence between the two variables.

When there are two or more dependent variables, only one is chosen by means of a hypothesis test at a specific significance level.

$$\begin{aligned} H_0: &\quad Variable\ A\ and\ Variable\ B\ are\ independent \\ H_A: &\quad Variable\ A\ and\ Variable\ B\ are\ not\ independent\,. \end{aligned} \quad (3.12)$$

The expected value is:

$$e_{i,j} = \frac{row\ i\ total\ *\ column\ j\ total}{n}\,, \quad (3.13)$$

and the Chi-Square Test of independence can be calculated as below:

$$\mathcal{X}^2_{(c-1)(r-1)} = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{(o_{i,j} - e_{i,j})^2}{e_{i,j}}, \tag{3.14}$$

where there are c, r categories in the respective variables being compared, that is r rows and c columns and $o_{i,j}$ is the observed cell count in the i^{th} row and j^{th} column of the frequency table. The value from Equation 3.14 is compared to the upper-tail critical values in the Chi-square table with $(c-1)(r-1)$ degrees of freedom. The null hypothesis is rejected if the test statistic value is greater than the Chi-square statistic with $(c-1)(r-1)$ degrees of freedom ($\mathcal{X}^2_{(c-1)(r-1)}$) at a given level of significance. Another option to draw a conclusion is to compute the $p - value$ for the calculated statistic. The $p - value$ gives the level of significance of the calculated statistic. p-values larger than 0.2 would indicate insufficient evidence in favour of the alternative hypothesis that the variables are dependent

3.4.4.1 Chi-square example calculation

The example below demonstrates the Chi-square test of independence performed on the variables age and income. The null hypothesis is that: age and income are independent. Table 3.4 shows the observed values as they are in the sample. The next step is to calculate the expected values.

Income	**Age**				
	Other	**[18;20]**	**[21;35]**	**[36;60]**	Total
No income	4	30	41	50	**125**
< R250k	32	258	355	430	**1 075**
[R250k; R500k)	23	186	266	310	**785**
[R500k; R750k)	14	108	149	180	**451**
>= R750k	2	18	25	30	**75**
Total	**75**	**600**	**836**	**1 000**	**2 511**

Table 3.4: Observed values

The expected values are calculated by Equation 3.13. The expectation in the first cell is:

$$e_{i,j} = \frac{row\, i\, total \ * \ column\, j\, total}{n} = \frac{125 * 75}{2511} = 3.73357, \tag{3.15}$$

and so on, resulting in Table 3.5.

Income	**Age**				
	Other	**[18;20]**	**[21;35]**	**[36;60]**	**Total**
No income	3.73357	29.86858	41.61689	49.78096	**125**
<= R250k	32.10872	256.86977	357.90522	428.11629	**1 075**
[R250k; R500k)	23.44683	187.57467	261.35404	312.62445	**785**
[R500k; R750 k)	13.47073	107.76583	150.15372	179.60972	**451**
>= R750 k	2.24014	17.92115	24.97013	29.86858	**75**
Total	**75**	**600**	**836**	**1 000**	**2 511**

Table 3.5: Expected values

Table 3.5 can now be used to calculate the scaled residuals and the scaled residual for the first cell is now:

$$\frac{(o_{i,j} - e_{i,j})^2}{e_{i,j}} = \frac{(4 - 3.73357)^2}{3.73357} = 0.01901\,, \tag{3.16}$$

and so on, resulting in Table 3.6. The Chi-Square statistic can now be calculated as the sum of all the cells.

Income	**Age**				
	Other	**[18;20]**	**[21;35]**	**[36;60]**	**Total**
No income	0.01901	0.00058	0.00914	0.00096	**0.02970**
<= R250k	0.00037	0.00497	0.02358	0.00829	**0.03721**
[R250k; R500k)	0.00852	0.01322	0.08259	0.02203	**0.12636**
[R500k; R750 k)	0.02080	0.00051	0.00886	0.00085	**0.03102**
>= R750 k	0.02574	0.00035	0.00004	0.00058	**0.02670**
Total	**0.07443**	**0.01963**	**0.12422**	**0.03271**	**0.25099**

Table 3.6: Chi-Square residuals

The Chi-Square statistic for this example is 0.25099. The *p-value* for this test statistic at 12 degrees of freedom is greater than 0.99999 and hence it can be concluded that there is insufficient evidence in favour of H_A, that the variables are dependent. This conclusion is consistent with the conclusion of the Pearson correlation. Therefore, both variables (Age and Income) can be included in the list of variables used to develop the scorecard.

3.4.5 k-means clustering

This method partitions the dataset into k clusters of variables. Variables are assessed on how they relate to the outcome variable and the amount of variation explained. The algorithm goes on to select k cluster centres and then refines the cluster centres by iterations until convergence or acceptable clusters are achieved. The acceptable number of clusters is determined by the required number of variables to develop a scorecard. The iterations ensure that the best variable in each cluster is chosen for the model. The best variables will be determined by their distance to the cluster center. This method is unsupervised in that only the k initial parameters are defined and the iterations are automatic. The k initial parameters will be the cluster centres, but if the the results are not acceptable, another set of k clusters centres can be selected and the algorithm can be run again. There is a risk that this method is sensitive to the initial parameters (the chosen k cluster centres), as seen in Figure 3.8 below. In Figure 3.8, the first two part of the figure shows how the two cluster centroids can be selected and how these choices can lead to different outcomes.

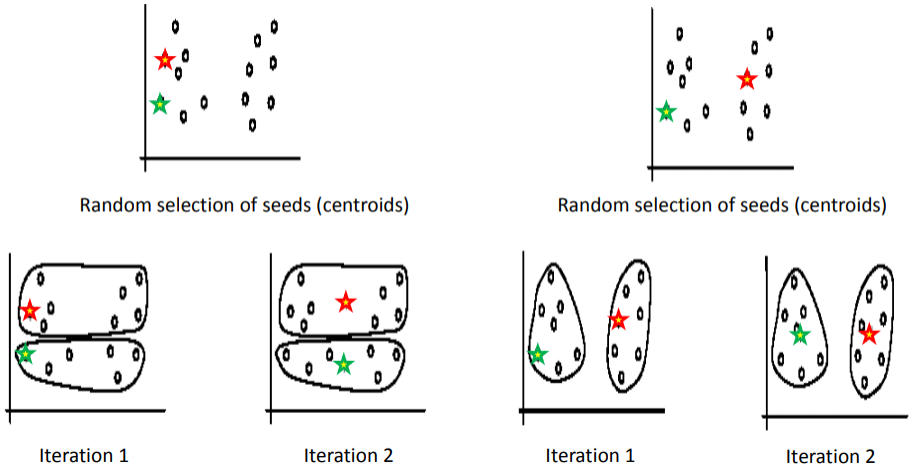


Figure 3.8: Sensitivity to Initial conditions adapted from Zhang et al. (2017)

The $k-$means algorithm is typically used to classify observations and not variables. However the algorithm is expanded also classify variables into clusters. Then variables in each cluster are compared with each other and only one is selected. The aim is to have each cluster represented in the final scorecard.

3.5 Statistical Modelling methods

Figure 3.9: Modelling

The literature review for the following modelling methods is included below: Logistic regression, Probit regression, Poisson Regression, and Linear Discriminant analysis.

3.5.1 Logistic Regression

Logistic Regression is a popular tool for credit scoring models. Abdou and Pointon (2011) showed that logistic regression utilises a dichotomous (0/1 i.e binary) dependent variable. It works by estimating and assigning numerical weights to variables involved in order to predict the probability of default. For all the methods the predicted outcome will be assigned the value of 1 if the probability of default is greater or equal to 0.5 and 0 otherwise. The probability of default given the data can be calculated by the following equation:

$$p = P(Outcome = 1|Data) = \pi(x) = \frac{e^{\beta_0 \ + \sum_{i=0}^{q} X_i\beta_i}}{1 + e^{\beta_0 \ + \sum_{i=0}^{q} X_i\beta_i}}\,, \tag{3.17}$$

rearranging this equation gives

$$\frac{p}{1-p} = e^{\beta_0 \ + \sum_{i=0}^{q} X_i\beta_i}\,, \tag{3.18}$$

but $\frac{p}{1-p} = odds$ and to get to the linear term of the regression function a logit transformation is needed

$$ln\Big(\frac{p}{1-p}\Big) = ln(odds) = \beta_0 \ + \sum_{i=0}^{q} X_i\beta_i\,, \tag{3.19}$$

where there are q variables. The set of weights β_i for $i \in (1, 2, 3...q)$ will be estimated by the

model algorithm via maximum likelihood estimation (MLE). The vector of input variables is the X_i's. This function is continuous and linear in its parameters. Equation 3.17 can also be rewritten in the following form:

$$p = P(Outcome = 1|Data) = \frac{1}{1 + e^{-(\beta_0 \ + \sum_{i=0}^{q} X_i \beta_i)}} \,. \tag{3.20}$$

The output from Equation 3.20 is the probability of default (PD), which can be converted to a score using Equation 3.21, introduced by Siddiqi (2017). The equation can be simplified to determine the values for offset and factor. The offset can be seen as the intercept and the factor as the gradient if Equation 3.21 is compared to a straight line equation.

$$Score = offset + Factor * ln(Odds) = offset + Factor * ln\Big(\frac{p}{1-p}\Big) \,. \tag{3.21}$$

Setting the $offset$ and $Factor$ in the above formula requires a combination of business rules set out by the Bank developing the scorecard. The reader is refereed to Siddiqi (2017) for further detail in the estimation of the offset and Factor.

3.5.2 Probit Regression

Probit regression, developed in the early 1930's, was one of the first methods to be used for credit scoring, as shown by Abdou and Pointon (2011). The data is inherently assumed to be normal since the underlying theory behind the probit model is the normal distribution. Equation 3.22 below shows how the PD is calculated for a probit model.

$$P(Outcome = 1|Data) = \Phi^{-1}(p) = \beta_0 \ + \sum_{i=1}^{q} \beta_i X_i \,, \tag{3.22}$$

and with transformation the equation can be re-written as:

$$P(Outcome = 1|Data) = 1 - \Phi\Big(- \frac{\sum_{i=0}^{q} X_i \beta_i}{\sigma}\Big) \,, \tag{3.23}$$

where:

$$\Phi = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\beta_0+\beta X} e^{-\left(\frac{z^2}{2}\right)} dz \,, \tag{3.24}$$

and Φ shows the Probit transformation of the predicted values, i.e the link function and the -1 superscript shows the inverse of the CDF of the standard normal distribution. The vector of input variables is X_i . The set of weights, i.e β, will be estimated by the model algorithm, and Equation 3.21 will be used to convert the PD to scores. Chasco, Aroca and Anselin (2019), Abdou and Pointon (2011) and Ren, Hou and Li (2013) note that Probit models must be implemented with caution as they are prone to multicollinearity.

3.5.3 Poisson Model

According to Henley (1995) and Karlis and Rahmouni (2007), the Poisson distribution is commonly used to model the number of events. The number of loan applications per customer or daily customer visits to a bank branch are examples of such events. The Poisson model can be changed to forecast the PD in credit scoring. The customer's PD will be derived from a Poisson distribution with parameter u, according to the model. This will be defined as the likelihood of the outcome occurring at least once during the time period specified i.e the number of default. However, this research is interested in at least one default occurring or no defaults occurring. Equation 3.26 is derived from Equation 3.25.

$$P(Outcome = y_i) = \frac{u^{y_i} e^{-u}}{y_i!} \,, \tag{3.25}$$

where even though y_i is the number of defaults and can be more than 1 default, the interest is in the first default and Equation 3.26 modifies this to predict only one default as:

$$P(Outcome = 1|Data) = 1 - P(Outcome = 0|Data) = 1 - \frac{u^0 e^{-u}}{0!} = 1 - e^{-u} \,, \tag{3.26}$$

where the outcome value can either be 1 or 0 and u is the Poisson parameter which can be

estimated using a log-linear link function in the form:

$$ln(u) = \beta_0 + \sum_{i=1}^{q} X_i \beta_i \,, \tag{3.27}$$

where X_i and is a vector of q input variables and β_i for $i \in (1, 2, 3...q)$ is a set of corresponding coefficients or weights for the variables and β_0 is the intercept. The coefficients as in Logistic regression are estimated by maximum likelihood estimation (MLE). The estimated mean of the Poisson distribution is u. The Poisson model is related to the logistic model, in that both models make use of the exponential function.

3.5.4 Linear Discriminant Analysis

Hooman et al. (2016) and Henley (1995) explain that Linear Discriminant Analysis (LDA) is an algorithm that classifies data using the similarities in the selected variables. The goal is to find a linear combination of the chosen variables that will distinguish between potentially excellent and potentially poor clients using discriminant criteria. This method aims to find optimal weight values (β_i), for the linear combination;

$$Z_i = \beta_0 + \sum_{i=1}^{q} X_i \beta_i \,, \tag{3.28}$$

where Z_i is the discriminant score. Like before, the β_i for $i \in (1, 2, 3...q)$ is a set of corresponding coefficients or weights for the variables and β_0 is the intercept. Suppose the dataset is in two classes, say c_1 and c_2. In this study c_1 is a class that contains good customers and c_2 is the class containing bad customers. The LDA model assumes that both classes are multivariate normal with means u_1 and u_2 and the variance matrix ν. The LDA model will then classify the i^{th} customer with q-dimensional vector X to c_2 if:

$$X^T \hat{\nu}^{-1}(\hat{u_1} - \hat{u_2}) > \frac{1}{2} u_2^T \hat{\nu}^{-1} \hat{u_2} - \frac{1}{2} u_1^T \hat{\nu}^{-1} \hat{u_1} + ln(\pi_1) - ln(\pi_2) \,, \tag{3.29}$$

where π_1 and π_2 are prior probabilities of falling into c_1 and c_2 respectively, and can be expressed as b and $1 - b$. Furthermore, the prior probabilities will be estimated from proportions in the training dataset. The output is also given as a probability of belonging to one of the groups, which in turn can be converted into a score using Equation 3.21.

3.6 Machine Learning Modelling Methods

The literature review for the following modelling methods is included below: Artificial Neural Networks, Support Vector Machines, and Random Forest Regression.

3.6.1 Artificial Neural Network

West (2000) and Pacelli and Azzollini (2011) showed that Artificial Neural Networks (ANNs) can be thought of as many combined logistic regression functions. The most common ANNs, multilayer perceptrons (MLPs) have three main layer types, the input layer, the hidden layer(s) and the output layer. All the layers have varying number of nodes which are referred to as neurons. The neurons in the input layer are all the chosen variables from the variable selection steps in the scorecard building process. Figure 3.10 shows the structure of an ANN, the input layer takes on the variables as inputs and the output layer produces the probabilities.

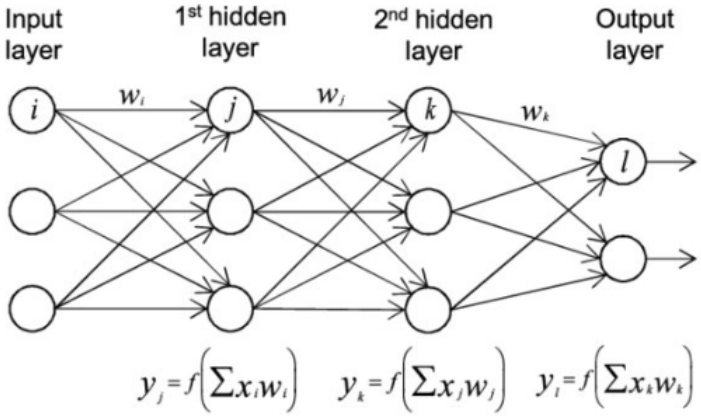


Figure 3.10: ANN Example adapted from Gillani and Zubair (2016)

The layers are connected by weights that improve by iterations as the algorithm gets more data. The underlying theory behind ANN is that the model learns and improves as it is given more data. The trade off for the choice of the number of hidden layers and neurons is the model's training time. A model's training time is the time taken to provide an output when data is loaded. The output layer contains the probability of default and can be converted to score points. The PD from a MLP with three layers, q variables, and r neurons in the hidden layer can be represented by the following formula;

$$P(Outcome = 1|Data) = \sum_{h=1}^{r} w_j(g(\sum_{j=1}^{q}\sum_{i=1}^{q} w_{ij}X_i) + b_0) + b_1\,, \tag{3.30}$$

where b_0 and b_1 are constant weight biases for the input and hidden layer respectively and the output layer does not require a bias input. The number of constant biases is related to the number layers in the MLP, that is; $m-1$ biases would be needed when there are m layers. Equation 3.30 becomes complicated to write out as the number of hidden layers and neurons are increased. When the model is being trained for every iteration and incorrect response, the weights are updated to improve the overall model based on the variance of the output.

$$w^c = w^{c-1} + \eta(y - \hat{y}^{c-1})\mathbf{x}\,, \tag{3.31}$$

where η is the learning rate, w^c is the weight vector for the c^{th} iteration, $\mathbf{x}$ is a vector of input variables of length q, y is the expected output and $\hat{y}^c$ is the estimated output of the c^{th} iteration.

3.6.2 Support Vector Machines

Van Gestel et al. (2003) described the Support Vector Machine (SVM) algorithm as a separation technique. The SVM algorithm separates the groups with the maximum distance between them, this distance is referred to as the margin. SVMs work by assuming non-linear transformation of the data and minimising the estimation error. The population set S can be represented as two samples c_1 and c_2, when one group contains good customers and the other contains bad customers. Class c_1 contains good customers and is shown by 1 and c_2 contains bad customers and even though it is usually shown by 0, for better computations in the SVM algorithm bad clients are shown by -1. Figure 3.11 shows a visual representation of how the SVM algorithm aims to classify the data into two classes. The two classes are separated by a hyperplane and margins on each side of the hyperplane. The further away that the observations are from the margins, the higher the probability of the predicted observation actually belonging to that class.

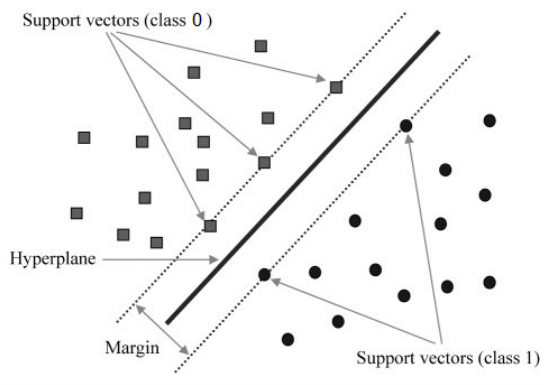


Figure 3.11: SVM Example adapted from Van Gestel et al. (2003)

The probability for class $x \, \epsilon \, c_1$ can be written out as

$$g(x) = \sum_{j=1}^{q} a_j K(x, x_j) + b \,, \tag{3.32}$$

with

$$K(x, x_j) = exp(-\frac{|x - x_j|^2}{2\sigma^2}) \,, \tag{3.33}$$

and

$$y = \begin{cases} 0, & if \; g(x) \; < \; 0.5 \\ 1, & if \; g(x) \; \geqslant \; 0.5 \end{cases}, \tag{3.34}$$

where b is the intercept and x can belong to either c_1 or c_2, $K(x, x_j)$ is the Gaussian kernel for the two observations, the a_j's are the Lagrange multipliers and are more significant for observations closer to the boundaries of classes and lastly y is the output showing which class x belongs to. The score can be derived from the probability as shown in Equation 3.21.

3.6.3 Random Forest Regression

Random Forest Regression (RFR) is a machine learning classification technique. This algorithm fits multiple decision trees to a number of samples of the data. A decision tree is defined as a decision support tool that uses a tree-like model of decisions. In conventional

decision trees, every node is divided using all the features. The RFR algorithm differs here by using a subset of features that were randomly selected at the particular node.

Segal (2004) shows that when modelling, the prediction for a random forest is the unweighted mean over the following collection:

$$\overline{f}(\mathbf{x}) = \frac{1}{q}\sum_{i=1}^{q} f(\mathbf{x};\beta_i)\,, \tag{3.35}$$

where $f(\mathbf{x};\beta_i), i = 1, 2, ...q$ is a collection of decision tree predictors, $\mathbf{x}$ is a vector of input variables of length q, as there are q variables and there are q random vectors, denoted by β_i, which are independent and identically distributed. Segal (2004) shows that the outcome variable y can be predicted using Equation 3.35, and q has to be taken to infinity.

As $q \to \infty$ then:

$$E_{\mathbf{x},y}(y - \overline{f}(\mathbf{x}))^2 \to E_{\mathbf{x},y}(y - E_{\beta} f(\mathbf{x};\beta))^2\,. \tag{3.36}$$

The algorithm converges as seen in Equation 3.36, the convergence then helps to control over-fitting and improve the predictive accuracy. The RFR algorithm can be used for Regression, classification or optimisation.
The process for the RFR algorithm entails:

1. Drawing n samples from the training dataset.
2. For each of the samples construct a regression decision tree. Then obtain the prediction results from every decision tree.
3. Score new data by aggregating the predictions of the n trees. For regression, the average for all trees is used.

The following diagram (Figure 3.12) illustrates the algorithm:

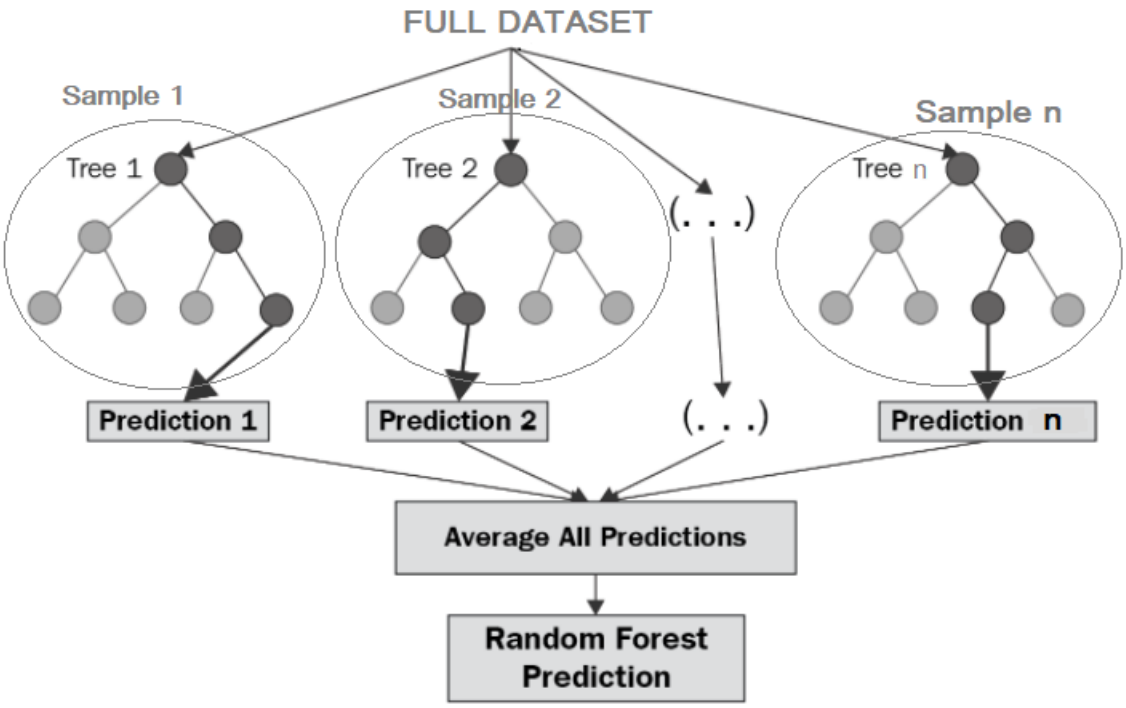


Figure 3.12: Random Forest algorithm adapted from Yang et al. (2019)

3.7 Survival Modelling

Survival analysis primarily models time (T) to failure. In credit scoring this is the time until a default on the loan occurs. Man (2014) shows that the unpredictability of T can be described in three standard ways, namely the survival function $S(t)$, the density function $f(t)$ and the hazard function $h(t)$. The survival function, $S(t)$, gives the probability that the time to default (T) is greater compared to a time (t). The density function $f(t)$ shows the probability that the default occurs at exactly time t. Finally the hazard function $h(t)$ shows the probability that if a customer does not default between time 0 and time t, then the customer will default in the next instant.

If the distribution function of T is defined as $F(t) = P(T \leqslant t)$ then $S(t)$, $f(t)$ and $h(t)$ can be computed as:

$$S(t) = 1 - F(t)\,, \tag{3.37}$$

$$f(t) = \lim_{\delta t \to 0} \frac{P(t \leqslant T \leqslant t + \delta t)}{\delta t}\,, \tag{3.38}$$

$$h(t) = \lim_{\delta t \to 0} \frac{P(t < T \leqslant t + \delta t | T > t)}{\delta t} = \frac{f(t)}{1 - F(t)} = \frac{f(t)}{S(t)}\,. \tag{3.39}$$

This has advantages as the three standard describing functions (Equations 3.37, 3.38 and 3.39) are fully defined. This study investigates a survival model fitted with an exponential distribution with parameter u. Hence $f(t)$, $S(t)$, $F(t)$ and $h(t)$ become:

$$f(t) = ue^{-ut}\,, \tag{3.40}$$

$$F(t) = 1 - e^{-ut}\,, \tag{3.41}$$

$$S(t) = e^{-ut}\,, \tag{3.42}$$

$$h(t) = u\,. \tag{3.43}$$

3.8 Scorecard Evaluation

Figure 3.13: Evaluation

A scorecard's performance can be assessed in various ways. The aim is to select a scorecard that best separates the good and bad clients. The performance of all the scorecards that are investigated through this research will be measured using Kolmogorov-Smirnov Statistic (KS), Gini coefficient, confusion matrix and the area under receiver operating characteristic curve (ROC). Before the scorecards are tested for the OOT performance, the variables from

both samples, the BVT and OOT sample, will be check for stability using the Stability Index.

3.8.1 Kolmogorov-Smirnov (KS) Statistic

The Kolmogorov-Smirnov (KS) goodness of fit test compares the maximum vertical deviation between the empirical distribution of good and bad customers. This statistic measures the deviation at only one point, where the separation is at its maximum as seen in Figure 3.14 below. The distribution of good and bad customers can be represented as the outcome variable and is defined:

$$D_{KS} = \begin{cases} 0, & if\ the\ customer\ is\ good \\ 1, & if\ customer\ is\ bad \end{cases}, \tag{3.44}$$

$$F_{b,BAD}(x) = \frac{1}{b}\sum_{j=1}^{b} I(s_j \leqslant x \wedge D_{KS} = 1)\,, \tag{3.45}$$

$$F_{g,GOOD}(x) = \frac{1}{g}\sum_{j=1}^{g} I(s_j \leqslant x \wedge D_{KS} = 1)\,, \tag{3.46}$$

$$KS = max_x |F_{b,BAD}(x) - F_{g,GOOD}(x)|\,, \tag{3.47}$$

where there are b bad clients in the population, g good customers, s_j is the score for the j^{th} customer and I is the indicator function that takes values of either 0 or 1. Figure 3.14 gives an example of how the KS value would be deduced from the cumulative probability plot. The scorecard with a higher relative KS value is the one that is preferred because it best separates the good from the bad customers. This means that it will be easier to separate the customers, which is the goal of credit scoring.

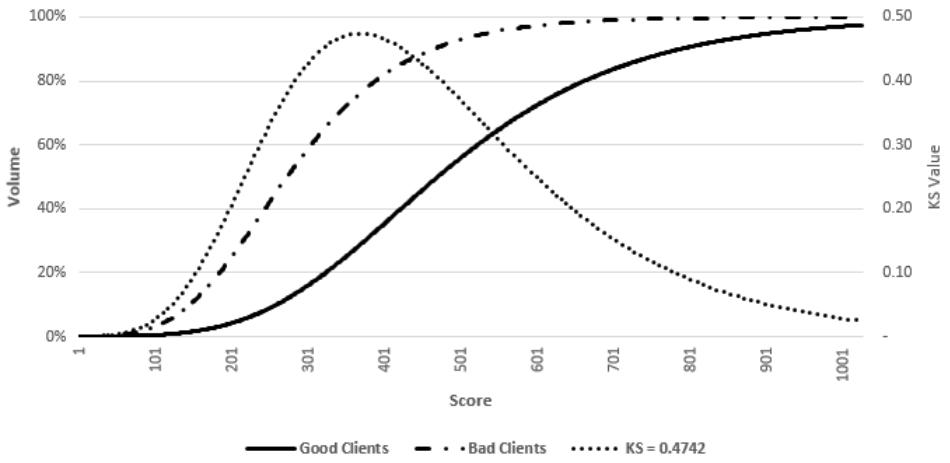


Figure 3.14: Kolmogorov-Smirnov statistic example adapted from Drezner et al. (2008)

3.8.2 Confusion matrix

A confusion matrix is a not necessarily a measure, but it tabulates the number of actual outcomes against the number of the predicted outcomes. In this research, the outcome variable is binary and there are four possible outcomes for each scored input; namely, true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN). This study looks at "bad" customers as positives and "good" customers as negatives, then P and N are respectively replaced by B and G. TB is the number of customers that are correctly predicted to be Bad whereas FB shows the number of customers that are incorrectly predicted to be bad. Similarly TG shows the number of customers that were correctly predicted to be good and FG shows the number of customers that were incorrectly predicted to be good. Multiple Statistical measures can be calculated from the confusion matrix. This study calculates Accuracy and Specificity.

		Predicted	
		GOOD (0)	**BAD (1)**
Actual	**GOOD (0)**	TG	FB
	BAD (1)	FG	TB

Figure 3.15: Confusion Matrix example adapted from Zeng (2020)

3.8.2.1 Accuracy

Accuracy shows how often the scorecard will be correct. It is calculated by summing the number of correct predictions divided by the total number of predictions.

$$Accuracy = \frac{TG + TB}{TOTAL} \,. \tag{3.48}$$

3.8.2.2 Specificity

Specificity shows that if the scorecard predicts a customer to be "Good", how often it is actually correct. The best value for specificity is 1 and the worst would be 0. It is calculated as the number of true "goods" divided by the total number of "goods" i.e. true "goods" and false "bads" as shown of Equation 3.49.

$$Specificity = \frac{TG}{TG + FB} \,. \tag{3.49}$$

3.8.3 Gini coefficient

The Gini coefficient describes the global quality of a scoring function. It has an output range of (−1,1). A perfect scorecard i.e. one that accurately separates good and bad customers, has a Gini coefficient of 1. A Gini coefficient of 0 indicates that the scorecard randomly classifies customers. Negative values of the Gini coefficient show that the scorecard assigns scores inaccurately. The coefficient can be calculated for the overall scorecard and also on an individual variable basis, and this means that it can be utilised for variable selection. The higher the Gini the better the predictive ability of the variable and the better the scorecard can separate between good and bad customers. Equation 3.50 below shows how the Gini coefficient is calculated.

$$Gini = 1 - \sum_{i=1}^{m} \left[\left(F_{BAD_i} - F_{BAD_{i-1}} \right) \left(F_{GOOD_i} - F_{GOOD_{i-1}} \right) \right] , \tag{3.50}$$

where F_{BAD_i} and F_{GOOD_i} are the i^{th} vector values of the empirical distribution function of bad and good clients respectively. Siddiqi (2017) gives a comprehensive review of this formula. The Gini coefficient can also be calculated for each variable to assess the variable's

predictive power. The value of the Gini coefficient is the area A in Figure 3.16.

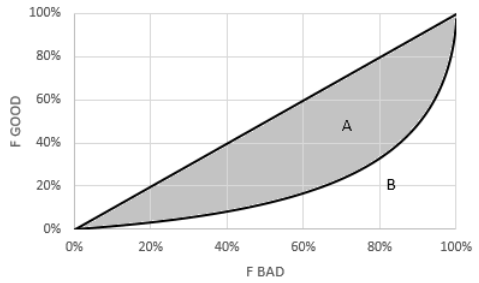


Figure 3.16: GINI example adapted from Bijak and Thomas (2012)

3.8.4 Area Under Receiver Operating Characteristic Curve (ROC)

A receiver operating characteristic curve, or ROC curve, is a plot that shows the diagnostic ability of binary classifiers. The ROC curve is plotted on a 2 dimensional plane, where the true positive rate is on the Y-axis and the false positive rate is on the X-axis. The area under the ROC curve (AUC) provides a combined measure of performance across all possible classification thresholds of the true positive (TP) and false positive (FP) rates. A perfect model will have an AUC of 1. Figure 3.17 shows the different values for the AUC.

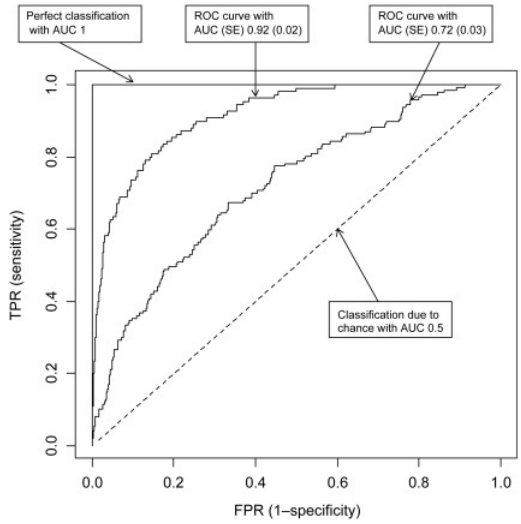


Figure 3.17: ROC curve example adapted from Habibzadeh et al. (2016)

3.8.5 Variable Stability Index

Since the OOT sample is used to evaluate the performance of scorecards developed by the BVT sample, it is desirable that the variables in the OOT sample is stable and hence can be compared to those of the BVT sample. Siddiqi (2017) gives a formula to calculate the stability index for a variable as:

$$Stability\ Index = \sum_{i}^{g} \Big((BVT\_\%)_i - (OOT\_\%)_i \Big) \times ln\Big(\frac{(BVT\_\%)_i}{(OOT\_\%)_i} \Big) \times 1000\,. \tag{3.51}$$

Where g is the number of bins in the variable, $(BVT\_\%)_i$ is the percentage in bin i of the variable in the BVT sample and $(OOT\_\%)_i$ is the percentage in bin i of the variable in the OOT sample.

The scale to interpret the stability index is shown in Table 3.7. Variables with a stability index below 100 imply that the variables in the OOT sample are stable when compared to the BVT sample and can be used to test the scorecard(s). If the stability index is greater that 249 means that the distribution within variables has shifted and that the grouping within the variable will need to be reconsidered. Those variables that show a slight shift, a further investigation may take place but does not necessarily mandate a re-grouping of the variables.

Stability index	**Outcome**	**Comment**
<100	No Shift	No Concern
100 - 249	Slight shift	Monitor variable closely
250 +	Servere shift	Variable may need re-grouping

Table 3.7: Stability Index interpretation

3.9 Summary

This study investigates a number of different methods and compares them. A scorecard is classified as best performing compared to the other scorecards if it maximises performance for most of the metrics. The OOT performance will be assessed separately from the development performance. A desirable OOT performance will be one for which the OOT metrics are stable and improve when compared to the development metrics.

Chapter 4

Data

4.1 Introduction

This chapter reviews the data to be used in developing the scorecard. The source of the data is given as well as the composition of the data.

4.2 Data

This research utilises real-world open source credit scoring data. The data was obtained from the data science website Kaggle (www.kaggle.com). The data was posted on Kaggle to solve a question of the length of time it took a customer to default. This research uses the data to develop account management scorecards to predict the probability of default and not the time to default, hence posing no conflict. The data contains loan details for customers of a provider named The Lending Club and does not contain any customer-identifying features. All loans are short-term loans with a repayment term of 3 years.

The full dataset has monthly data for the period of 2013 to 2018 and has a total of 1.6 million records. Each record in the dataset is a granted loan. The full list of variables can be seen in Table A.1 in the appendix. Out of the total dataset, 10.61% (162 893) is classified as "bad" and hence this is the default rate. The default rate needs to be consistent in the

sample(s) utilised in the actual scorecard development. There are about 150 variables for each customer in the original dataset. Table 4.1 shows the summary of the dataset used by this research. The loans are of a three year term and the default. The three variables that are completely empty and are not useful to this study.

	Lending Club Dataset
Type	Customer Loan Data
Loan Term	3 years
# Observations	1 534 750
# Variables	147
# Empty Variables	10
# Defaults	162 893
% Defaults	10.61 %

Table 4.1: Data Summary

4.3 Data cleaning

The dataset has a mixture of 80 categorical and 70 quantitative variables. The variables are shown in the Appendix. Since this study will be developing scorecards, all the quantitative variables will be binned and treated as categorical. Only three variables are not completely populated, namely Identity number (ID), Member name (Member ID) and Address. These variables would potentially contravene the anonymity of the data if populated, hence they were omitted from the source.

4.4 Summary

This dataset has properties that are desirable for developing AM scorecards. The individual customers in the dataset are unknown and untraceable. This study investigates the outlined machine learning and statistical methods of credit scoring using this dataset. The outcome variable is defined as default if the customer has missed an instalment date by more than 90 days. The data is historical and the scorecards will be developed as if future data is not available. The BVT data is historical and the future data is the OOT data. The OOT data will not be included in building any of the scorecards hence it is the future data. Thereafter, the future data will be used to compare the predicted output with the actual output.

Chapter 5

Sampling and Variable Preparation

5.1 Introduction

This chapter shows the process that was followed when dividing the data into the built, validation and testing samples (BVT samples) as well as the "out of time" sample. The scorecards are then developed using the BVT samples. The chapter concludes with the discussion on the preparation of variables. The scorecard development and analysis in this chapter are done using the Python packages: "Imbalanced-learn" developed by Lemaître, Nogueira and Aridas (2017), "Varclus" by Douzas, Bacao and Last (2018), "Deslib" by Cruz, Hafemann, Sabourin, and Cavalcanti (2020) and "Scikit-learn" by Pedregosa, Varoquaux, Gramfort, Michel et al. (2011).

5.2 Sampling

The data was divided by date into two sub-populations. The first sub-population is the data used to develop, validate and test the scorecards, also known as the BVT and the second will be used to assess how the scorecards perform over time known as the OOT data, all introduced in the earlier chapters. This study uses loan accounts granted between 2013 and 2017 as the BVT data and those granted during 2018 as the OOT data.

	Lending Club Dataset	BVT	OOT
# Observations	1 534 750	869 660	365 090
# Variables	147	147	147
# Empty Variables	3	3	3
# Defaults	162 893	125 198	20 701
% Defaults	10.61 %	14.40%	5.67%
Duration	2013 - 2018	2013 - 2017	2018

Table 5.1: BVT and OOT Data

Samples are now drawn from the BVT data using both machine learning and classical statistical sampling techniques. The Built samples will be used to train the scorecards. The Validation sample will be used to confirm that the scorecard can be applied to a subset of the population, that the variables are optimised and that the scorecard does not overfit the data. The Test sample is then used as an extra test for robustness of the scorecard over and above what the validation sample. The techniques used are CBOS, ROS and SMOTE. Three samples are produced for which the scorecards will be developed. Table 5.2 summarises the details of the sample datasets produced. The code used for each of the sampling methods is shown in the appendix. The number of observations in each sample is hard coded to 250 000. This is more than the minimum data needed to develop a credit scorecard. Figure 5.1 shows how the data is divided into the BVT and OOT samples, and how the BVT sample is further divided into the ROS, CBOS and SMOTE samples.

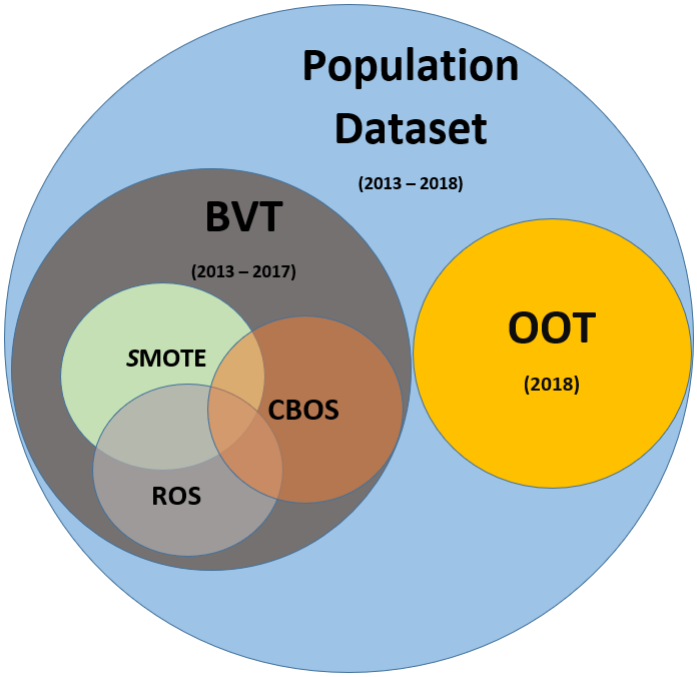


Figure 5.1: Data Sampling

It is possible that observations can appear in more than one sample since these sampling techniques are applied independently. It is also possible that some observations in the BVT samples do not appear in any of the samples. The samples are a mere representation and subset of the full population dataset and this process looks at which method will produce the best representation of the BVT samples. As seen in Table 5.2, there were three sample generated from the full BVT sample. Each sample was generated using the sampling methods chosen in an earlier chapter and each of the samples will be used to build similar scorecards as part of testing the impact of the sampling method on the overall performance of a scorecard.

	BVT	**ROS BVT**	**CBOS BVT**	**SMOTE BVT**
# Observations	869 660	250 000	250 000	250 000
# Defaults	125 198	50 000	57 701	56 674
% Defaults	14.40%	20.00%	23.10%	22.60%

Table 5.2: Sampled Data

5.3 Variable Selection

The number of variables in the population dataset makes it impossible to manually detect which variables may be useful in scorecard development. This study tackles variables with missing values first. Missing values in some of the variables will be grouped in their own group and be assigned a default score. Missing values can in fact provide useful information, as this may involve clients intentionally withholding information, and for this reason, missing does not automatically imply 0. Another example is that there are different implications to knowing with certainty that a client has no credit application compared to "not having the information", the missing group fall into the category of "not having the information". However, if the variable has missing values for more than 85% of the data, then the variable will be removed and not considered when developing the scorecard. Removing some variables is permitted because there are 147 variables to choose from. If the variable pool was limited, imputation methods for missing values would need to be employed. The list of all variables, their names and descriptions are shown in the Table A.1.

It is observed that three of the variables are not populated with any data, this means that the missing value rate for the the three variables is 100%, hence these variables will be removed along with ones which have a missing value rate of greater than 85%. Secondly variables with only two or less classes were also removed, this is done because the outcome variable is binary as well.

This study uses a combination of techniques to reduce the number of variables available. The techniques used are IV, chi square test of independence, k-means clustering and correlation.

The IV measure is advantageous because it can be calculated with raw data for both categorical and quantitative variables even before binning them. The variables will first be reduced using the IV measure by ranking them high to low, the rule of thumb is to have and IV of at least 0.01, this can be adjusted based on the data and the availability of variables. The IV statistic for the top 20 ranked variables is shown in Table 5.3 and the rest in Table A.2 in the Appendix.

Variable	**Information Value**
TOTAL_REC_PRNCP	4.79205
REVOL_BAL	0.53106
LAST_PYMNT_AMNT	0.52132
AVG_CUR_BAL	0.51822
BC_OPEN_TO_BUY	0.50532
TOTAL_BAL_IL	0.43215
INT_RATE	0.41648
TOTAL_IL_HIGH_CREDIT_LIMIT	0.38190
MAX_BAL_BC	0.36061
DTI	0.34549
ANNUAL_INC	0.32935
TOTAL_BAL_EX_MORT	0.30800
TOTAL_REC_INT	0.25405
TOT_HI_CRED_LIM	0.23620
TOTAL_REV_HI_LIM	0.21695
TOTAL_PYMNT	0.19101
TOT_CUR_BAL	0.18971
OUT_PRNCP_INV	0.18852
OUT_PRNCP	0.18615
TOTAL_PYMNT_INV	0.18010

Table 5.3: Information Value sample

5.4 Variable Clustering

The $k-$means clustering technique groups variables into clusters. One variable per cluster is then chosen to be used in the model. The selection of a variable in each cluster was based on the variable with the lowest correlation with other variables in the cluster and highest R^2. It is then tested for independence with the other chosen variables from other clusters and the outcome variable. This is the variable which best represents the cluster. Table 5.4 below shows the results of the k-means clustering technique produced by this study. It shows the

number of clusters and the variables in each cluster, as well as the variance explained by each cluster. When developing the scorecards, at least one variable from each cluster will be selected. Table A.3 in the appendix shows the full details of all of the variables in each cluster.

Cluster	**Number of Variables**
1	5
2	5
3	3
4	4
5	4
6	5
7	3
8	6
9	3
10	2
11	2
12	3
13	3
14	3
15	3
16	2
17	2
18	1
19	2
20	1

Table 5.4: Variable Cluster Summary

Table 5.4 indicates that there are 20 clusters of variables. Each cluster explains a part of the outcome variable and hence it is enough to use at least one variable per cluster. A Pearson correlation test between variables in the same cluster will show a high linear association as the variables explain the same part of the data.

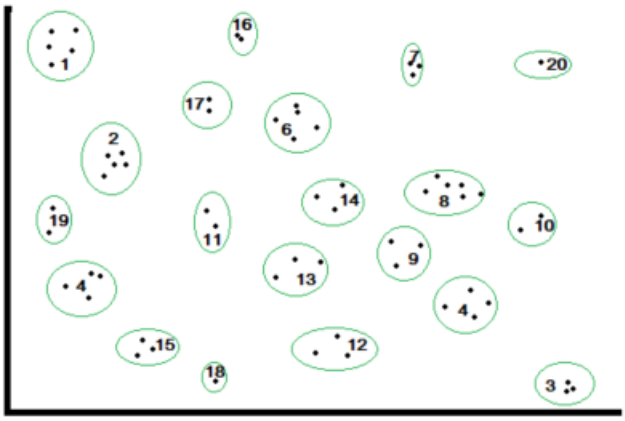


Figure 5.2: Data Sampling

Figure 5.2 shows a rough approximation of the 20 clusters. Variables that are closer together in the variable space are seen to be similar and explain similar parts of the outcome variable. Cluster number eight has six variables and can be reasonably grouped into two clusters and this is the case with other clusters. This is a consequence of the algorithm being sensitive to initial parameters. Figure 5.3 gives an example of how the grouping could have taken place.

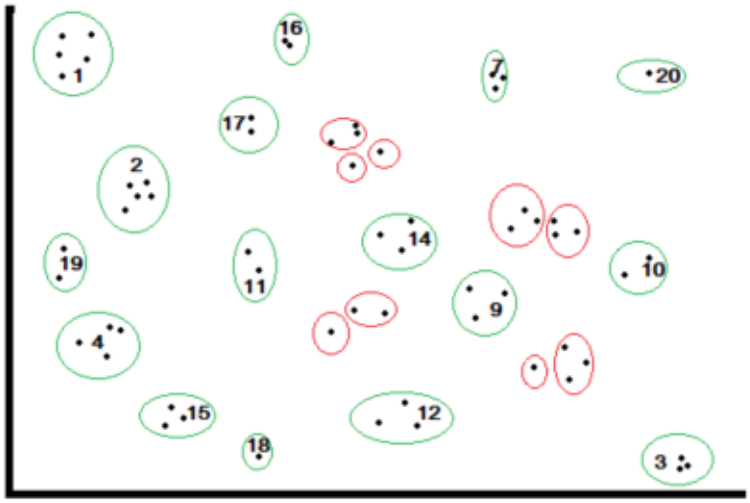


Figure 5.3: Data Sampling

Table 5.5 shows the R-Squared of the chosen variables per cluster. The full description of these variables is given in the Appendix.

Cluster	Variable	Variable Full Name	R-Squared
1	TOT_CUR_BAL	Total current balance	0.87699
2	MO_REV_TL_OP	Age of recent revolving account	0.69729
3	OPEN_IL_12M	Opened instalment accounts in the last 12 months	0.88294
4	TOTAL_PYMNT	Total payment	0.96249
5	INT_RATE	Interest rate	0.95111
6	BC_UTIL	Bankcard utilisation	0.86390
7	FUNDED_AMNT	Funded amount	0.98375
8	TOTAL_REV_HI_LIM	Total revolving high credit limit	0.86929
9	OUT_PRNCP	Outstanding Principal	0.95589
10	NUM_ACTV_REV_TL	Number of Active revolving accounts	0.99254
11	TOTAL_BAL_EX_MORT	Total balance excluding mortgage	0.84196
12	INQ_LAST_6MTHS	Inquiries in past six months	0.78598
13	TOTAL_ACC	Total accounts	0.56590
14	OPEN_RV_12M	Revolving accounts opened in the last 12 months	0.84163
15	IL_UTIL	Instalment account utilisation	0.66251
16	INQ_FI	Personal loan enquires	0.75133
17	ANNUAL_INC	Annual income	0.62231
18	TOTAL_REC_LATE_FEE	Total recovered late fees	1.00000
19	DTI	Debt to Income	0.54106
20	MTHS_SINCE_LAST_RECORD	Months since last public record	1.00000

Table 5.5: Selected Variables

5.5 Variable Independence

The correlation coefficient and chi-square test of independence test for the similarity between the variables were performed on the data, and a sample of the results is shown in Table 5.6 and 5.7, respectively. As outlined in the earlier chapter, the correlation coefficient looks at how one variable relates to other variables with regards to predictive ability and the Chi-Square test of independence looks at the impact of introducing a new variable to a group of variables.

Variable	annual_inc	bc_util	dti	funded_amnt	il_util	inq_fi	inq_last_6mths	int_rate	mo_sin_rcnt_rev_tl_op	total_rec_late_fee	mths_since_last_record	num_actv_rev_tl	open_il_12m	open_rv_12m	out_prncp	tot_cur_bal	total_acc	total_bal_ex_mort	total_pymnt	total_rev_hi_lim
annual_inc	1.00	0.00	-0.16	0.33	-0.05	0.07	0.03	-0.13	0.04	-0.03	-0.08	0.08	0.08	-0.01	0.07	0.28	0.18	0.33	0.31	0.32
bc_util	0.00	1.00	0.16	0.03	-0.04	-0.05	-0.07	0.25	0.13	0.03	0.10	0.09	-0.07	-0.19	0.01	0.03	-0.09	0.08	0.04	-0.18
dti	-0.16	0.16	1.00	-0.01	0.01	0.08	0.01	0.16	-0.03	0.02	0.07	0.23	0.17	0.00	0.01	0.00	0.19	0.25	-0.02	0.07
funded_amnt	0.33	0.03	-0.01	1.00	-0.12	0.01	-0.02	-0.07	0.05	-0.02	-0.03	0.16	-0.01	-0.04	0.18	0.31	0.20	0.28	0.29	0.25
il_util	-0.05	-0.04	0.01	-0.12	1.00	0.14	0.08	0.18	-0.06	0.01	-0.01	-0.07	0.32	0.06	-0.08	0.02	0.11	0.23	-0.13	-0.11
inq_fi	0.07	-0.05	0.08	0.01	0.14	1.00	0.22	0.16	-0.07	0.01	-0.05	0.01	0.25	0.10	-0.03	0.10	0.15	0.11	-0.01	-0.03
inq_last_6mths	0.03	-0.07	0.01	-0.02	0.08	0.22	1.00	0.25	-0.20	-0.02	-0.06	0.13	0.15	0.35	-0.04	0.03	0.16	0.03	-0.03	0.01
int_rate	-0.13	0.25	0.16	-0.07	0.18	0.16	0.25	1.00	-0.13	-0.02	0.00	0.05	0.21	0.18	0.00	-0.16	-0.12	-0.07	-0.10	-0.24
mo_sin_rcnt_rev_tl_op	0.04	0.13	-0.03	0.05	-0.06	-0.07	-0.20	-0.13	1.00	-0.08	0.05	-0.23	-0.06	-0.47	0.03	0.04	-0.16	0.02	0.06	-0.04
total_rec_late_fee	-0.03	0.03	0.02	-0.02	0.01	0.01	-0.02	-0.02	-0.08	1.00	0.08	0.04	0.03	0.03	0.01	-0.06	-0.05	-0.04	-0.02	0.00
mths_since_last_record	-0.08	0.10	0.07	-0.03	-0.01	-0.05	-0.06	0.00	0.05	0.08	1.00	0.04	0.00	-0.09	-0.03	-0.03	-0.16	-0.04	-0.02	-0.03
num_actv_rev_tl	0.08	0.09	0.23	0.16	-0.07	0.01	0.13	0.05	-0.23	0.04	0.04	1.00	-0.01	0.37	0.03	0.10	0.29	0.14	0.14	0.34
open_il_12m	0.08	-0.07	0.17	-0.01	0.32	0.25	0.15	0.21	-0.06	0.03	0.00	-0.01	1.00	0.07	-0.06	0.10	0.25	0.23	-0.03	-0.01
open_rv_12m	-0.01	-0.19	0.00	-0.04	0.06	0.10	0.35	0.18	-0.47	0.03	-0.09	0.37	0.07	1.00	-0.05	-0.03	0.25	-0.01	-0.05	0.09
out_prncp	0.07	0.01	0.01	0.18	-0.08	-0.03	-0.04	0.00	0.03	0.01	-0.03	0.03	-0.06	-0.05	1.00	0.05	0.00	0.06	0.13	0.07
tot_cur_bal	0.28	0.03	0.00	0.31	0.02	0.10	0.03	-0.16	0.04	-0.06	-0.03	0.10	0.10	-0.03	0.05	1.00	0.32	0.11	0.30	0.12
total_acc	0.18	-0.09	0.19	0.20	0.11	0.15	0.16	-0.12	-0.16	-0.05	-0.16	0.29	0.25	0.25	0.00	0.32	1.00	0.19	0.19	0.31
total_bal_ex_mort	0.33	0.08	0.25	0.28	0.23	0.11	0.03	-0.07	0.02	-0.04	-0.04	0.14	0.23	-0.01	0.06	0.11	0.19	1.00	0.26	0.29
total_pymnt	0.31	0.04	-0.02	0.29	-0.13	-0.01	-0.03	-0.10	0.06	-0.02	-0.02	0.14	-0.03	-0.05	0.13	0.30	0.19	0.26	1.00	0.10
total_rev_hi_lim	0.32	-0.18	0.07	0.25	-0.11	-0.03	0.01	-0.24	-0.04	0.00	-0.03	0.34	-0.01	0.09	0.07	0.12	0.31	0.29	0.11	1.00

Table 5.6: Correlation between variables

Table 5.6 shows there is linear association between the selected variables. The highest linear association in the above table is between OPEN_RV_12M and NUM_ACTV_REV_TL, with a correlation coefficient of 0.37.

Variable	Corr Outcome Variable	IV	Chi-Square	P-value
annual_inc	-0.05408	0.32934	0.00108	>0.99999
bc_util	0.05498	0.09989	0.00274	>0.99999
dti	0.08699	0.34549	1.00033	>0.99999
funded_amnt	-0.0092	0.14099	0.00145	>0.99999
il_util	0.07971	0.02970	0.00106	>0.99999
inq_fi	0.07576	0.01581	0.00065	>0.99999
int_rate	0.23400	0.4165	0.00047	>0.99999
mo_sin_rcnt_rev_tl_op	-0.0675	0.05041	0.00045	>0.99999
mths_since_last_record	0.00807	0.01697	0.00139	>0.99999
num_actv_rev_tl	0.05093	0.02007	0.00101	>0.99999
inq_last_6mths	0.08588	0.03920	0.00032	>0.99999
open_il_12m	0.09529	0.01821	0.00079	>0.99999
open_rv_12m	0.09139	0.02127	0.00075	>0.99999
out_prncp	-0.08217	0.18615	0.00044	>0.99999
tot_cur_bal	-0.08093	0.18971	0.00016	>0.99999
total_acc	-0.01616	0.01131	0.00010	>0.99999
total_bal_ex_mort	-0.02720	0.30890	0.00037	>0.99999
total_pymnt	-0.29248	0.19101	0.00031	>0.99999
total_rev_hi_lim	-0.07751	0.21695	0.00039	>0.99999
total_rec_late_fee	-0.03561	0.25310	0.00052	>0.99999

Table 5.7: Correlation with the outcome variable, IV and Chi-Squared value

The tolerance for the correlation coefficient is usually left to interpretation. Table 3.2 shows the scales with the strength of association. All the variables shown in Table 5.7 do not show evidence of linear association with the outcome variable, as they all have absolute correlation coefficients with the outcome variable of below 0.3, which shows a weak association as indicated in Table 3.2. The *p-value* for each of the variables is greater than 0.99999, thus the null hypothesis cannot be rejected.

5.6 Variable Binning

The WOE metric is utilised in this study to bin the variables. As outlined in the earlier chapters, quantitative variables will be divided into intervals and each interval will be assigned a score in the modelling stage. For categorical variables, categories with the same WOE can be grouped together. This grouping may result in the volumes in each bin being unequal. However, it is common practice that each group has at least 5% volume. Bins with less than 5% volume will be grouped together with the closest group. Business rules and

assumptions also need to be taken into account when binning the variables. The grouping may alter the predictive power of the variable and, in the end, may eliminate some variables from the scorecard development entirely.

The variables were binned based on a random sample of 100 000 observations, as this seen as sufficient for this study. The following section shows the bins in the selected variables and visually assesses the relationship between the selected variable and the outcome variable. The tables in the following section are calculated using the formulae outlined in sub-section 3.2.2.1 (WOE and IV example).

5.6.1 Total current balance (TOT_CUR_BAL)

This variable shows the total current balance of all the customer accounts. For customers with more than one account, the current balance in these individual accounts was aggregated. Table 5.8 shows the binning of the variable TOT_CUR_BAL. The bin with the least volume is [100k, 140k) with a volume of 7.39%. This does not violate the practice of having at least 5% in each bin.

Bin	**Count**	**Bin %**	**# Good**	**# Bad**	**Bad Rate**	**Good %**	**Bad %**	**WOE**	**Bin IV**
[Low, 100k)	58 509	58.51%	45 228	13 281	22.70%	56.52%	66.49%	-0.16248	0.01620
[100k, 140k)	7 391	7.39%	5 963	1 428	19.32%	7.45%	7.15%	0.04144	0.00013
[140k, 220k)	13 569	13.57%	11 204	2 365	17.43%	14.00%	11.84%	0.16764	0.00362
[220k, High)	20 531	20.53%	17 630	2 901	14.13%	22.03%	14.52%	0.41669	0.03128
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		**0.05123**

Table 5.8: TOT_CUR_BAL

It can been seen from Figure 5.4 below that the lower the total current balance, the higher the bad rate. Additionally, most of the customers in the data have a balance of less than R100 000. Customers with relatively higher total current balances have a lower bad rate and this is consistent with what was anticipated.

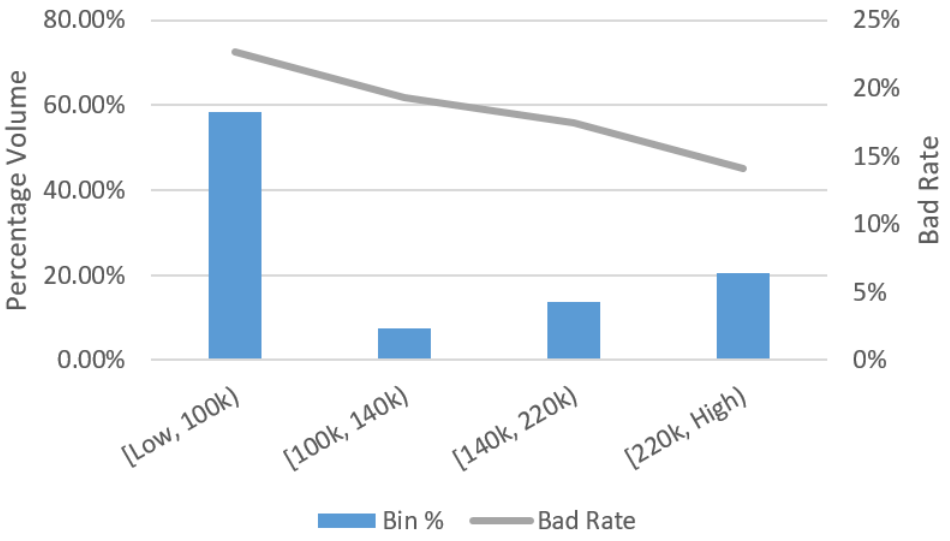


Figure 5.4: TOT_CUR_BAL

5.6.2 Age of recent revolving account opened (MO_REV_TL_OP)

The number of credit applications shows the behaviour of customers. This variable measures the number of months since the most recent revolving account was opened by the customer. It can be anticipated that customers with very recent credit applications are likely to be bad customers. This may be due to the strain on income from the new credit taken on, and if the credit applications were not approved, this may indicate how poor the customer is.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 4.0)	24 501	24.50%	18 754	5 747	23.46%	23.44%	28.77%	-0.20513	0.01095
[4.0, 10.0)	32 490	32.49%	25 612	6 878	21.17%	32.00%	34.43%	-0.07313	0.00178
[10.0, 17.0)	19 157	19.16%	15 457	3 700	19.31%	19.32%	18.52%	0.04187	0.00033
[17.0, 29.0)	13 041	13.04%	10 834	2 207	16.92%	13.54%	11.05%	0.20320	0.00506
[29.0, High)	10 811	10.81%	9 368	1 443	13.35%	11.71%	7.22%	0.48272	0.02164
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.03975

Table 5.9: MO_REV_TL_OP

It can be seen in the below image that customers with very recent revolving accounts are likely to have a relatively higher bad rate at 23%, compared to those with revolving accounts that are 29 months old or more. More than 50% of the population has revolving accounts that are 10 months or younger. This can be a sign of a young population, those who are still starting out in their credit journeys, and hence more opportunities for cross-selling from the bank.

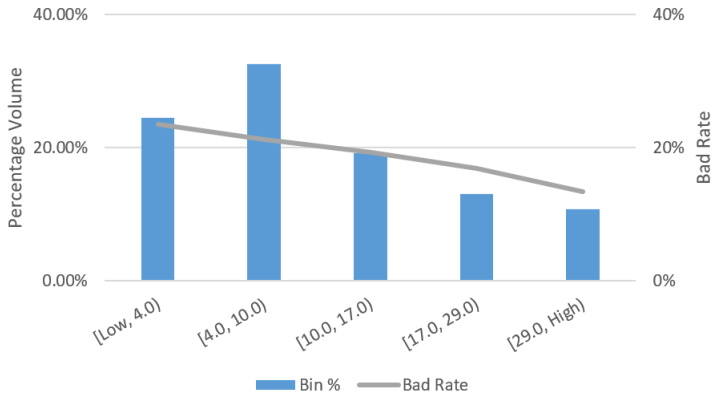


Figure 5.5: MO_REV_TL_OP

5.6.3 Opened instalment accounts in the last 12 months (OPEN_IL_12M)

The number of instalment accounts that a customer has can impact their ability to maintain any new credit. This ability is further hindered if the customer has recently taken on instalment accounts. This variable records the number of instalment accounts that the customer opened in the last 12 months.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
missing	61 015	61.02%	49 143	11 872	19.46%	61.41%	59.43%	0.03269	0.00065
[Low, 1.0)	20 737	20.74%	17 062	3 675	17.72%	21.32%	18.40%	0.14744	0.00431
[1.0, 2.0)	11 682	11.68%	9 125	2 557	21.89%	11.40%	12.80%	-0.11567	0.00162
[2.0, High)	6 566	6.57%	4 695	1 871	28.50%	5.87%	9.37%	-0.46783	0.01637
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.02295

Table 5.10: OPEN_IL_12M

It can be seen in the figure below that more than 60% of the customers in the population did not open any instalment accounts in the last 12 months. These customers appear in the bin of "missing". Customers who have had more than two open instalment accounts in the previous 12 months have a higher relative failure rate, which is consistent with expectations.

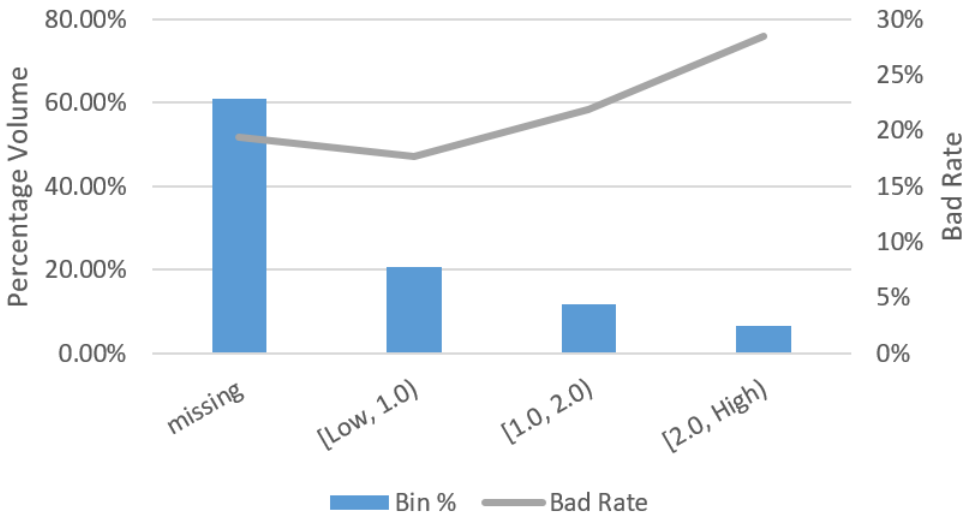


Figure 5.6: OPEN_IL_12M

5.6.4 Total payment (TOTAL_PYMNT)

Assessing whether the customer can afford any more credit products involves taking into account the customers' behaviour with the current credit the customer has. This variable measures the amount that the customer has paid to date. The anticipation for this variable is that customers who have paid more toward their funded amount are likely to have a low bad rate.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 5k)	17 065	17.07%	8 576	8 489	49.75%	10.72%	42.50%	-1.37766	0.43784
[5k, 10k)	29 090	29.09%	22 630	6 460	22.21%	28.28%	32.34%	-0.13421	0.00545
[10k, 21k)	35 605	35.61%	31 510	4 095	11.50%	39.38%	20.50%	0.65268	0.12319
[21k, High)	18 240	18.24%	17 309	931	5.10%	21.63%	4.66%	1.53487	0.26045
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.82693

Table 5.11: TOTAL_PYMNT

Customers who have made lower total payments towards their funded amount have a higher bad rate, in fact, a bad rate of 50%. There are 17% of such customers and the rest of the customers have paid more than 5 000 towards this funded amount and the bad rate lowers as the total amount paid increases. This variable gives a good separation between potentially good and bad customers.

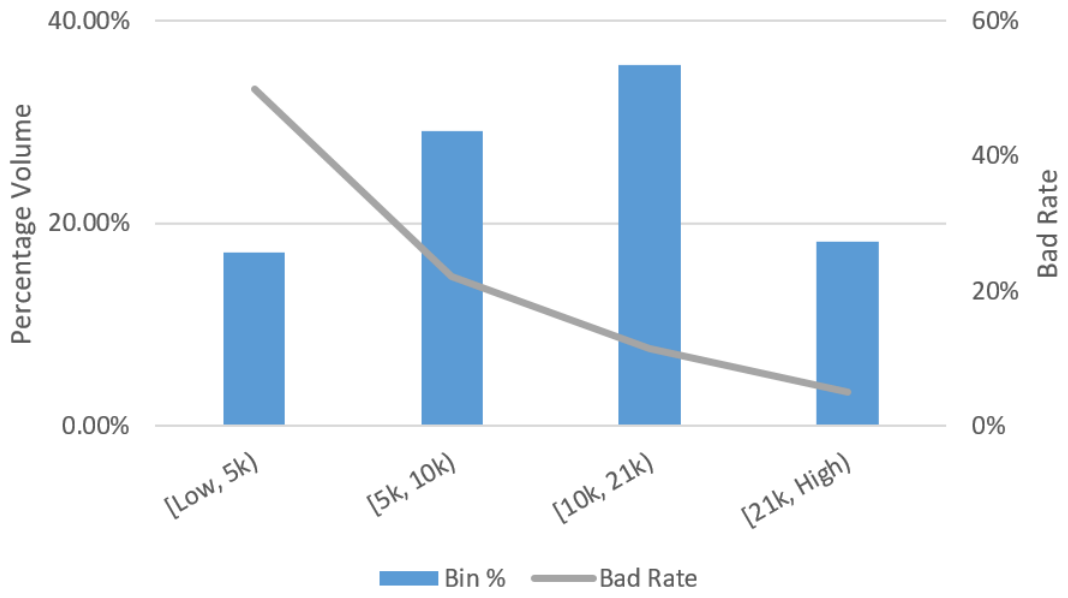


Figure 5.7: TOTAL_PYMNT

5.6.5 Interest Rate (INT_RATE)

This variable shows the interest rate that the customer was quoted at when they took up the loan. The anticipation of this variable is that customers who were noted with a higher interest rate initially are likely to be categorised as bad. This is because the bank is unsure about the customer's ability to pay back and hence they charge a higher interest.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 8.0)	18 616	18.62%	17 269	1 347	7.24%	21.58%	6.74%	1.16318	0.17257
[8.0, 11.0)	23 804	23.80%	20 467	3 337	14.02%	25.58%	16.71%	0.42588	0.03778
[11.0, 13.0)	20 626	20.63%	16 528	4 098	19.87%	20.65%	20.52%	0.00670	0.00002
[13.0, 15.0)	15 799	15.80%	11 842	3 957	25.05%	14.80%	19.81%	-0.29169	0.01462
[15.0, High)	21 155	21.16%	13 919	7 236	34.02%	17.39%	36.23%	-0.73367	0.13817
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.36314

Table 5.12: INT_RATE

The population is approximately equally distributed amongst the different interest rate bins. The bad rate is higher in the higher interest rate bins. This shows that the anticipation that customers who were noted to have a higher interest rate initially are likely to be categorised as bad customers, and that the AO scorecard used, correctly ranks the customers. It may also be as a result of a higher interest rate implying higher instalment amounts, which makes it difficult for applicants with higher quoted interest rate to maintain their credit facilities.

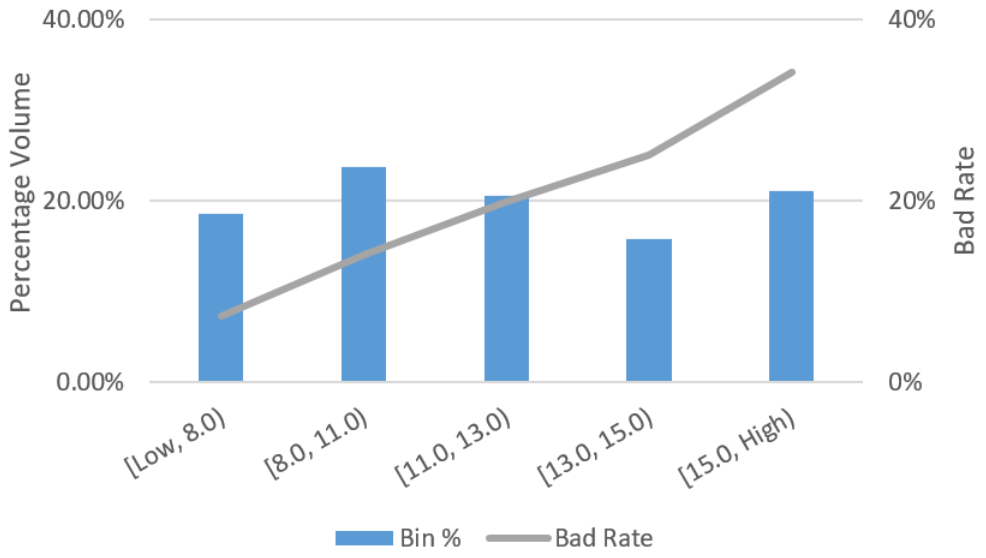


Figure 5.8: INT_RATE

5.6.6 Bankcard Utilisation (BC_UTIL)

This variable shows the ratio of total current balance to credit limit for all accounts, i.e., how much of the available credit does the customer utilise. A customer who uses all their available credit is likely to have a higher bad rate, since it means that they are in need of funds and using credit to fund this need. Table 5.13 shows the binning of the variable BC_UTIL, the minimum volume (Bin %) in all bins is 14.79%, and the bad rate increases with the bankcard utilisation.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 40.0)	25 581	24.44%	21 211	4 370	16.84%	26.51%	21.88%	0.20896	0.01022
[40.0, 55.0)	14 789	14.79%	12 014	2 775	18.76%	15.01%	13.89%	0.07756	0.00087
[55.0, 90.0)	41 006	41.01%	32 483	8 523	20.78%	40.59%	42.67%	-0.04991	0.00103
[90.0, High)	18 624	18.62%	14 317	4 307	23.13%	17.89%	21.56%	-0.18665	0.00685
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.01898

Table 5.13: BC_UTIL

It can be seen in the figure below that with higher utilisation, the bad rate is also higher, and hence this is a desirable variable as it can help classify customers. Most of the customers in the population have a utilisation of between 55% and 90%, with a bad rate of 21%. The impact of this kind of distribution depends on the business rules.

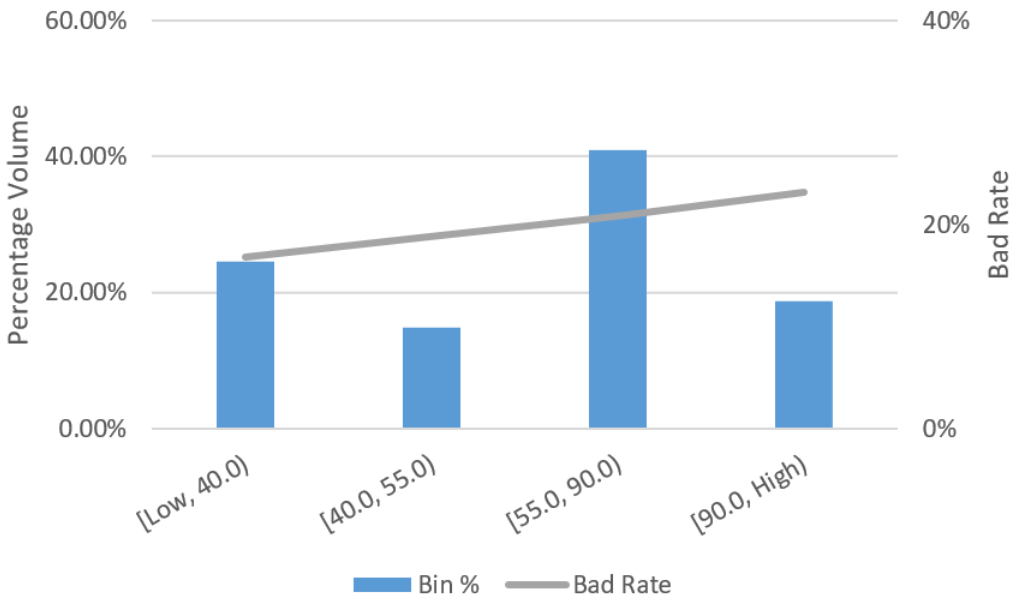


Figure 5.9: BC_UTIL

5.6.7 Funded amount (FUNDED_AMNT)

The amount of credit given to any customer can be an indication of how good the customer is at maintaining their credit. In particular, this variable gives the size of the loan that the customer has with the bank. The anticipation for this variable is that the higher the funded amount, the lower the bad rate. This is likely because the creditors would not intentionally risk high amounts with clients who have higher bad rates.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 4k)	7 804	7.80%	6 331	1 473	18.87%	7.91%	7.37%	0.07030	0.00038
[4k, 10k)	35 456	35.46%	27 939	7 517	21.20%	34.91%	37.63%	-0.07500	0.00204
[10k, 19k)	35 330	35.33%	28 364	6 966	19.72%	35.44%	34.87%	0.01622	0.00009
[19k, 24k)	8 910	8.91%	7 234	1 676	18.81%	9.04%	8.39%	0.07452	0.00048
[24k, 29k)	6 488	6.49%	5 407	1 081	16.66%	6.76%	5.41%	0.22195	0.00296
[29k, High)	6 012	6.01%	4 750	1 262	20.99%	5.94%	6.32%	-0.06241	0.00024
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.00622

Table 5.14: FUNDED_AMNT

It can be seen from Table 5.14 above and Figure 5.10 below, that the anticipation for this variable is partially correct. There is no clear relationship between the bad rate and the amount of the loan given. More than 80% of the loans are for amounts of less than 19k. This is in line with expectations as the data is based on a book of short-term loans.

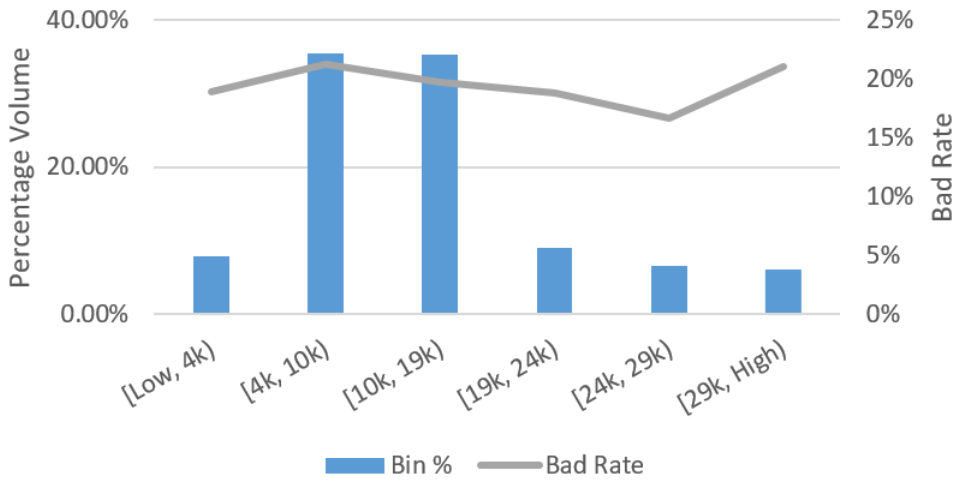


Figure 5.10: FUNDED_AMNT

5.6.8 Total revolving high credit limit (TOT_REV_HI_LIM)

This variable shows the total revolving limit that was granted and is available to the customer. The anticipation for this variable is that customers with higher credit facilities available to them are less likely to default on their loan.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 6k)	5 943	5.94%	4 345	1 598	26.89%	5.43%	8.00%	-0.38759	0.00996
[6k, 20k)	38 697	38.70%	29 909	8 788	22.71%	37.37%	43.99%	-0.16309	0.01080
[20k, 40k)	32 108	32.11%	25 855	6 253	19.47%	32.31%	31.30%	0.03159	0.00032
[40k, 70k)	15 735	15.74%	13 266	2 469	15.69%	16.58%	12.36%	0.29353	0.01238
[70k, High)	7 517	7.52%	6 650	867	11.53%	8.31%	4.34%	0.64948	0.02578
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.05924

Table 5.15: TOT_REV_HI_LIM

The figure below shows a desirable monotonic relationship between the bad rate and the amount of revolving credit that is available to the customer. As anticipated, customers with lesser revolving credit available to them have a higher relative bad rate.

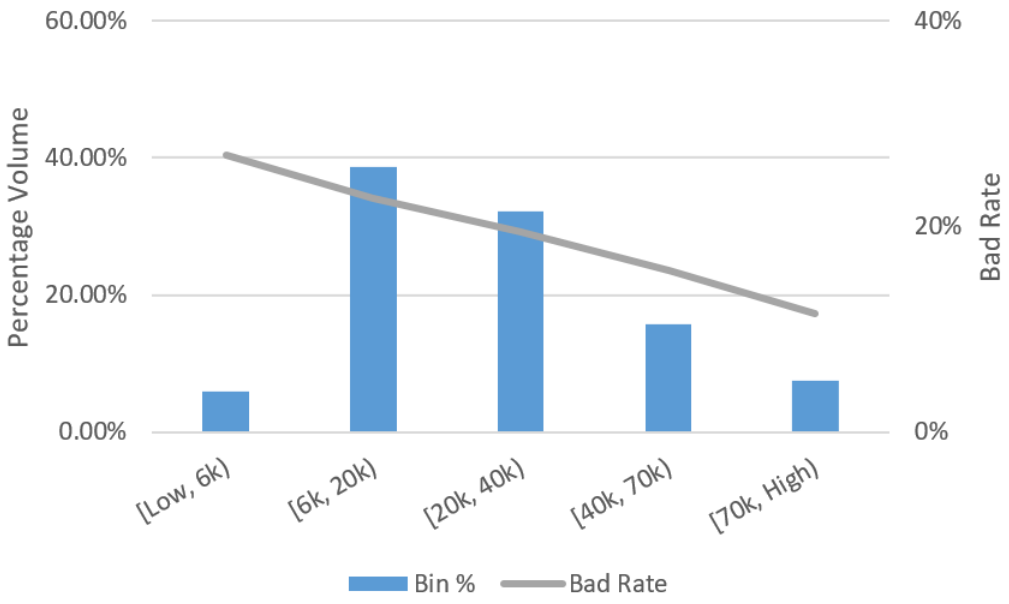


Figure 5.11: TOT_REV_HI_LIM

5.6.9 Outstanding Principal (OUT_PRNCP)

This variable shows the remaining outstanding principal for the total amount funded. The expectation is that customers who have a lower outstanding principal are likely to have a higher bad rate. This is because those customers have probably had the loan for a longer period of time and have possibly missed some payments.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 2000.0)	88 528	88.53%	69 108	19 420	21.94%	86.36%	97.22%	-0.11849	0.01287
[2000.0, 5000.0)	6 380	6.38%	6 162	218	3.42%	7.70%	1.09%	1.95380	0.12912
[5000.0, High)	5 092	5.09%	4 755	337	6.62%	5.94%	1.69%	1.25901	0.05357
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.19556

Table 5.16: OUT_PRNCP

It can be seen that almost 90% of the loans are close to being paid off as they are on a bin of [Low, 2000). This variable helps identify those who can be marketed other products to be able to extend the relationship with the bank. The bad rate is higher for customers who have a lower outstanding principal, in line with presuppositions.

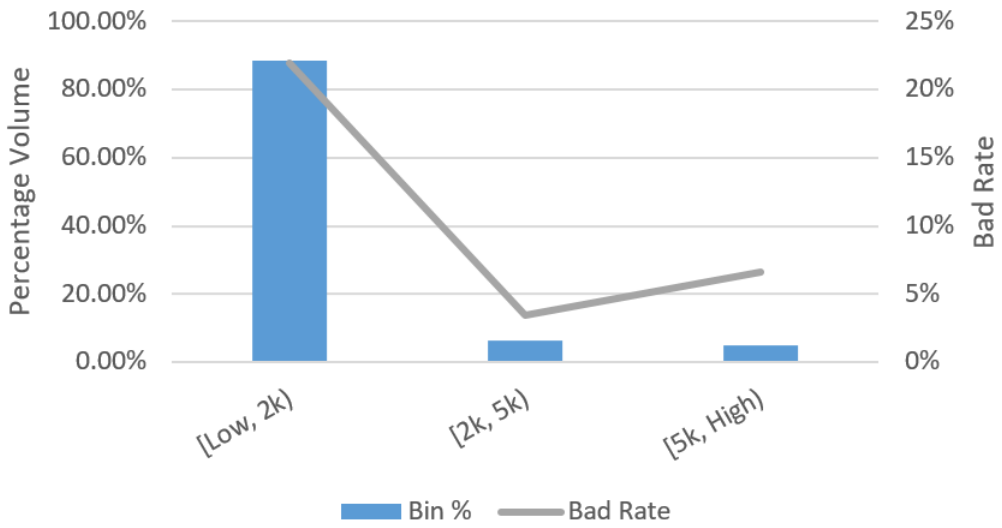


Figure 5.12: OUT_PRNCP

5.6.10 Number of Active revolving accounts (NUM_ACTV_REV_TL)

Customers that have multiple revolving credit accounts are considered higher risk, and any unforeseen event can have a big impact on their credit. This variable counts the number of revolving accounts that the customer has, and the expectation is that the more active revolving accounts the customer has, the more likely to have a higher bad rate. The distribution of the variable NUM_ACTV_REV_TL is shown in Table 5.17, and as can be seen, all of the bins have volumes greater than 5%.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 4.0)	26 624	26.62%	22 008	4 616	17.34%	27.50%	23.11%	0.17402	0.00764
[4.0, 6.0)	29 706	29.71%	23 902	5 804	19.54%	29.87%	29.06%	0.02756	0.00022
[6.0, 8.0)	20 820	20.82%	16 521	4 299	20.65%	20.64%	21.52%	-0.04161	0.00037
[8.0, 10.0)	11 529	11.53%	8 986	2 543	22.06%	11.23%	12.73%	-0.12553	0.00189
[10.0, High)	11 321	11.32%	8 608	2 713	23.96%	10.76%	13.58%	-0.23322	0.00659
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.01671

Table 5.17: NUM_ACTV_REV_TL

More than 50% of the population has fewer than six active revolving accounts, and as anticipated, the bad debt rate is lower for these groups. The bad rate is monotonically increasing as the number of active accounts for that customer has also increased.

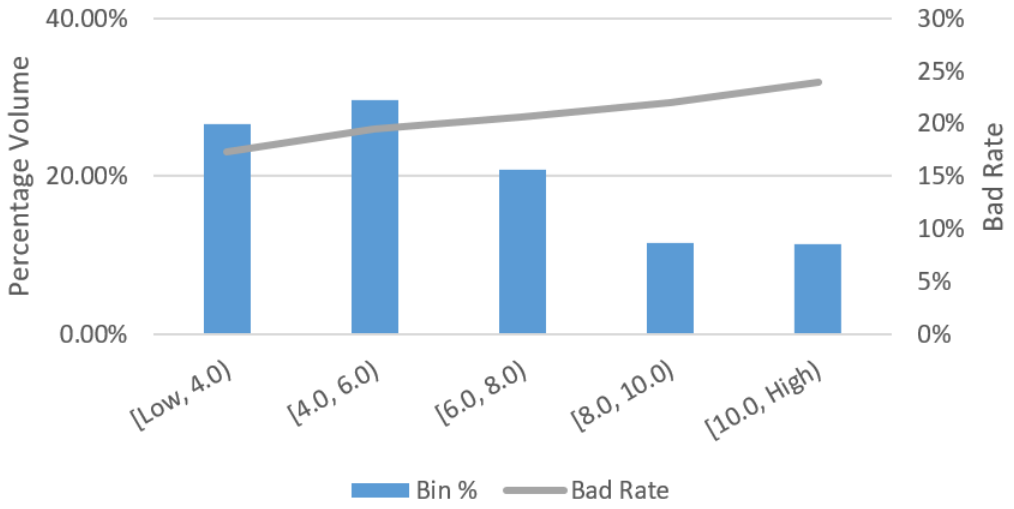


Figure 5.13: NUM_ACTV_REV_TL

5.6.11 Total balance excluding mortgage (TOTAL_BAL_EX_MORT)

This variable gives the total balance owed by the customer, excluding the mortgage. The variable focuses more on unsecured credit, since a mortgage is an example of secure lending. The anticipation for this variable is that customers with higher value for this variable are likely to be lower risk and hence have a lower bad rate. This is because they were approved for the high balance due to good traits that the creditors identified.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 25k)	34 748	34.75%	27 520	7 228	20.80%	34.39%	36.19%	-0.05091	0.00091
[25k, 50k)	32 992	32.99%	26 282	6 710	20.34%	32.84%	33.59%	-0.02257	0.00017
[50k, 105k)	24 034	24.03%	19 385	4 649	19.34%	24.22%	23.27%	0.03999	0.00038
[105k, High)	8 226	8.23%	6 838	1 388	16.87%	8.54%	6.95%	0.20677	0.00330
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.00476

Table 5.18: TOTAL_BAL_EX_MORT

Most of the population have a balance lower than 50 000 and they have a higher bad rate as anticipated. The relationship between the bad rate and the balance is monotonic and decreasing, which also fits the anticipations, based on best business practices.

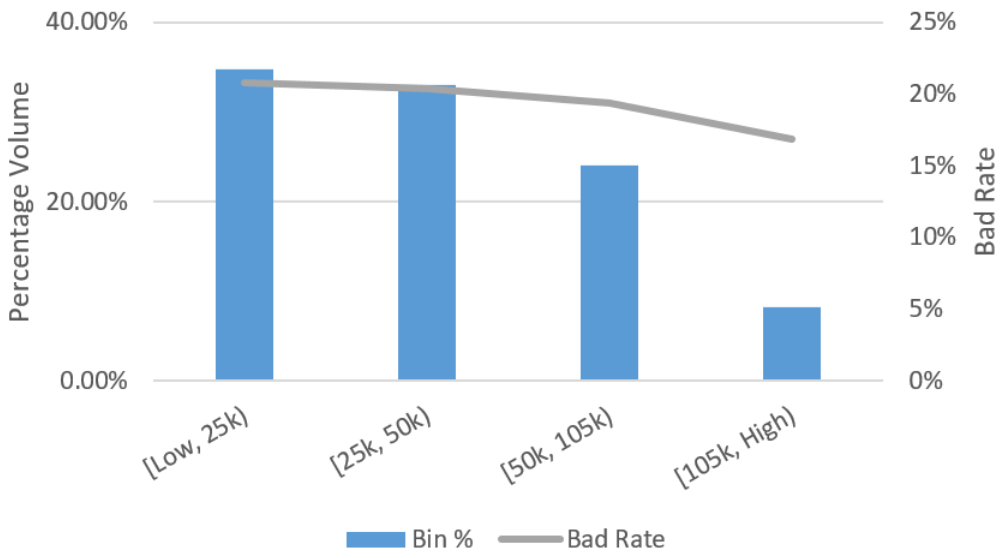


Figure 5.14: TOTAL_BAL_EX_MORT

5.6.12 Inquiries in past six months (INQ_LAST_6MTHS)

As seen in the previous variables, it is important to capture the customer's credit history. This variable gives the number of credit inquiries in the past six months (excluding vehicle and mortgage inquiries). The anticipation for this variable is that the more credit inquiries in the past 6 months that the customer has, the more likely it is to have a higher bad rate.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 1.0)	58 250	58.25%	48 060	10 190	17.49%	60.06%	51.01%	0.16319	0.01476
[1.0, 2.0)	27 148	27.15%	21 302	5 846	21.53%	26.62%	29.27%	-0.09481	0.00251
[2.0, High)	14 602	14.60%	10 663	3 939	26.98%	13.32%	19.72%	-0.39201	0.02507
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.04234

Table 5.19: INQ_LAST_6MTHS

Most of the customers have fewer than two credit inquiries in the past 6 months, and the bad rate is consistent with what was anticipated. The bad rate can be seen in Figure 5.15 can be seen to be monotonically increasing.

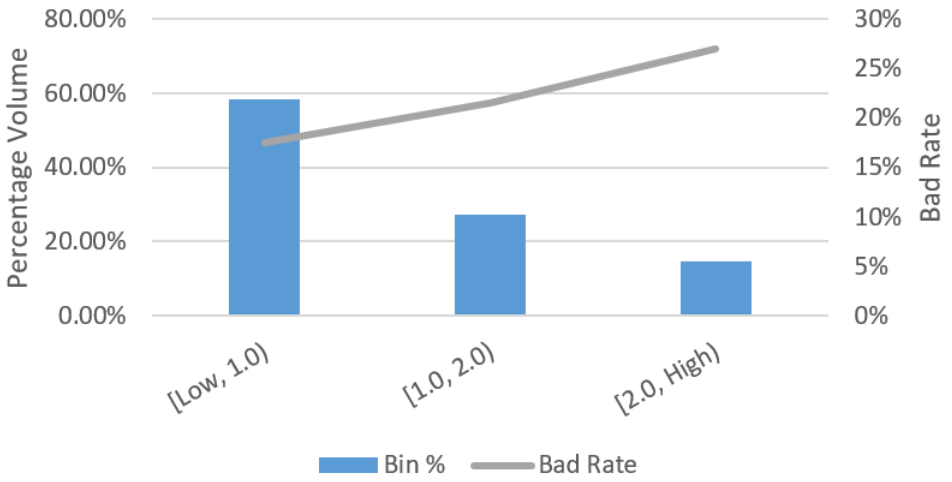


Figure 5.15: INQ_LAST_6MTHS

5.6.13 Total accounts (TOTAL_ACC)

This variable shows the number of credit lines that are in the customer's name. The anticipation for this variable is that customers with many credit lines are likely to have had a good credit score to have allowed them to qualify for the multiple credit lines and hence had a good credit score at the time of application. It can also be interpreted that since they have to pay multiple instalments, they have a higher probability of missing a payment than those who have fewer credit lines.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 2.0)	7 041	7.04%	5 463	1 578	22.41%	6.83%	7.90%	-0.14602	0.00157
[2.0, 6.0)	17 446	17.45%	13 677	3 769	21.60%	17.09%	18.87%	-0.09895	0.00176
[6.0, 10.0)	28 444	28.44%	22 749	5 695	20.02%	28.43%	28.51%	-0.00293	0.00000
[10.0, 16.0)	41 609	41.61%	33 817	7 792	18.73%	42.26%	39.01%	0.08001	0.00260
[16.0, High)	5 460	5.46%	4 319	1 141	20.90%	5.40%	5.71%	-0.05674	0.00018
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.00611

Table 5.20: TOTAL_ACC

There is a slight negative correlation between the number of credit lines and the bad rate. The volume in the highest bin of [16.0 high) has 5.46%, which is just above the required limit.

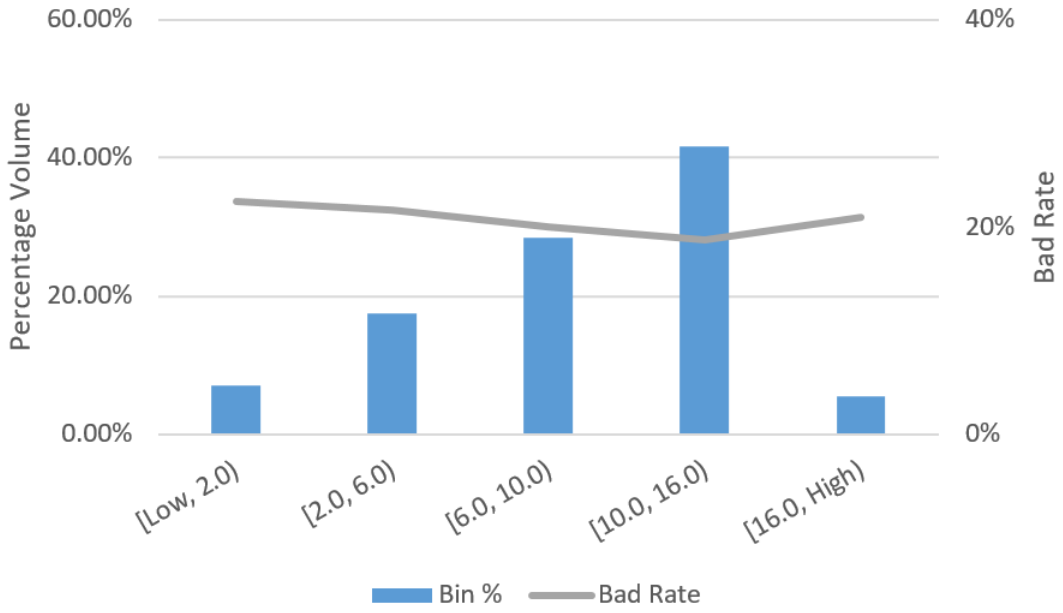


Figure 5.16: TOTAL_ACC

5.6.14 Revolving accounts opened in the last 12 months (OPEN_RV_12M)

This variable shows the number of revolving accounts that the customer has opened in the last 12 months. A revolving account is an account that gives the customer a credit line and allows for changing availability.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
missing	61 015	61.02%	49 143	11 872	19.46%	61.41%	59.43%	0.03269	0.00065
[Low,1.0)	12 899	12.90%	10 717	2 182	16.92%	13.39%	10.92%	0.20373	0.00503
[1.0,2.0)	11 404	11.40%	9 122	2 282	20.01%	11.40%	11.42%	-0.00222	0.00000
[2.0,3.0)	7 245	7.25%	5 565	1 680	23.19%	6.95%	8.41%	-0.19015	0.00277
[3.0,High)	7 437	7.44%	5 478	1 959	26.34%	6.85%	9.81%	-0.35955	0.01065
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.01909

Table 5.21: OPEN_RV_12M

As seen in the table above and the figure below, most of the customers have not opened a revolving account in the last 12 months. The bad rate trend is as anticipated, in that the more revolving accounts are opened by the customers, the higher their bad rate. All revolving accounts need to be paid regularly as they are used. A customer who opens multiple revolving accounts in a short period has a higher chance of falling into difficulties later when having to pay back the instalments.

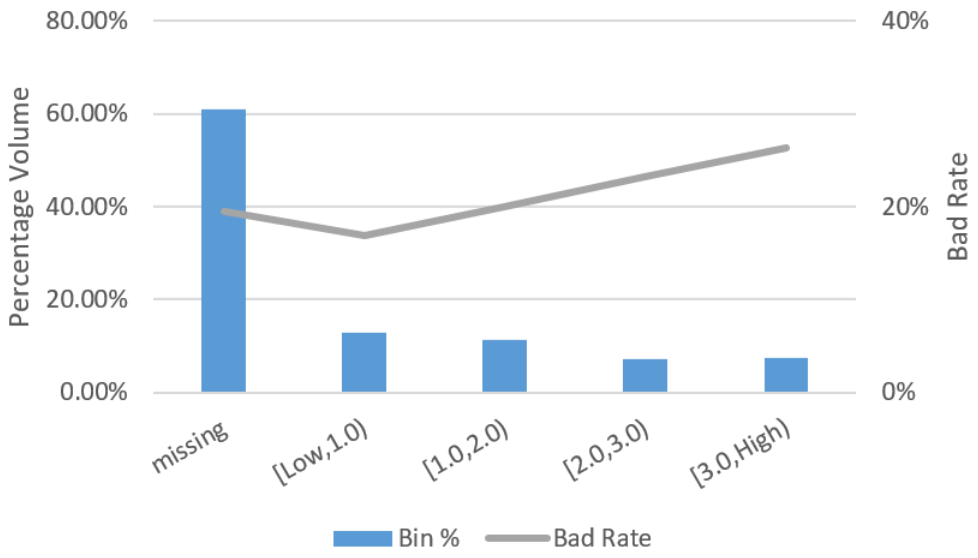


Figure 5.17: OPEN_RV_12M

5.6.15 Instalment account utilisation (IL_UTIL)

The variable shows the ratio of total current balance to high credit/credit limit on all the customer's instalment accounts. These are two different accounts, one is the current account and the other is the credit account. The current account shows what the customers earn and have available to pay credit instalments. The anticipation for this variable is that the higher the ratio, the higher the bad rate, since the ratio being high means that the total current balance is very close to the instalment account balance, and hence there are fewer available funds to service the instalments.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
missing	66 385	66.39%	53 526	12 859	19.37%	66.89%	64.38%	0.03827	0.00096
[Low, 65.0)	11 376	11.38%	9 467	1 909	16.78%	11.83%	9.56%	0.21337	0.00485
[65.0, 80.0)	8 770	8.77%	6 973	1 797	20.49%	8.71%	9.00%	-0.03193	0.00009
[80.0, 90.0)	6 543	6.54%	4 919	1 624	24.82%	6.15%	8.13%	-0.27965	0.00555
[90.0, High)	6 926	6.93%	5 140	1 786	25.79%	6.42%	8.94%	-0.33078	0.00833
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.01978

Table 5.22: IL_UTIL

Most of the customers do not have a value for this variable as they are classified in the "missing" bin. This shows that most of the customers do not disclose their instalment accounts. The bad rate behaves as anticipated for all the other bins.

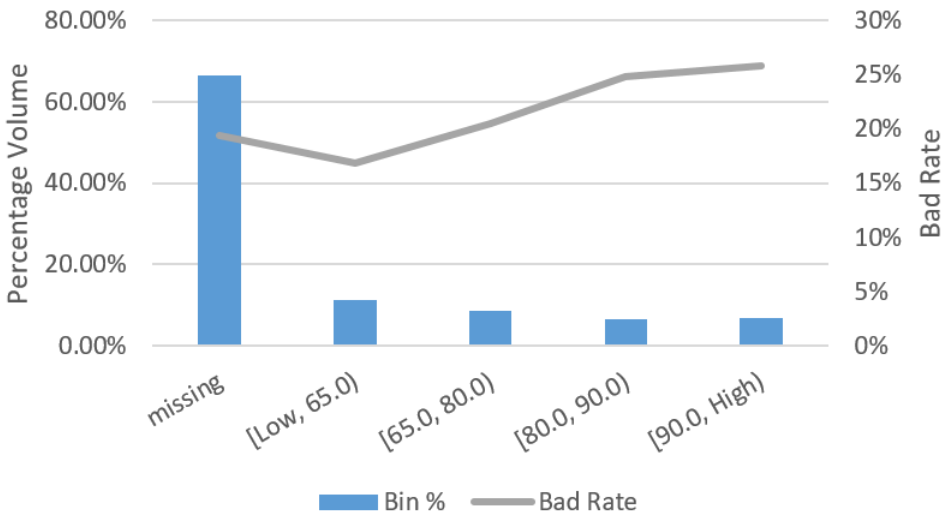


Figure 5.18: IL_UTIL

5.6.16 Personal loan enquires (INQ_FI)

A personal loan is a form of unsecured credit. It is one of the most expensive products amongst the available credit products for customers, resulting in customers applying for these only as a last resort. This variable gives the number of times that the customer has been queried for a personal loan over their credit lifetime. The anticipation of this variable is that, since they are expensive, customers who opt for many of these will likely have a higher bad rate.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
missing	61 015	61.02%	49 143	11 872	19.46%	61.41%	59.43%	0.03269	0.00065
[Low, 1.0)	21 125	21.13%	17 214	3 911	18.51%	21.51%	19.58%	0.09407	0.00182
[1.0, 2.0)	8 961	8.96%	7 056	1 905	21.26%	8.82%	9.54%	-0.07846	0.00057
[2.0, High)	8 899	8.90%	6 612	2 287	25.70%	8.26%	11.45%	-0.32621	0.01040
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.01342

Table 5.23: INQ_FI

It can be seen from Table 5.23 above and Figure 5.19 below that most of the customers have not had any personal loan queries as they are in the "missing" bin. The population decreases as the number of personal loan enquiries increases. The bad rate is consistent with the anticipation.

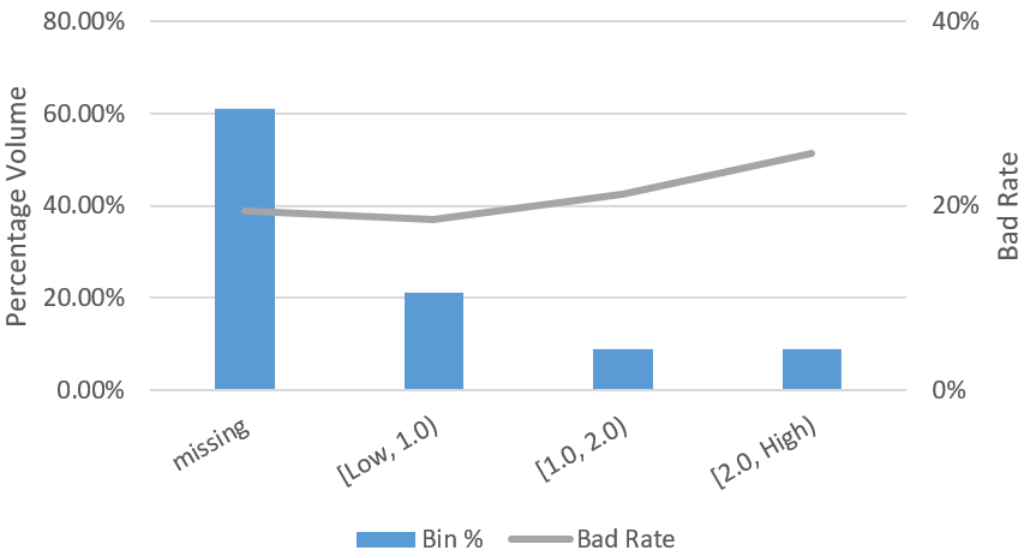


Figure 5.19: INQ_FI

5.6.17 Annual income (ANNUAL_INC)

Annual income is a good indicator of assessing whether the customer can maintain their credit. The anticipation is that customers with a higher annual income will have a lower bad rate. This is because they have relatively more money to maintain their credit commitments.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 300k)	7 106	7.11%	5 114	1 992	28.03%	6.39%	9.97%	-0.44502	0.01594
[300k, 450k)	18 216	18.22%	13 831	4 385	24.07%	17.28%	21.95%	-0.23914	0.01117
[450k, 650k)	26 553	26.55%	21 000	5 553	20.91%	26.24%	27.80%	-0.05767	0.00090
[650k, 900k)	22 950	22.95%	18 699	4 251	18.52%	23.37%	21.28%	0.09346	0.00195
[900k, High)	25 175	25.18%	21 381	3 794	15.07%	26.72%	18.99%	0.34122	0.02636
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.05631

Table 5.24: ANNUAL_INC

Most of the customers fall on the high-income side, with more than 70% earning 450 000 or more. The bad rate is in line with the anticipation of higher income, which corresponds to a lower bad rate.

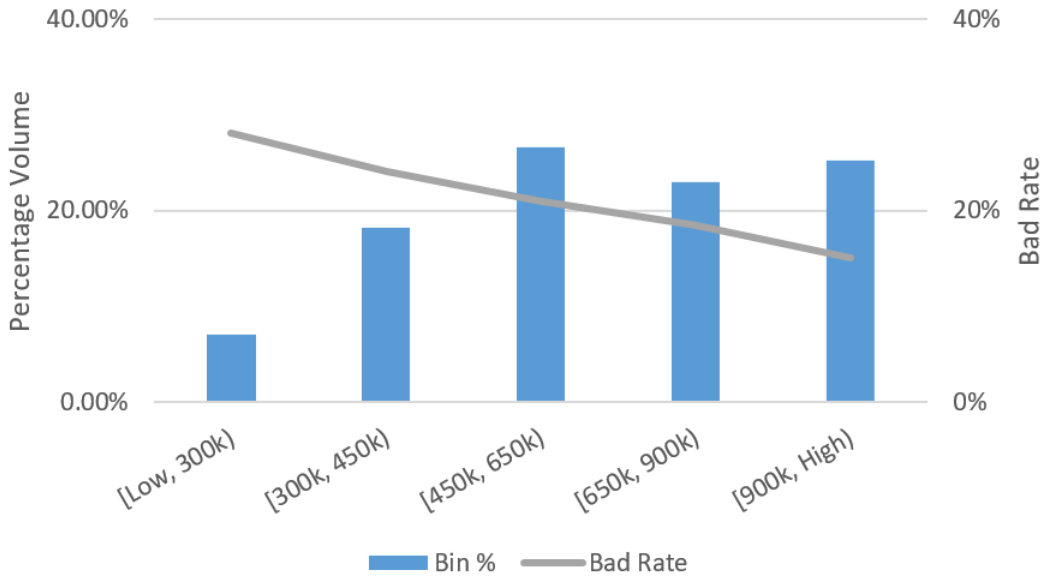


Figure 5.20: ANNUAL_INC

5.6.18 Total recovered late fees (TOTAL_REC_LATE_FEE)

If a customer does not pay an instalment by the agreed timeline with the bank, they are classified as late. There is a late fee charged to the client in addition to their instalment. This variable aggregates the number of late fees that have been recovered every time the customer has been late with a payment. The anticipation is that the higher the amount of late fees recovered from a customer, the lower the bad rate. This is because the more often the customer is able to pay the late fees, this would not contribute to the bad rate, thus lowering it.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 2k)	7 131	7.13%	1 520	5 611	78.68%	1.90%	28.09%	-2.69388	0.70555
[2k, 5k)	16 424	16.42%	9 347	7 077	43.09%	11.68%	35.43%	-1.10965	0.26353
[5k, 15k)	49 215	49.22%	42 829	6 386	12.98%	53.52%	31.97%	0.51525	0.11103
[15k, 24k)	17 700	17.70%	16 909	791	4.47%	21.13%	3.96%	1.67445	0.28750
[24k, High)	9 530	9.53%	9 420	110	1.15%	11.77%	0.55%	3.06225	0.34360
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		1.71121

Table 5.25: TOTAL_REC_LATE_FEE

The bad rate follows the same trend as anticipated. Table 5.25 shows that most customers have at least 5 000 in late fees collected.

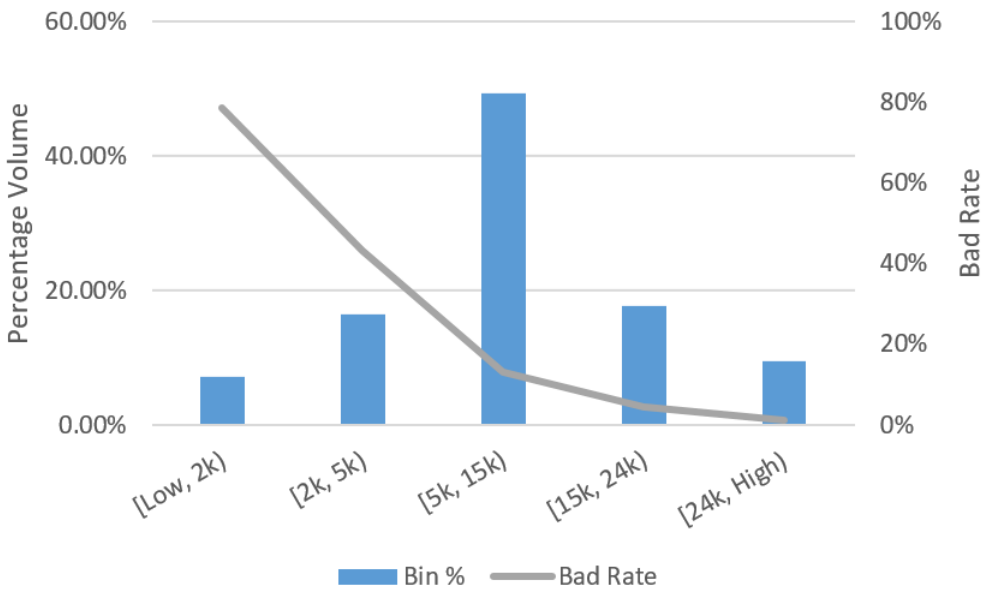


Figure 5.21: TOTAL_REC_LATE_FEE

5.6.19 Debt to Income (DTI)

The variable shows a ratio calculated using the borrowers total monthly debt payments on the total debt obligations, excluding mortgage and the requested loan, divided by the borrowers self-reported monthly income. It is anticipated that with this ratio being higher, the bad rate will be higher.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
[Low, 9.0)	14 741	14.74%	12 521	2 220	15.06%	15.65%	11.11%	0.34204	0.01550
[9.0, 14.0)	19 614	19.61%	16 309	3 305	16.85%	20.38%	16.55%	0.20842	0.00799
[14.0, 21.0)	29 423	29.42%	23 826	5 597	19.02%	29.77%	28.02%	0.06069	0.00106
[21.0, 30.0)	26 703	26.70%	20 532	6 171	23.11%	25.66%	30.89%	-0.18573	0.00973
[30.0, High)	9 516	9.52%	6 835	2 681	28.17%	8.54%	13.42%	-0.45199	0.02206
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.05636

Table 5.26: DTI

The population is fairly distributed across the bins, with [14.0, 21.0) having the highest volume at 29.42%.The bad rate behaves as anticipated and is monotonically increasing.

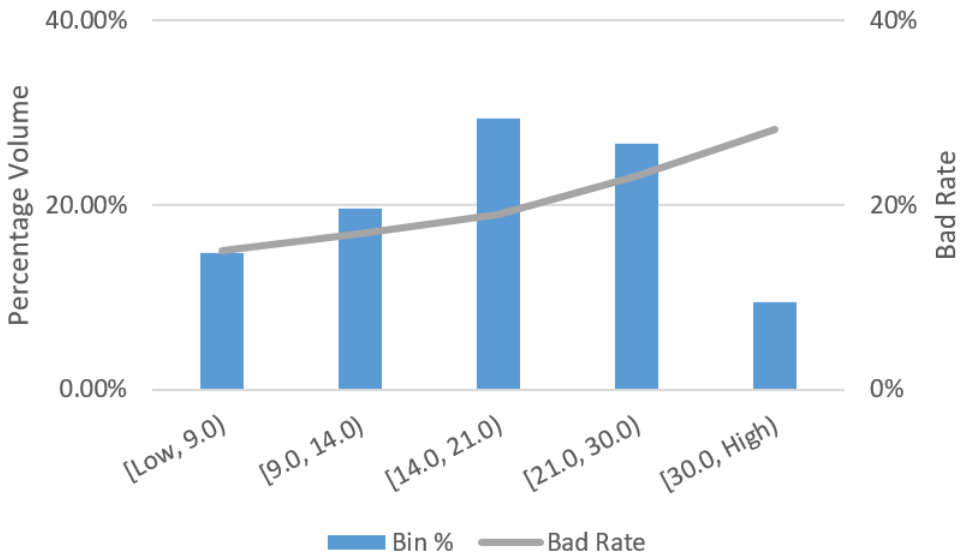


Figure 5.22: DTI

5.6.20 Months since last public record (MTHS_SINCE_LAST_RECORD)

A customer with multiple public records is generally expected to have a higher bad rate. Public records are bad credit records recorded by credit bureaus and the information is supplied to banks. This variable measures the number of months since the customer has had a public record. The expectation is that the more recent the public record, the higher the bad rate.

Bin	Count	Bin %	# Good	# Bad	Bad Rate	Good %	Bad %	WOE	Bin IV
missing	81 588	81.59%	65 917	15 671	19.21%	82.37%	78.45%	0.04873	0.00191
[Low, 12.0)	9 303	9.30%	7 215	2 088	22.44%	9.02%	10.45%	-0.14790	0.00213
[12.0, High)	9 109	9.11%	6 893	2 216	24.33%	8.61%	11.09%	-0.25306	0.00628
Total	100 000	100.00%	80 025	19 975		100.00%	100.00%		0.01031

Table 5.27: MTHS_SINCE_LAST_RECORD

Most of the customers do not have a public record, which is a desirable trait for a population. The bad rate is not consistent with the anticipated idea that the more recent the public record, the lower the bad rate. This might be because those with a recent public record may have been in the goo for longer, and the recent behaviour doesn't reflect the true behaviour.

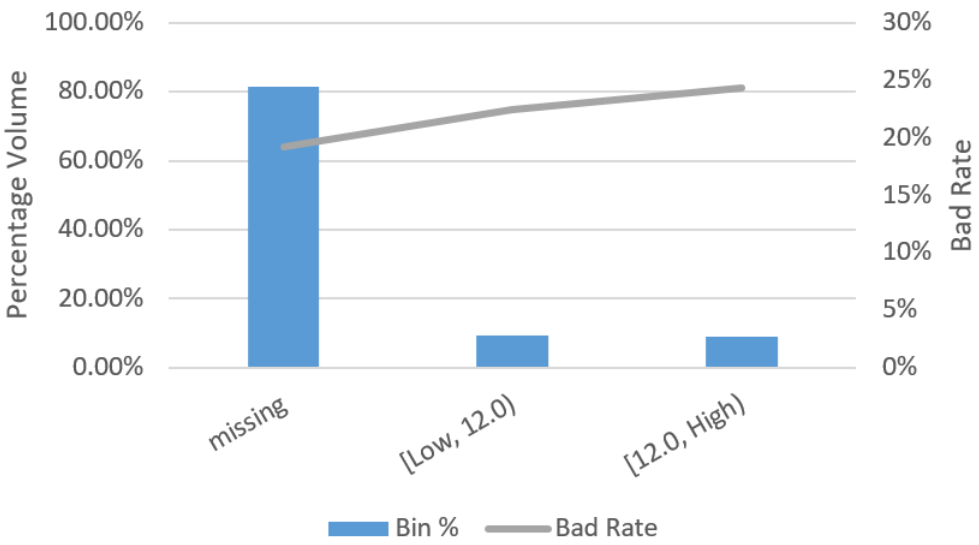


Figure 5.23: MTHS_SINCE_LAST_RECORD

5.7 Summary

The data is ready for modelling since samples and the variables to model (shown in Table 5.5) have been chosen. The number of variables has been reduced to remain with only the statistically significant ones. This implies that a combination of the 20 variables is enough to predict the outcome variable. As seen in the results of the k-means clustering, the population can be fully explained by 20 clusters of variables, which implies that the maximum number of variables for any scorecard to be developed is 20. It is also possible that a scorecard can be sufficiently developed with fewer than 20 variables if the model scores sufficiently on the performance metrics. Furthermore, the scorecards can now be developed using the different techniques and samples.

The full list of variables is given in the appendix. No further details are given about those variables as they are not used in developing any of the scorecards.

Chapter 6

Scorecard Results

6.1 Introduction

This chapter shows the resultant scorecards from the modelling processes. For each of the three samples, ROS, CBOS and SMOTE, scorecards were developed using the modelling techniques discussed in the earlier chapters, which are: Logistic Regression (LR), Probit Regression (PR), Poisson Regression (POI), Linear Discriminant Analysis (LDA), Artificial Neural Networks (ANN), Support Vector Machines (SVM), Random Forest Regression (RFR) and Survival Modelling (SV). The scorecard evaluation metrics (Gini, ROC curve, KS statistic and Confusion matrices) are calculated and displayed for each scorecard developed and used to select the scorecard with the best relative performance using the Built sample.

6.2 ROS BVT

The sample generated using ROS had 250 000 observations, of which 50 000 were 'bad', i.e they had an actual outcome of 1. This gives a bad rate of 20%, which is slightly higher in the sample compared to the BVT sample, as the BVT sample had a bad rate of 14.4% while the population had a bad rate of 10.6%. The ROS BVT sample was further divided into the Built, Validation and Test samples as seen in Table 6.1.

	Built	Validation	Test	ROS BVT
# Good	140 000	41 548	18 452	200 000
# Bad	35 000	8 452	6 548	50 000
Proportion of ROS BVT	70%	20%	10%	100%
Bad Rate	20.00%	16.90%	26.19%	20.00%
Total	**175 000**	**50 000**	**25 000**	**250 000**

Table 6.1: ROS BVT

6.2.1 GINI

Figure 6.1 below shows the Gini calculated from the scoring results using the ROS BVT sample. The RFR scorecard, developed with the ROS sample, outperforms all the scorecards, with a Gini coefficient of 0.73454. The poorest performing scorecard is the one developed using the SVM technique, with a Gini of 0.61698.

Figure 6.1: Gini ROS Scorecards

6.2.2 Kolmogorov-Smirnov (KS)

Figure 6.2 below shows the KS statistic calculated from the scoring results using the ROS BVT sample. The ANN scorecard gives the best KS statistic amongst the scorecards developed using the ROS data with the value of 0.36061, while the SVM scorecard has the lowest KS at 0.31715.

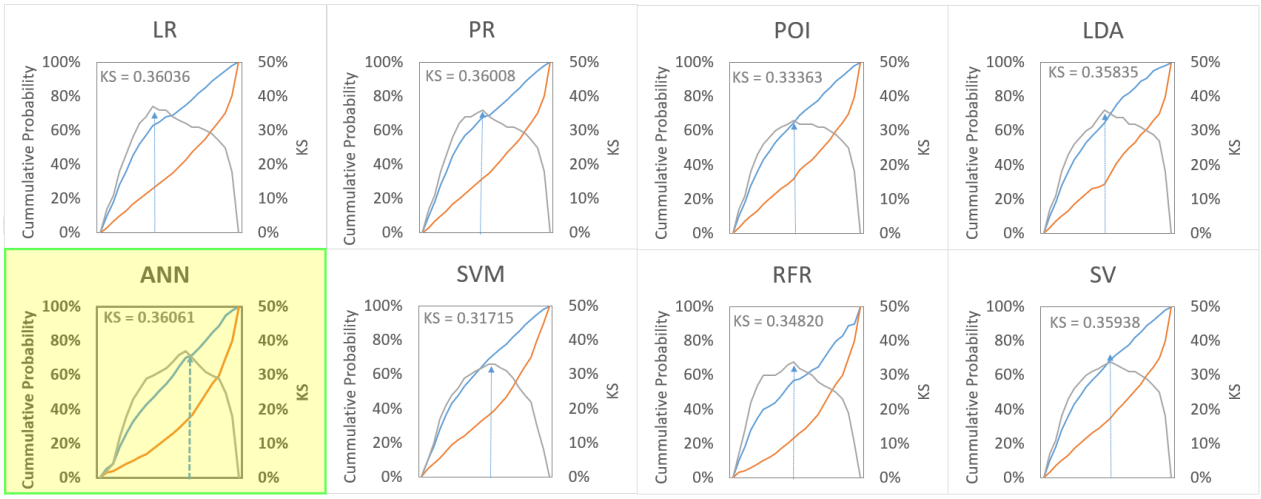


Figure 6.2: KS ROS Scorecards

6.2.3 ROC Curve

Figure 6.3 below shows the ROC constructed from the scoring results using the ROS BVT sample and shows the value of the AUC. The RFR scorecard shows the highest AUC of all scorecards built using the ROS BVT data and SVM has the lowest AUC.

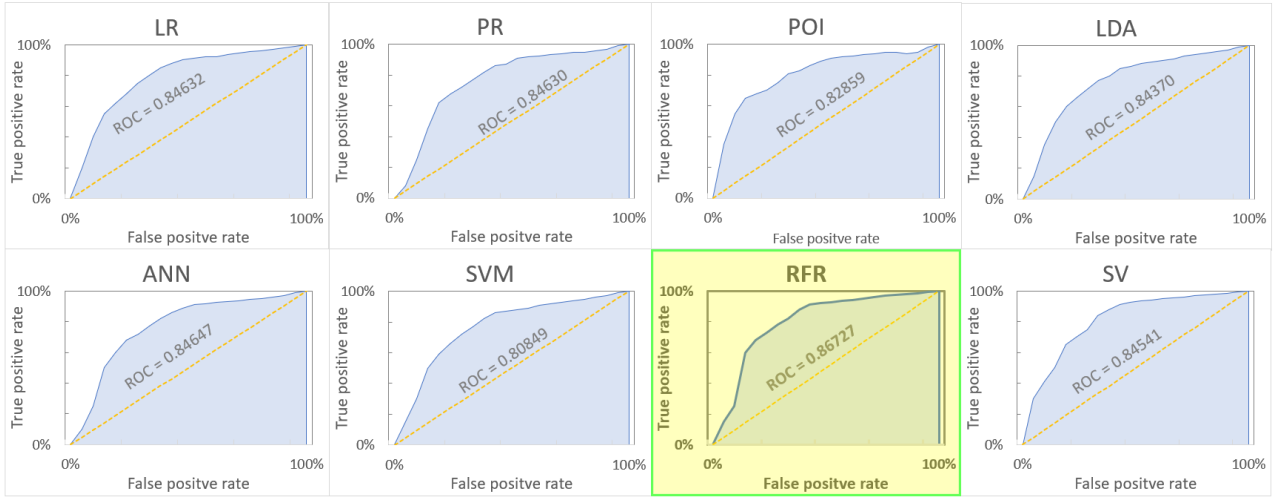


Figure 6.3: ROC ROS Scorecards

6.2.4 Confusion Matrix

Table 6.2 below shows the confusion matrices constructed from the scoring results using the ROS BVT sample. For each scorecard developed, a relevant confusion matrix was constructed, the results of which are used to calculate the Accuracy and Specificity.

		LR		**PR**		**POI**		**LDA**	
		Predicted		**Predicted**		**Predicted**		**Predicted**	
		0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Actual	**0**	114 818	25 182	114 821	25 179	121 015	18 985	114 923	25 077
	1	7 661	27 339	7 664	27 336	26 881	8 119	8 058	26 942
		ANN		**SVM**		**RFR**		**SV**	
		Predicted		**Predicted**		**Predicted**		**Predicted**	
		0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Actual	**0**	114 956	25 044	114 799	25 201	134 848	5 152	113 827	26 173
	1	7 890	27 110	8 283	26 717	1 244	33 756	7 127	27 873

Table 6.2: ROS Confusion Matrices

Based on Table 6.2, the Accuracy and Specificity for the LR scorecard are:

$$Accuracy = \frac{TG + TB}{TOTAL} = \frac{114818 + 27339}{175000} = 0.81232 \tag{6.1}$$

$$Specificity = \frac{TG}{TG + FB} = \frac{114818}{114818 + 25182} = 0.82012 \tag{6.2}$$

The scorecard results using the ROS BVT sample are summarised below in Table 6.3. It can be seen that the RFR scorecard has superior performance amongst scorecards developed using the ROS BVT sample across all methods except for the KS statistic.

Scorecard	GINI	KS	AUC	Accuracy	Specificity
LR	0.69264	0.36036	0.84632	0.81232	0.82012
PR	0.69261	0.36008	0.84630	0.81232	0.82015
POI	0.65719	0.33363	0.82859	0.73790	0.86439
LDA	0.68741	0.35835	0.84370	0.81065	0.82087
ANN	0.69266	**0.36061**	0.84647	0.81181	0.82111
SVM	0.61698	0.31715	0.80849	0.80867	0.81999
RFR	**0.73454**	0.34820	**0.86727**	**0.96345**	**0.96320**
SV	0.69083	0.35938	0.84541	0.80971	0.81304

Table 6.3: ROS Metrics

6.3 CBOS BVT

The sample generated using CBOS had 250 000 observations, of which 57 701 were 'bad', i.e they had an actual outcome of 1. This gives a bad rate of 23.08%, which is higher in the sample compared to the BVT sample, as the BVT sample had a bad rate 14.4% . The CBOS BVT sample was further divided into the Built, Validation and Test samples as seen in Table 6.4.

	Built	Validation	Test	CBOS BVT
# Good	134 609	38 338	19 352	192 299
# Bad	40 391	11 662	5 648	57 701
Proportion of CBOS BVT	70%	20%	10%	100%
Bad Rate	23.08%	23.32%	22.59%	23.08%
Total	**175000**	**50000**	**25000**	**250000**

Table 6.4: CBOS BVT

6.3.1 GINI

Figure 6.4 shows the Gini calculated from the scoring results across the different methods using the CBOS BVT sample. The RFR scorecard, developed with the CBOS sample, outperforms all other scorecards, with a Gini coefficient of 0.70414. The poorest performing scorecard is the one developed using the SVM technique, with a Gini of 0.61415.

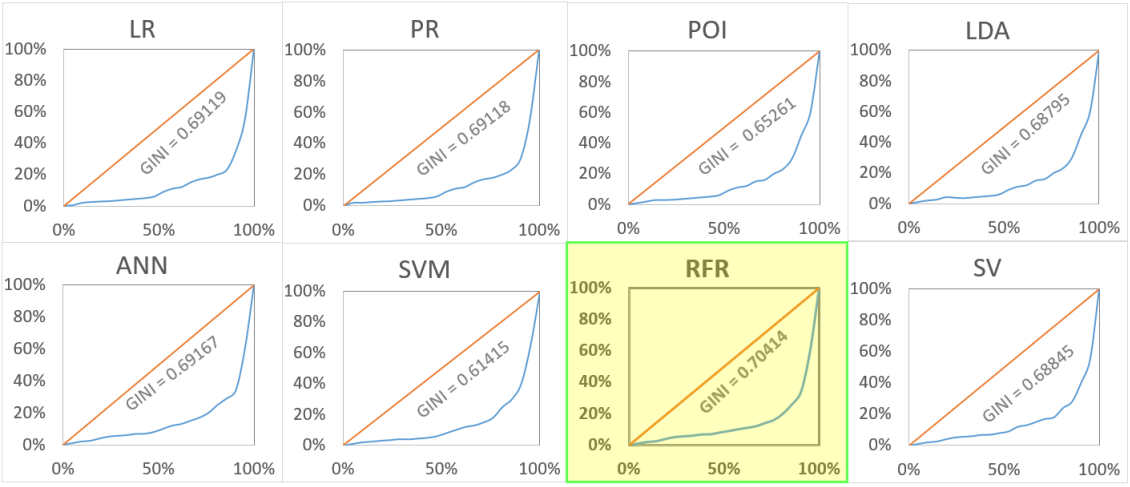


Figure 6.4: Gini CBOS Scorecards

6.3.2 Kolmogorov-Smirnov (KS)

Figure 6.5 below shows the KS statistic calculated from the scoring results using the CBOS BVT sample. The ANN scorecard gives the best KS statistic amongst the scorecards developed using the ROS data with the value of 0.35546, while the RFR scorecard has the lowest KS at 0.31153.

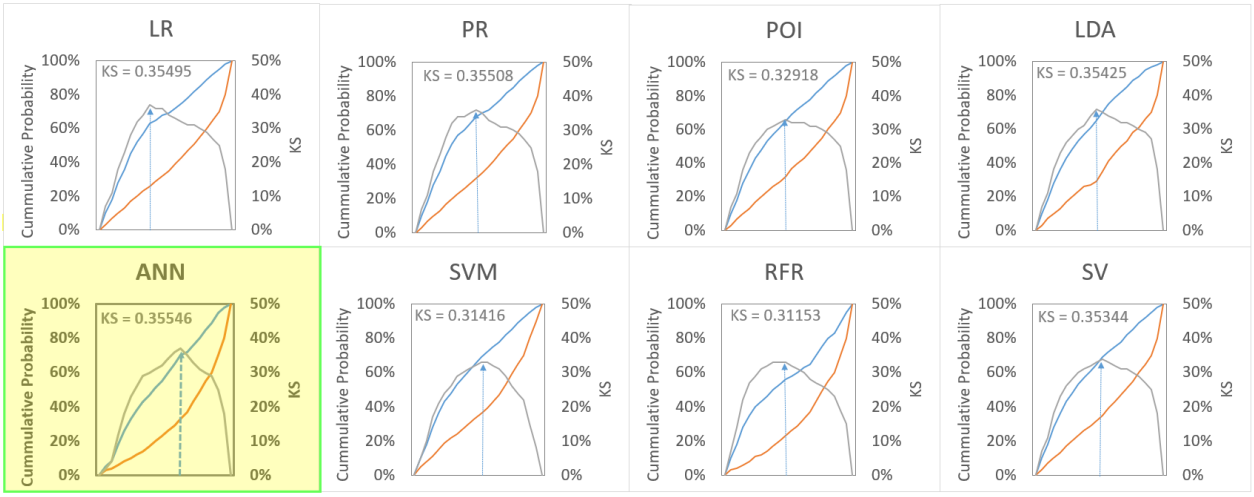


Figure 6.5: KS CBOS Scorecards

6.3.3 ROC Curve

Figure 6.6 below shows the ROC constructed from the scoring results using the CBOS BVT sample and shows the value of the AUC. The RFR scorecard shows the highest AUC from all scorecards built using the CBOS BVT data and SVM has the lowest AUC.

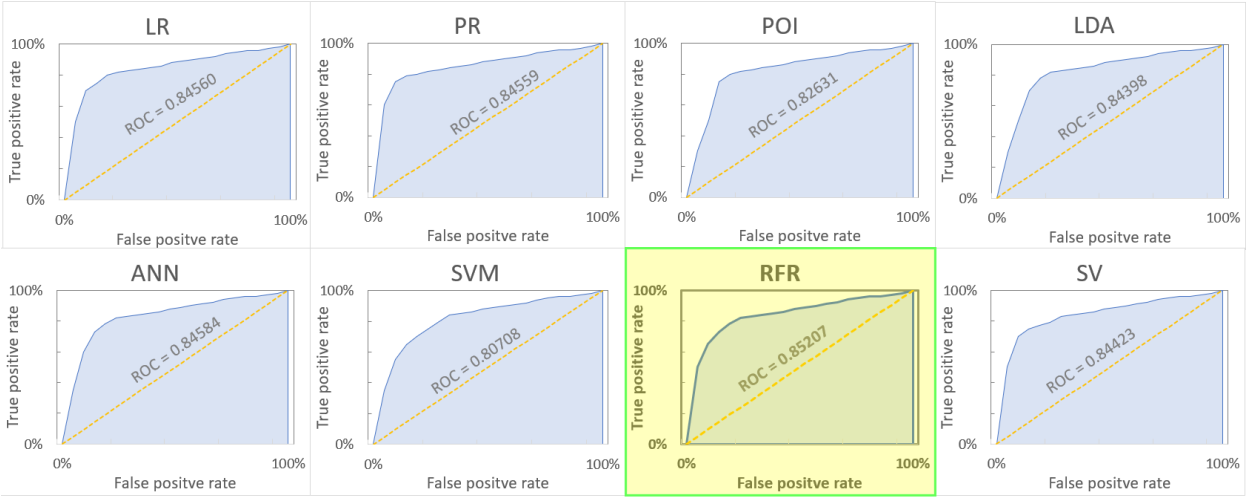


Figure 6.6: ROC CBOS Scorecards

6.3.4 Confusion Matrix

Table 6.5 below shows the confusion matrices constructed from the scoring results using the CBOS BVT sample. For each scorecard developed, a relevant confusion matrix was constructed. The results of which are used to calculate the Accuracy and Specificity.

		LR		**PR**		**POI**		**LDA**	
		Predicted		**Predicted**		**Predicted**		**Predicted**	
		0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Actual	**0**	106 992	27 617	107 000	27 609	111 410	23 199	106 933	27 676
	1	9 935	30 456	9 940	30 451	22 274	18 117	9 977	30 414
		ANN		**SVM**		**RFR**		**SV**	
		Predicted		**Predicted**		**Predicted**		**Predicted**	
		0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Actual	**0**	107 192	27 417	106 417	28 192	131 327	3 282	105 676	28 933
	1	10 152	30 239	9 958	30 433	1 515	38 876	9 147	31 244

Table 6.5: CBOS Confusion Matrices

Based on Table 6.5, the Accuracy and Specificity for the LR scorecard are:

$$Accuracy = \frac{TG + TB}{TOTAL} = \frac{106992 + 30456}{175000} = 0.78541 \tag{6.3}$$

$$Specificity = \frac{TG}{TG + FB} = \frac{106992}{106992 + 27617} = 0.79483 \tag{6.4}$$

The scorecard results using the CBOS generated sample are shown below. As with the ROS data, the RFR scorecard outperforms all other techniques. It is only for KS where RFR is not the best amongst the scorecards. Based on the two sampling techniques, the same conclusion can be made, that the RFR scorecard offers the best performance.

Scorecard	**GINI**	**KS**	**AUC**	**Accuracy**	**Specificity**
LR	0.69119	0.35495	0.84560	0.78541	0.79483
PR	0.69118	0.35508	0.84559	0.78543	0.79489
POI	0.65261	0.32918	0.82631	0.74015	0.82765
LDA	0.68795	0.35425	0.84398	0.78484	0.79439
ANN	0.69167	**0.35546**	0.84584	0.78532	0.79632
SVM	0.61415	0.31416	0.80708	0.78199	0.79056
RFR	0.**70414**	0.31153	**0.85207**	**0.97258**	**0.97562**
SV	0.68845	0.35344	0.84423	0.78239	0.78505

Table 6.6: CBOS Metrics

6.4 SMOTE

The sample generated using SMOTE had 250 000 observations, of which 56 674 were 'bad', i.e they had an actual outcome of 1. This gives a bad rate of 22.67%, which is more than the bad rate of the BVT sample at 14.4%. The SMOTE BVT sample was further divided into the Built, Validation and Test samples as seen in Table 6.7.

	Built	Validation	Test	SMOTE BVT
# Good	129 796	43 032	20 498	193 326
# Bad	45 204	6 968	4 502	56 674
Proportion of SMOTE BVT	70%	20%	10%	100%
Bad Rate	25.83%	13.94%	18.01%	22.67%
Total	**175 000**	**50 000**	**25 000**	**250 000**

Table 6.7: SMOTE BVT

6.4.1 GINI

Figure 6.7 below shows the Gini calculated from the scoring results using the SMOTE BVT sample. The RFR scorecard, developed with the CBOS sample, out-performs all other scorecards, with a Gini coefficient of 0.78733. The poorest performing scorecard is the one developed using the SVM technique, with a Gini of 0.62532.

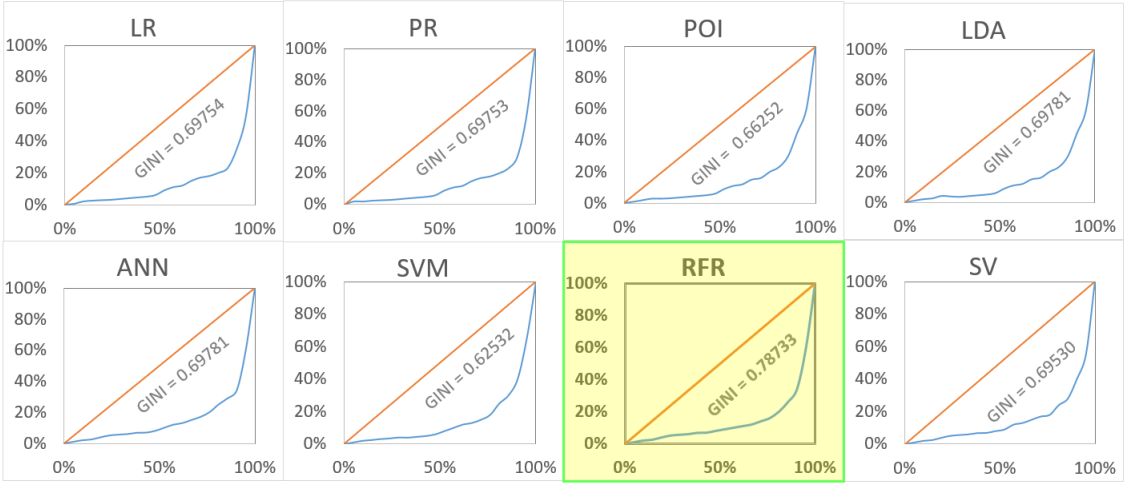


Figure 6.7: Gini SMOTE Scorecards

6.4.2 Kolmogorov-Smirnov (KS)

Figure 6.8 shows the KS statistic calculated from the scoring results using the SMOTE BVT sample. The ANN scorecard gives the best KS statistic amongst the scorecards developed using the ROS data with the value of 0.36081, while the RFR scorecard has the lowest KS at 0.31499.

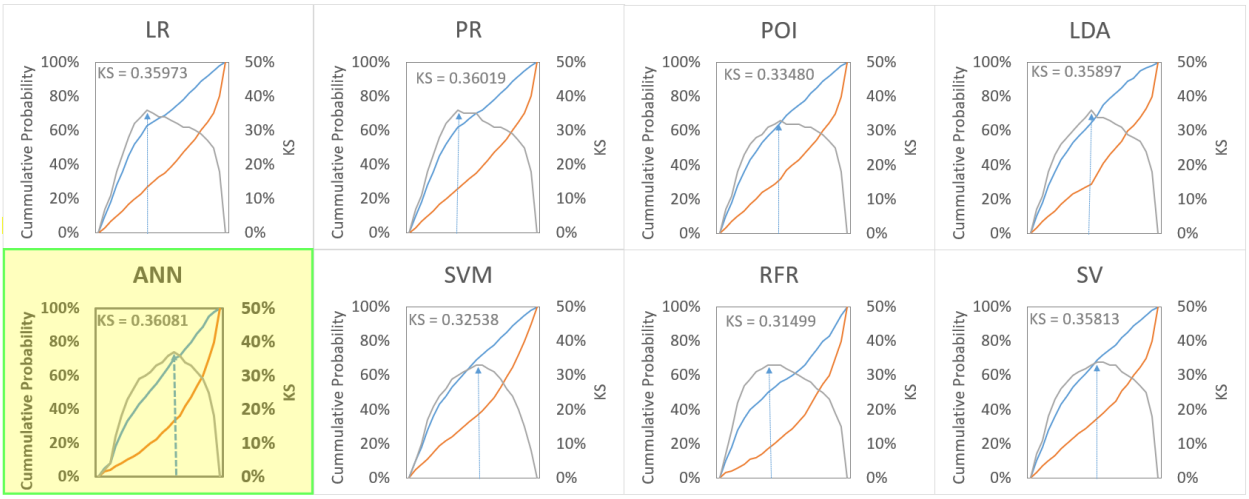


Figure 6.8: KS SMOTE Scorecards

6.4.3 ROC Curve

Figure 6.9 below shows the ROC constructed from the scoring results using the SMOTE BVT sample and shows the value of the AUC. The RFR scorecard shows the highest AUC of all scorecards built using the CBOS BVT data and SVM has the lowest AUC.

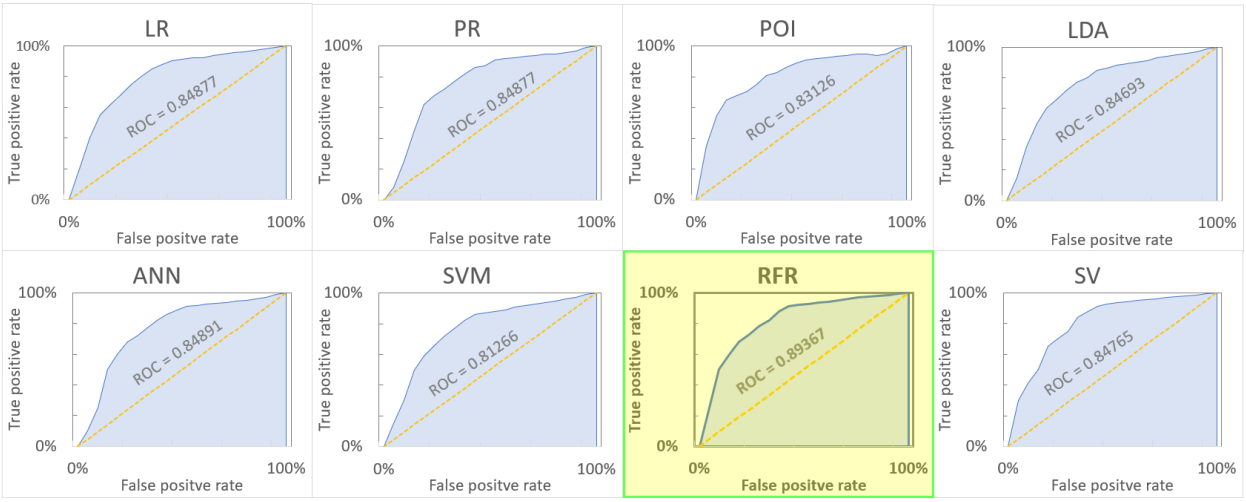


Figure 6.9: ROC SMOTE Scorecards

6.4.4 Confusion Matrix

Table 6.8 below shows the confusion matrices constructed from the scoring results using the SMOTE BVT sample. For each scorecard developed, a relevant confusion matrix was constructed, the results of which are used to calculate the Accuracy and Specificity.

		LR		**PR**		**POI**		**LDA**	
		Predicted		**Predicted**		**Predicted**		**Predicted**	
		0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Actual	**0**	100 745	29 051	100 751	29 045	102 551	27 245	100 405	29 391
	1	11 968	33 236	11 980	33 224	15 385	29 819	11 876	33 328
		ANN		**SVM**		**RFR**		**SV**	
		Predicted		**Predicted**		**Predicted**		**Predicted**	
		0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Actual	**0**	100 986	28 810	99 362	30 434	125 480	4 316	98 819	30 977
	1	12 258	32 946	11 599	33 605	1 696	43 508	11 079	34 125

Table 6.8: SMOTE Confusion Matrices

Based on Table 6.8, the Accuracy and Specificity for the LR scorecard are:

$$Accuracy = \frac{TG + TB}{TOTAL} = \frac{100745 + 33236}{175000} = 0.76560 \tag{6.5}$$

$$Specificity = \frac{TG}{TG + FB} = \frac{100745}{100745 + 29051} = 0.77618 \tag{6.6}$$

The scorecard results using the SMOTE generated sample are shown in Table 6.9 below. The RFR scorecard is the best performing scorecard in four out of the five metrics used. The same conclusion can be drawn for all the sampling techniques.

Scorecard	**GINI**	**KS**	**AUC**	**Accuracy**	**Specificity**
LR	0.69754	0.35973	0.84877	0.76560	0.77618
PR	0.69753	0.36019	0.84877	0.84877	0.77622
POI	0.66252	0.33480	0.83126	0.75639	0.79009
LDA	0.69385	0.35897	0.84693	0.76418	0.77356
ANN	0.69781	**0.36081**	0.84891	0.76532	0.77803
SVM	0.62532	0.32538	0.81266	0.75981	0.76552
RFR	**0.78733**	0.31499	**0.89367**	**0.96564**	**0.96675**
SV	0.69530	0.35813	0.84765	0.75968	0.76134

Table 6.9: SMOTE Metrics

6.5 Summary

The results for all the scorecards from all the sampling techniques are shown in Table 6.10 below. For all the sampling techniques, the RFR scorecards provide the best relative performance. There is also performance improvements brought about by the sampling technique used. The SMOTE BVT managed to produce the best performance for the RFR scorecard, while CBOS produced the worst performing of the RFR scorecards. As seen in Table 6.10, the other scorecards show the best performance from the ROS sample.

	Scorecard	GINI	KS	AUC	Accuracy	Specificity
ROS	LR	0.69264	0.36036	0.84632	0.81232	0.82012
	PR	0.69261	0.36008	0.84630	0.81232	0.82015
	POI	0.65719	0.33363	0.82859	0.73790	0.86439
	LDA	0.68741	0.35835	0.84370	0.81065	0.82087
	ANN	0.69266	0.36061	0.84647	0.81181	0.82111
	SVM	0.61698	0.31715	0.80849	0.80867	0.81999
	RFR	0.73454	0.34820	0.86727	0.96345	0.96320
	SV	0.69083	0.35938	0.84541	0.80971	0.81304
CBOS	LR	0.69119	0.35495	0.84560	0.78541	0.79483
	PR	0.69118	0.35508	0.84559	0.78543	0.79489
	POI	0.65261	0.32918	0.82631	0.74015	0.82765
	LDA	0.68795	0.35425	0.84398	0.78484	0.79439
	ANN	0.69167	0.35546	0.84584	0.78532	0.79632
	SVM	0.61415	0.31416	0.80708	0.78199	0.79056
	RFR	0.70414	0.31153	0.85207	0.97258	0.97562
	SV	0.68845	0.35344	0.84423	0.78239	0.78505
SMOTE	LR	0.69754	0.35973	0.84877	0.76561	0.77618
	PR	0.69753	0.36019	0.84877	0.76557	0.77622
	POI	0.66252	0.33480	0.83126	0.75640	0.79009
	LDA	0.69385	0.35897	0.84693	0.76419	0.77356
	ANN	0.69781	0.36081	0.84891	0.76533	0.77803
	SVM	0.62532	0.32538	0.81266	0.75981	0.76552
	RFR	0.78733	0.31499	0.89367	0.96565	0.96675
	SV	0.69530	0.35813	0.84765	0.75968	0.76134

Table 6.10: All Metrics

Chapter 7

Out of Time Performance

7.1 Introduction

This chapter assesses how the developed scorecards perform as new data is introduced. This simulates the real life scenario when new unseen customers are assessed using the developed model. Firstly the stability of the variables between the BVT and OOT periods is assessed. Thereafter, two scorecard implementation approaches are shown and details of both approaches are also given. The performances for the various scorecards are shown and compared per approach. Then the results are summarised.

7.2 Variable Stability

Before scoring the OOT data, it is important to compare its distribution with the BVT data. If the two samples are distributed differently, then the results from scoring the OOT data would be affected. The stability index is used for this purpose, as indicated by Siddiqi (2017). Table 7.1 shows the calculation of the variable stability for the variable shows the number of active revolving accounts(NUM_ACTV_REV_TL) using Equation 3.51.

Bins	% BVT	% OOT	LN(% BVT/% OOT)	% BVT - % OOT	Stability Index
[Low, 4.0)	26.62%	20.00%	0.28608	6.62%	18.95
[4.0, 6.0)	29.71%	41.11%	-0.32490	-11.40%	37.05
[6.0, 8.0)	20.82%	20.56%	0.01257	0.26%	0.03
[8.0, 10.0)	11.53%	11.83%	-0.02577	-0.30%	0.08
[10.0, High)	11.32%	6.50%	0.55486	4.82%	26.75
	100.00%	100.00%			82.86

Table 7.1: NUM_ACTV_REV_TL STABILITY

Figure 7.1 shows the stability index of the OOT data compared to the BVT. Most of the variables have a stability index in the green area. This then shows that there is no evidence of a shift in the distribution of the OOT data as compared to the BVT data, which is highly desirable. There are four variables that show evidence of a slight shift in their respective distributions namely: INT_RATE, TOTAL_REC_LATE_FEE, OPEN_IL_12M and TOT_CUR_BAL. Even though the variables are slightly not stable it, they are closer to being stable than not and do not warrant changes to the variable bins as at BVT.

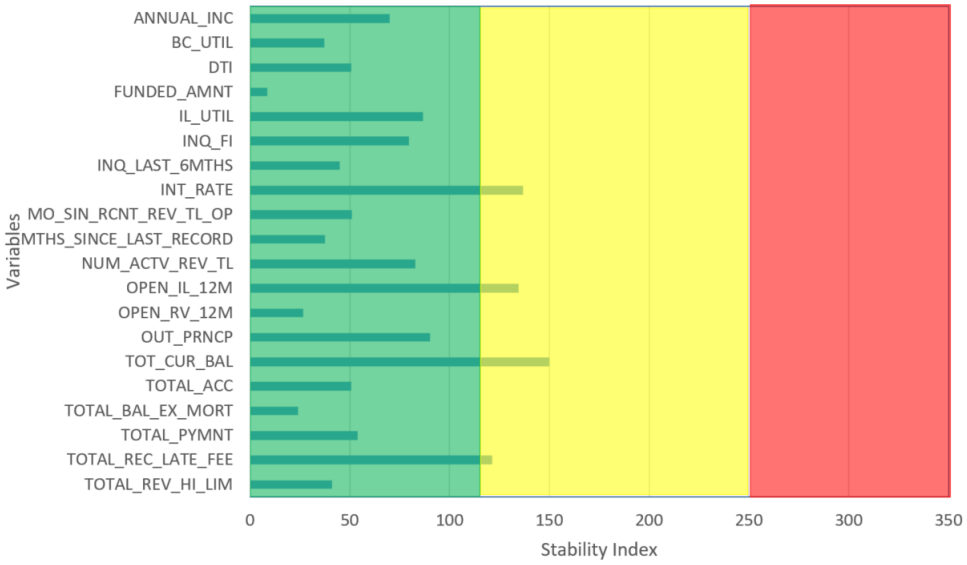


Figure 7.1: Variable Stability

Stability is also desired in the performance metrics, because a stable metric can be predicted with a higher confidence level than an unstable one. Equation 3.51 can be modified slightly to calculate the stability index for the different metrics. The analysis below is concerned with the absolute size of the metric and its relative stability.

7.3 OOT Implementation

This research investigates two approaches that can be employed to implement the scorecards. The implementation and testing of the scorecards requires the partial scores from each of the variables in the scorecard, which is developed using the BVT data and the monthly OOT data. The static approach is to score each month in the OOT period using the BVT scorecard. This means that the BVT partial scores do not change. Instead, there will be a regular review of the scorecards, and redevelopment may be required if the performance deteriorates beyond the Bank's acceptable levels. This is the approach that is currently adopted by many Banks, as noted by Siddiqi (2017). The dynamic approach does not use constant scores over the lifetime of the scorecard. This is achieved by scoring the first monthly OOT data (first month of the OOT period) as was done in the static approach, then immediately updating the BVT scores, before scoring the next monthly data.

In essence, the scorecards are retrained using the current month's OOT data as part of the BVT data, to produce new scores which are expected to fit the data properly. The idea is for the scorecard to adapt as the data is scored, which should fundamentally produce better results. The following sections show the results from the two approaches of the OOT scoring. The results from the static approach will be indicated by the word "STATIC" in their caption, while the ones from the dynamic approach will have the word "DYNAMIC" in their caption. Figures 7.2 and 7.3 give a visual representation of approaches one and two respectively.

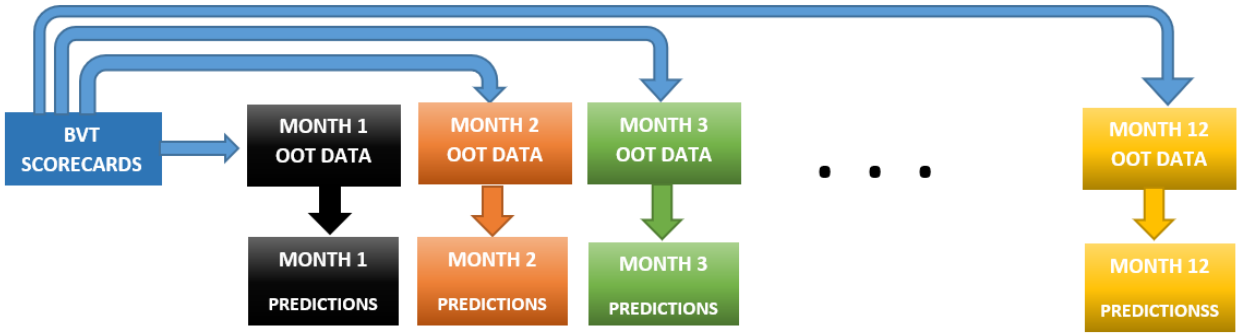


Figure 7.2: Static Approach

Figure 7.3: Dynamic Approach

7.4 ROS OOT

This section gives the OOT results produced by the scorecards developed using the ROS data. The scorecards are implemented using two approaches as outlined above. The scorecard results are quantified by Gini, KS statistic, AUC, Accuracy and Specificity. Each of the measures will be presented for both implementation approaches.

Figure 7.4: ROS OOT process

7.4.1 GINI

Figure 7.5 shows the static approach to implementing the scorecards. The RFR scorecard showed the best Gini for the ROS BVT data. The scorecard's performance then deteriorates for the OOT period and is not stable. Figure 7.5 shows the dynamic approach to implementing the scorecards. The BVT and first month of the OOT performance is similar to that of the static approach. From the second month, the scores have been updated with the first month's data. The OOT Gini improves for nearly all of the scorecards. The SVM scorecard shows the most improved Gini.

The RFR and ANN scorecards also show improvement in Gini from the first month of the OOT period. The POI scorecard shows the lowest Gini in the OOT period, while others

do not improve immediately into the OOT period. The scorecards developed using other techniques are more stable than the RFR scorecard. The SV scorecard is also less stable, but the OOT Gini is better than the RFR scorecard. The PR scorecard is the most stable of them all, as seen in Figure 7.5, with the PR Gini looking quite linear.

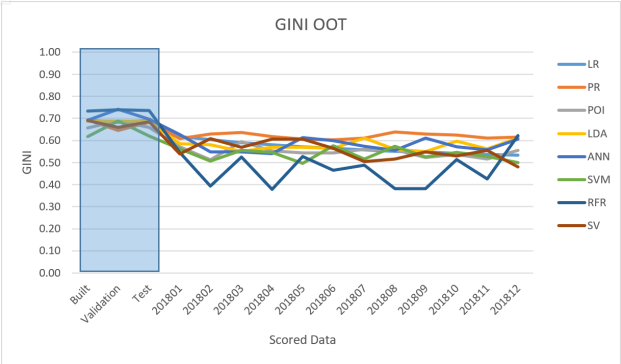


Figure 7.5: ROS GINI OOT STATIC

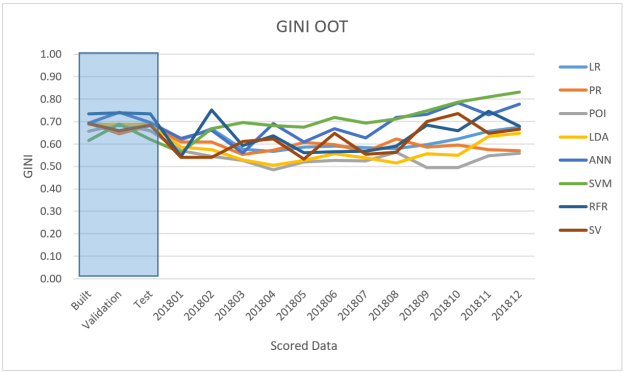


Figure 7.6: ROS GINI OOT DYNAMIC

Figure 7.6 shows the Gini performance for the dynamic approach of implementing the scorecards. The BVT and first month of the OOT performance are similar to the static approach. From the second month, the scores have been updated with the first month's data. The OOT Gini improves for nearly all of the scorecards. The SVM scorecard shows the most improved Gini. The RFR and ANN scorecards also show improvements in Gini from the first month of the OOT period. The POI scorecard shows the lowest Gini in the OOT period, while others do improve at various times into the period.

7.4.2 Kolmogorov-Smirnov (KS) statistic

Figure 7.7 shows the KS statistic performance resulting from the first implementation approach. The KS statistic deteriorates for all the scorecards in the OOT data. Despite the deterioration, PR and LR show straighter lines in Figure 7.7 indicating that they are more stable over time than the other scorecards and that these scorecards adapt to the OOT data fairly quickly. The SVM scorecard deteriorates heavily in the first months of the OOT periods, then stabilises in the later months. The RFR scorecards deteriorate initially, then continue to deteriorate over the months. The improvements are purely due to how the OOT data changes over the period, since the BVT scorecards are not being updated.

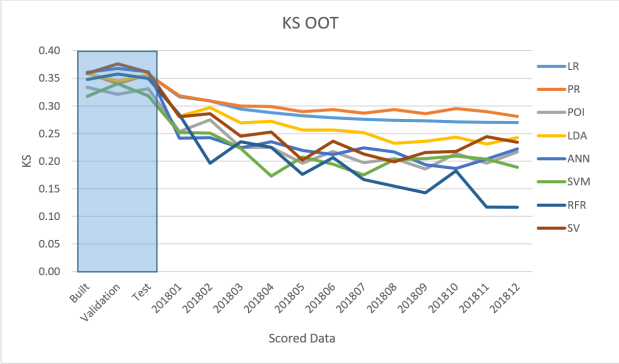


Figure 7.7: ROS KS OOT STATIC

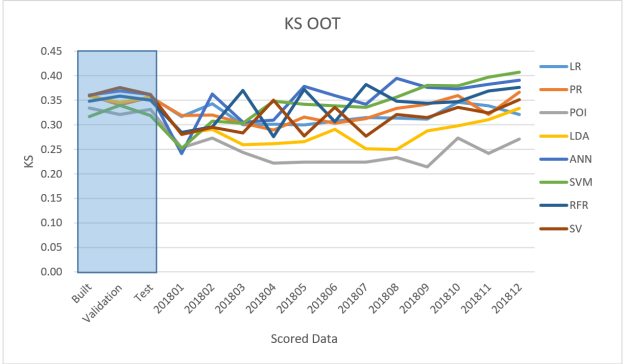


Figure 7.8: ROS KS OOT DYNAMIC

Figure 7.8 shows the OOT KS performance resulting from the dynamic approach for the ROS sample. The KS statistic is shown to improve for most of the scorecards, and it also shows signs of instability. The LDA and POI scorecards show the least improved KS statistic in the initial months of the OOT period, but there are improvements in the later months. The most improved KS statistic is for the SVM scorecard.

7.4.3 AUC

Figure 7.9 shows the OOT AUC performance resulting from the first implementation approach for the ROS sample. The AUC for all scorecards is very stable and does not drop off significantly in the OOT period when compared to the BVT data. The RFR scorecard dropped off in AUC more than all other scorecards and is less stable than all other scorecards. As seen in Figure 7.9, the orange line shows that the PR scorecard is more stable than all others.

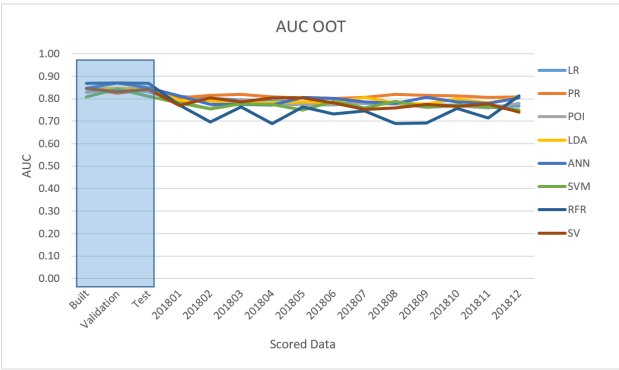


Figure 7.9: ROS AUC OOT STATIC

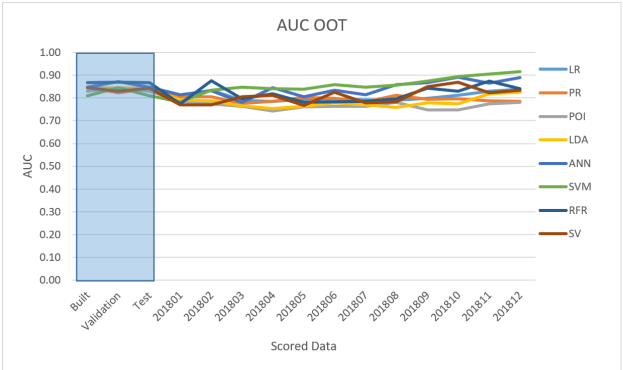


Figure 7.10: ROS AUC OOT DYNAMIC

Figure 7.10 shows the OOT AUC performance resulting from the second implementation

approach for the ROS sample. Most of the lines on the figure are sloping upwards, and this shows improvement in the OOT scores. The SVM shows the best AUC performance in the OOT period, and is the most improved scorecard as well.

7.4.4 ACCURACY

Figure 7.11 shows the OOT Accuracy performance resulting from the first implementation approach for the ROS sample and the same trend as seen in the previous metrics continues with the Accuracy metric. The RFR scorecard has the biggest drop off between BVT and OOT. The RFR score is also less stable than the other scorecards. The PR and LR scorecards are more stable as seen in Figure 7.11.

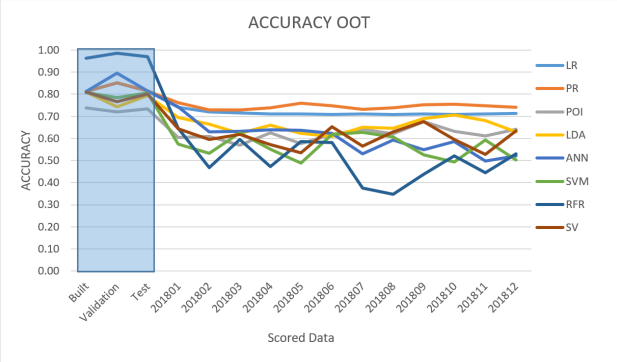


Figure 7.11: ROS ACCURACY OOT STATIC

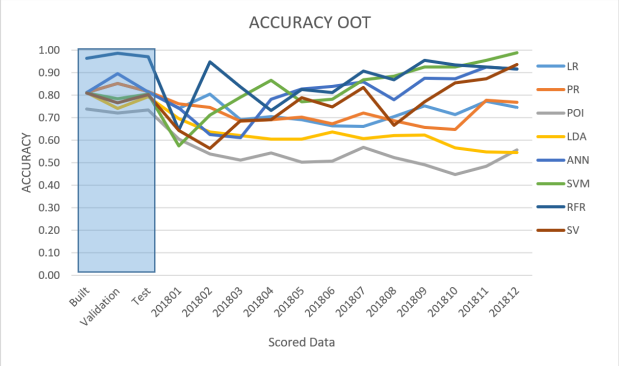


Figure 7.12: ROS ACCURACY OOT DYNAMIC

Figure 7.12 shows the OOT Accuracy performance resulting from the second implementation approach. The scorecards with the highest Accuracy measure at the end of the OOT period are the SVM, SV and RFR. During the OOT period the SV scorecard is the least stable from month to month. The POI and LDA scorecards show the least Accuracy measure, and there is deterioration over the OOT months.

7.4.5 SPECIFICITY

Figure 7.13 shows the OOT Specificity performance resulting from the first implementation approach for the ROS sample. For specificity, all the scorecards drop off in a similar manner and most of them are also not stable. Only PR and LR scorecards are more stable in the OOT period. The RFR scorecard is the most unstable of the scorecards, while the SV scorecard is stable but with a lower specificity value than the LR and PR scorecards. There

are however, hints of higher specificity values in the later months of the OOT period for the POI and RFR scorecards.

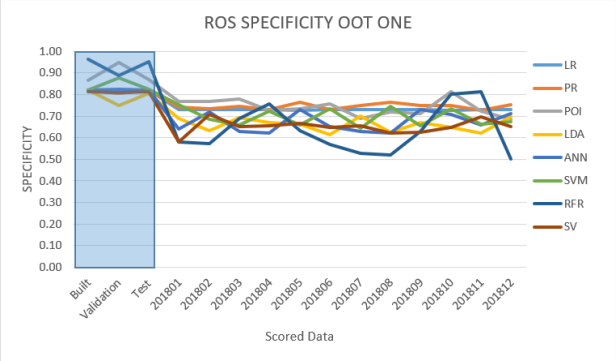


Figure 7.13: SPECIFICITY OOT STATIC

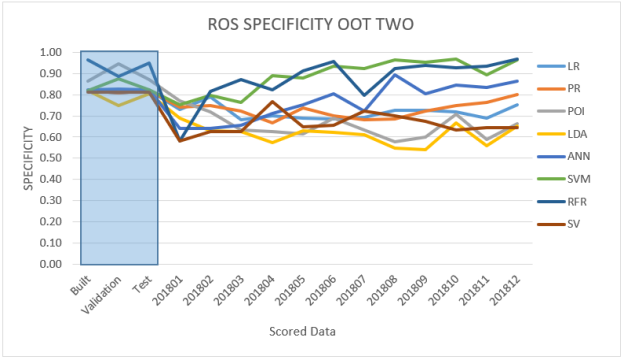


Figure 7.14: SPECIFICITY OOT DYNAMIC

Figure 7.14 shows the OOT Specificity performance resulting from the second implementation approach. The machine learning scorecards are showing the most improvement for specificity. The PR scorecard also shows improvements in the later months of the OOT period, while the SV scorecard improves slightly and then deteriorates in the same period.

7.5 CBOS OOT

This section gives the OOT results produced by the scorecards developed using the CBOS data. The scorecards are implemented using two approaches as outlined above. The scorecard results are quantified by Gini, KS statistic, AUC, Accuracy and Specificity. Each of the measures will be presented for both implementation approaches.

Figure 7.15: CBOS OOT process

7.5.1 GINI

Figure 7.16 shows the OOT Gini performance resulting from the first implementation approach for the CBOS sample. The RFR scorecard showed the best performance for the SMOTE BVT data with a Gini of over 0.7. The scorecard Gini then falls to below 0.4 and is not stable for the OOT period. However, it shows improvement in stability, towards the end of the OOT period. All other scorecards are relatively more stable and the SV scorecard shows a slight improvement in the Gini value but is still unstable. The PR scorecard has the best performance in the OOT period and it is more stable.

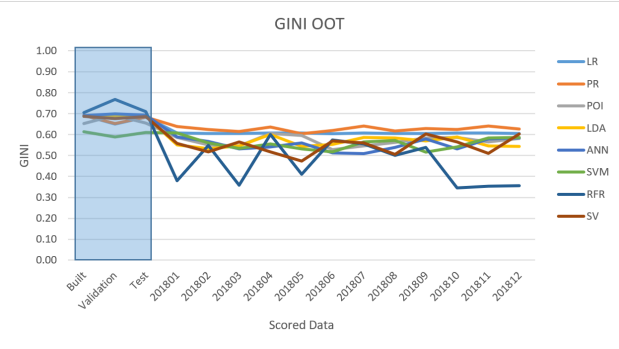


Figure 7.16: CBOS GINI OOT STATIC

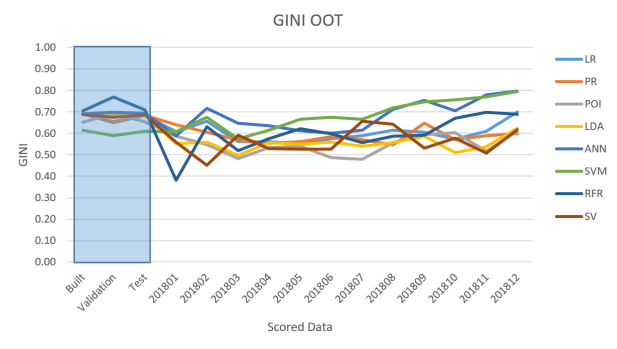


Figure 7.17: CBOS GINI OOT DYNAMIC

Figure 7.17 shows the OOT Gini performance resulting from the second implementation approach. The SVM and ANN scorecards show the most Gini improvement in the OOT period. All the scorecards are very unstable. The SV scorecard improves towards the end of the OOT period.

7.5.2 Kolmogorov-Smirnov (KS) statistic

Figure 7.18 shows the OOT KS performance resulting from the first implementation approach for the CBOS sample. All scorecards follow a similar trend when scoring the OOT data. The scorecard that maintains a higher KS statistic is PR, and it is relatively more stable as seen in Figure 7.18. The SV model is also seen to maintain a relatively higher KS statistic than most scorecards, even though it is not stable. The RFR scorecard is seen to have a big drop off in the KS value and shows the lowest KS value for the entire OOT period even though there is a slight improvement towards the end of the OOT period.

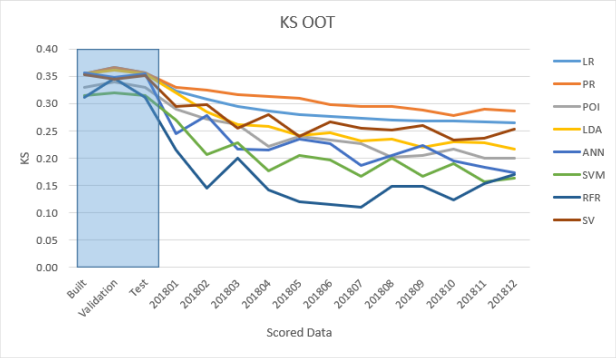


Figure 7.18: CBOS KS OOT STATIC

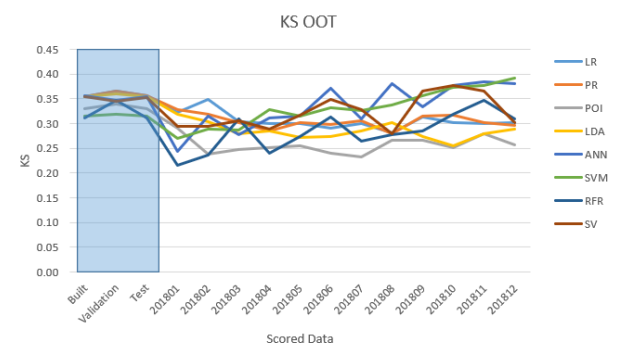


Figure 7.19: CBOS KS OOT DYNAMIC

Figure 7.19 shows the OOT KS performance resulting from the second implementation approach. The overall trend is different from the one shown in Figure 7.18. The SVM and ANN scorecards have the highest KS statistic at the end of the OOT period, while the LDA and POI scorecards have the lowest KS statistic. The RFR improves over the OOT period, but is unstable.

7.5.3 AUC

Figure 7.20 shows the OOT AUC performance resulting from the first implementation approach for the CBOS sample, and the scorecards are competitive. The AUC does not significantly drop off in the OOT period. This is not the case for the RFR scorecard, however, which has the lowest AUC.

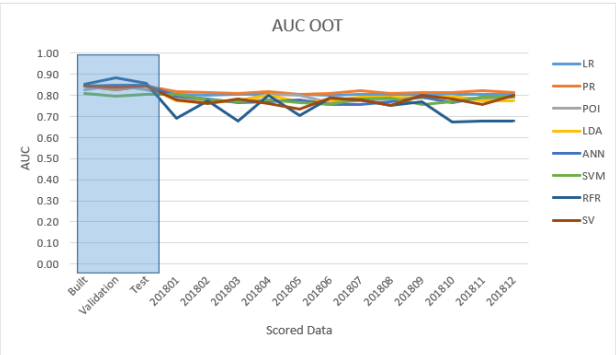


Figure 7.20: CBOS AUC OOT STATIC

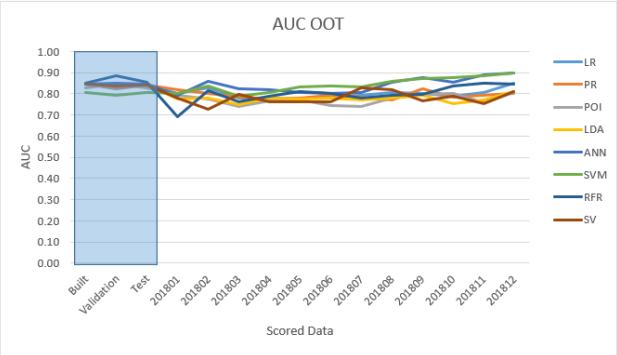


Figure 7.21: CBOS AUC OOT DYNAMIC

Figure 7.21 shows the OOT KS performance resulting from the second implementation approach. Overall, there is an upward trend in AUC. The RFR scorecard significantly drops off in the first month of the OOT period, and recovers in the month after the first.

7.5.4 ACCURACY

Figure 7.22 shows the OOT Accuracy performance resulting from the first implementation approach for the CBOS sample. The RFR scorecard has an almost perfect Accuracy for the BVT data, which heavily drops off to be the lowest at the beginning of the OOT period. The Accuracy improves in the middle period of the OOT period, and drops off again in the later months. A similar trend is followed by the SVM and ANN scorecards. There are some parts of the OOT period further away from the BVT period that show slight improvements. The PR and LR scorecards are more stable, with the PR scorecard just edging the LR with a higher accuracy value as seen in Figure 7.22.

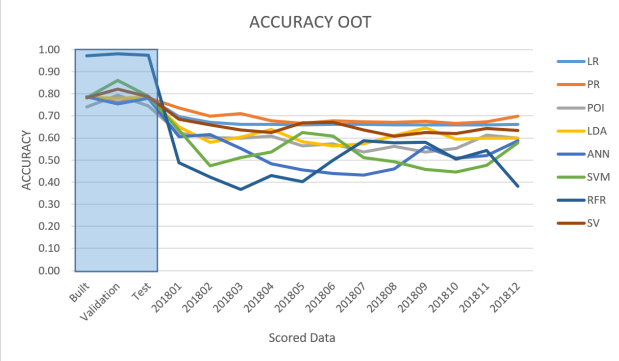


Figure 7.22: CBOS ACCURACY OOT STATIC

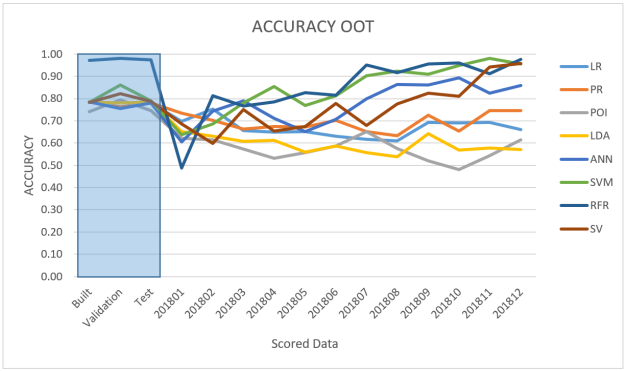


Figure 7.23: CBOS ACCURACY OOT DYNAMIC

Figure 7.21 shows the OOT Accuracy performance resulting from the second implementation approach. All the scorecards are not stable in the OOT period. The scorecards showing the most improvement are the SVM, ANN, SV and RFR. The LDA and POI scorecards show the least Accuracy in the OOT period among the scorecards.

7.5.5 SPECIFICITY

Figure 7.24 shows the OOT Specificity performance resulting from the first implementation approach for the CBOS sample. The RFR scorecard is the most unstable for Specificity. It does, however, show the best value for Specificity for some months in the OOT period. The LDA scorecard performs better than most of the scorecards, and is relatively more stable. The SV scorecard gives the worst Specificity values for the OOT data and is also unstable.

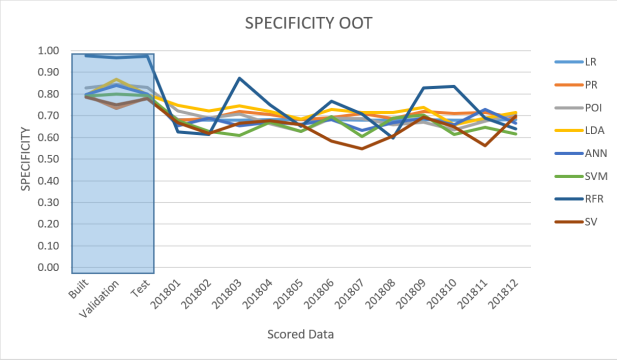


Figure 7.24: CBOS SPECIFICITY OOT STATIC

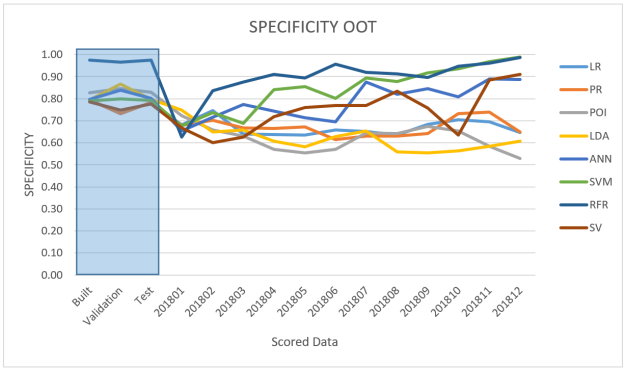


Figure 7.25: CBOS SPECIFICITY OOT DYNAMIC

Figure 7.25 shows the OOT Specificity performance resulting from the second implementation approach. The RFR scorecard shows the most improvement. It significantly drops off in the first month of the OOT period, and improves from there onwards. The LDA and POI scorecard show a worse performance than the first implementation approach.

7.6 SMOTE OOT

This section gives the OOT results produced by the scorecards developed using the SMOTE data. The scorecards are implemented using two approaches as outlined above. The scorecard results are quantified by Gini, KS statistic, AUC, Accuracy and Specificity. Each of the measures will be presented for both implementation approaches.

Figure 7.26: SMOTE OOT process

7.6.1 GINI

Figure 7.27 shows the OOT Gini performance resulting from the first implementation approach for the SMOTE sample. The Gini for the RFR scorecard is the highest for the BVT data and lowest for the OOT data. It is also more unstable, which makes the performance difficult

to predict. All other scorecards are relatively more stable. The PR scorecard shows the best Gini of all scorecards.

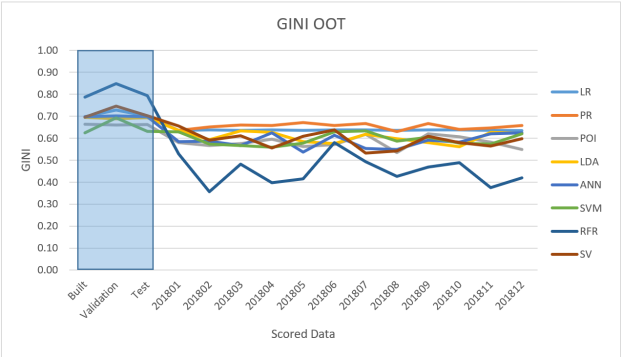


Figure 7.27: SMOTE GINI OOT STATIC

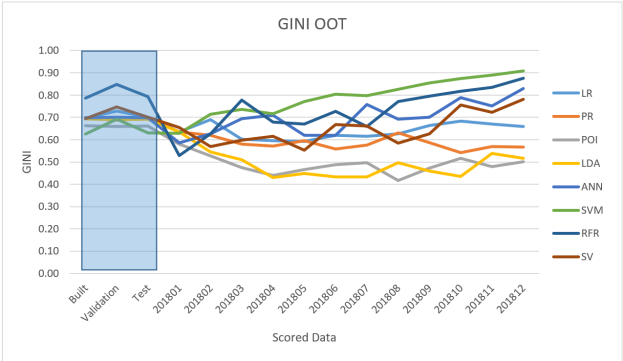


Figure 7.28: SMOTE GINI OOT DYNAMIC

Figure 7.28 shows the OOT Gini performance resulting from the second implementation approach. The SVM scorecard shows the most improved Gini of all the scorecards. This is followed by the RFR, ANN and SV scorecard. Figure 7.28 shows the OOT Gini performance resulting from the second implementation approach. The LDA and POI scorecards do not show improvement in the OOT period. In fact, the performance is worse than in the first implementation approach.

7.6.2 Kolmogorov-Smirnov (KS) statistic

Figure 7.29 shows the OOT KS statistic performance resulting from the first implementation approach for the SMOTE sample. The KS statistic for all scorecards drops off significantly between the BVT and OOT data. The RFR scorecard shows the biggest drop off between BVT and OOT. The PR and LR scorecards are more stable and show the best KS statistic value. The SV scorecard shows slight instability while maintaining a relatively high KS statistic value.

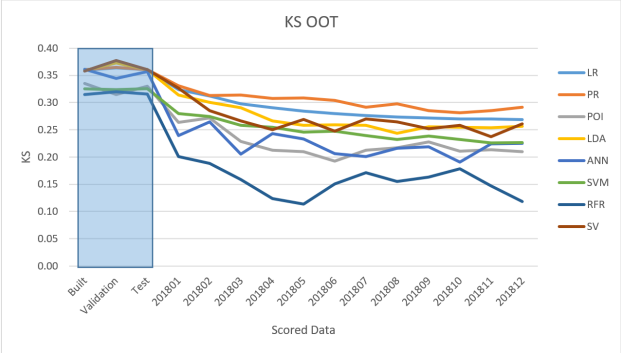


Figure 7.29: SMOTE KS OOT STATIC

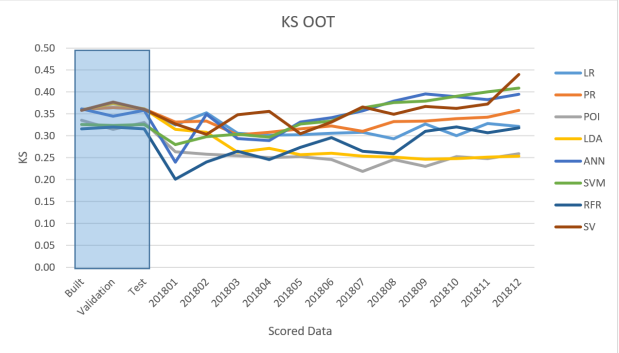


Figure 7.30: SMOTE KS OOT DYNAMIC

Figure 7.30 shows the OOT KS statistic performance resulting from the second implementation approach. The KS statistic improves for most of the scorecards in the OOT period. For POI and LDA, the KS statistic remains relatively similar to the one obtained from the implementation of the static approach. Stability is a concern for the SV and RFR scorecards even though the performance is trending upwards.

7.6.3 AUC

Figure 7.31 shows the OOT AUC performance resulting from the first implementation approach for the SMOTE sample. Most scorecards are fairly stable in the OOT period. The RFR scorecard shows instability and a lower AUC than all other scorecards. The PR scorecard shows a slightly better AUC than all other scorecards.

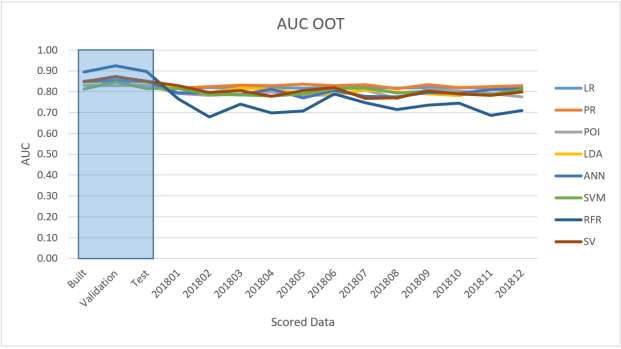


Figure 7.31: SMOTE AUC OOT STATIC

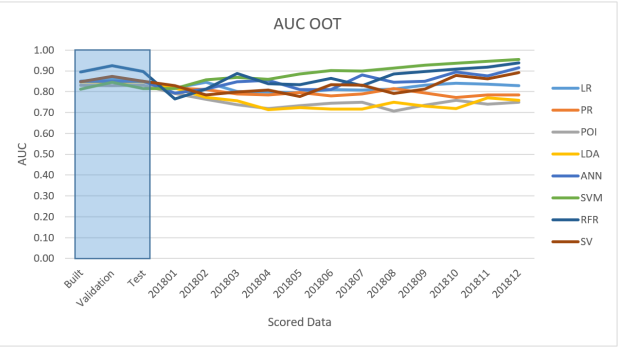


Figure 7.32: SMOTE AUC OOT DYNAMIC

Figure 7.32 shows the OOT Gini performance resulting from the second implementation approach. All scorecards have an AUC that is improved in the OOT period, except for the

LDA and POI scorecards. The SVM scorecard has the most improved AUC in the OOT period.

7.6.4 ACCURACY

Figure 7.33 shows the OOT Accuracy performance resulting from the first implementation approach for the SMOTE sample. There is a drop off for all scorecards between the BVT and OOT data and most of the scorecards are unstable in the OOT period, which makes the performance unpredictable. The RFR scorecard gives the lowest Accuracy values in the OOT period, as with the scorecards developed with the previous sampling techniques.

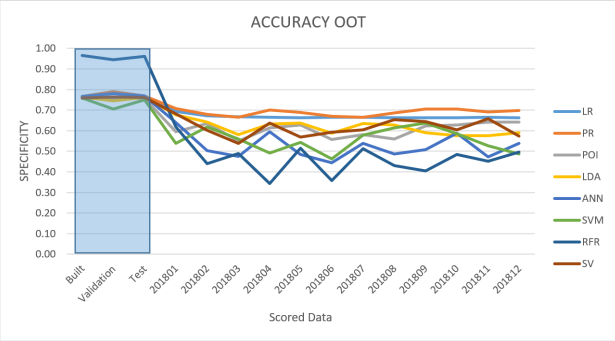


Figure 7.33: SMOTE ACCURACY OOT STATIC

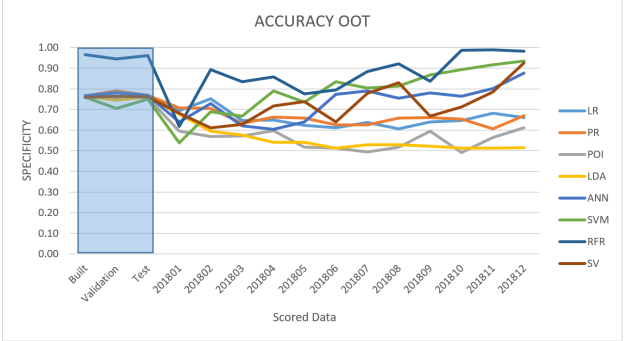


Figure 7.34: SMOTE ACCURACY OOT DYNAMIC

Figure 7.34 shows the OOT Accuracy performance resulting from the second implementation approach. Most of the scorecards show improved accuracy in the OOT period, and the one with the highest is the RFR scorecard. The RFR and SV scorecard are very unstable from month to month. The ANN and SVM scorecard improve with time in the OOT period.

7.6.5 SPECIFICITY

Figure 7.33 shows the OOT specificity performance resulting from the first implementation approach for the SMOTE sample. The RFR scorecard is the most unstable of them all, but it shows the best values of specificity. The other scorecards are more stable, and hence their Specificity can be predicted with a higher level of confidence. The PR scorecard has the second best specificity value behind the RFR scorecard.

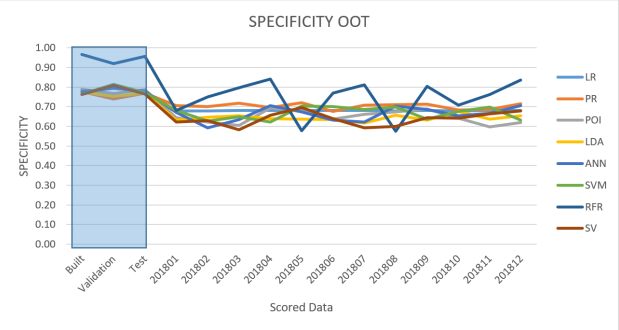


Figure 7.35: SMOTE SPECIFICITY OOT STATIC

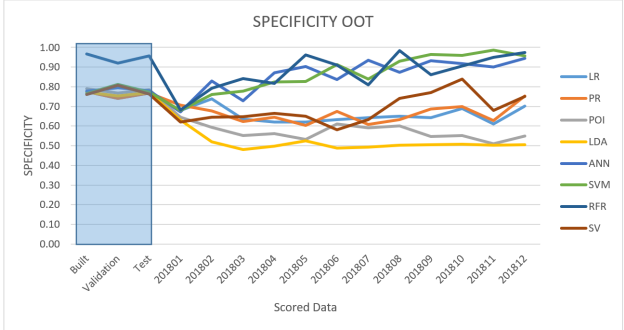


Figure 7.36: SMOTE SPECIFICITY OOT DYNAMIC

Figure 7.36 shows the OOT Specificity performance resulting from the second implementation approach. There is a clear separation in the OOT Specificity performance between the different scorecards. The ML scorecard improved at a higher rate than the others, followed closely by the SV scorecard.

7.7 Summary

The BVT and OOT performance for the scorecards show some differences. One key observation is that for all the scorecards and both approaches, there seems to be an effect were there is a drop in performance in the first month of the OOT period. This study measured the total change of the metric between BVT and OOT for all the scorecards, then an average change in metrics for each scorecard was compiled. The average change compares the BVT averages and the average change for all the months. This was done for both the "static" and "dynamic" approach. The scorecard with the lowest average change is preferred as no regular redevelopment of the scorecards will be needed. For instance, based on the static approach, the average change for the LR scorecard is -16%, which is calculated by averaging the change in Gini, KS, AUC, Accuracy and Specificity. Table 7.2 shows the average change based on the static approach. A positive change shows that the scorecard has improved in the OOT period compared to the BVT, while a negative sign shows that the scorecard has deteriorated as compared to the BVT.

As seen below, the PR scorecard has the lowest average change of all three sample datasets, while the RFR scorecard deteriorates the most. The average change is also slightly different for each sample, which shows that the sampling technique has an effect on the final result. It can be seen that the SV scorecard change is less with the CBOS data while the POI scorecard changes the most with SMOTE data and RFR changes less with ROS data. Most other scorecards have the smallest average changes when SMOTE data is used.

	Average Change		
	ROS	**CBOS**	**SMOTE**
LR	-16%	-15%	-13%
PR	-11%	-11%	-9%
POI	-18%	-18%	-20%
LDA	-17%	-20%	-16%
ANN	-21%	-23%	-18%
SVM	-25%	-20%	-17%
RFR	-36%	-42%	-38%
SV	-24%	-15%	-17%

Table 7.2: Metric Average change STATIC

Table 7.3 shows the average change based on the dynamic approach. Most of the scorecards have improved when compared to the static approach. The only scorecard that has deteriorated using the dynamic approach is the LDA scorecard developed using the SMOTE sample, with an average change of -27% compared to -20% for the static approach. All the ML and SV scorecards have improved compared to the BVT. In Scorecards where there is deterioration, it is small.

The SM scorecards have all deteriorated, with the highest deterioration seen on the LDA in the SMOTE sample. All other SM scorecards deteriorate slightly but are better than the static approach. Overall, the dynamic approach has improved the majority of the scorecards.

	Average Change		
	ROS	**CBOS**	**SMOTE**
LR	-6%	-10%	-11%
PR	-6%	-12%	-11%
POI	-17%	-17%	-20%
LDA	-14%	-16%	-27%
ANN	9%	10%	14%
SVM	23%	24%	27%
RFR	-1%	0%	5%
SV	-2%	2%	6%

Table 7.3: Metric Average change DYNAMIC

The study also measured the stability index for each metric in every scorecard and every sampling technique for both the first and dynamic approach. The average stability for each scorecard developed using the static approach was then compiled and is shown in Table 7.4. The scorecard with a low value for the stability index is preferred as this shows that the results are more predictable. A large stability index value shows an unstable scorecard. The LR scorecard is the most stable scorecard in all the sample datasets and it is most stable

with the SMOTE data, while the RFR scorecard is the least stable scorecard and performs the worst for the CBOS data. Most scorecards are at their most stable when the SMOTE dataset is used, and only the ANN and POI scorecards are at their most stable when using CBOS data.

	Average Stability Index		
	ROS	**CBOS**	**SMOTE**
LR	14.13	15.39	12.67
PR	22.91	24.49	22.33
POI	45.30	36.61	40.84
LDA	42.92	35.95	31.33
ANN	52.11	46.28	55.22
SVM	64.78	51.47	46.09
RFR	111.63	116.61	111.28
SV	56.54	53.56	49.62

Table 7.4: Metric Average Stability Index STATIC

Table 7.5 shows the average stability of the scorecard in the OOT period based on the second implementation approach. The desired stability index is 0, meaning that the scorecards do not change and hence enhance predictability. All scorecards show a stability index of less than 100. This is good value. All ML scorecards are less stable when compared to the SM scorecard. The dynamic approach has improved the stability index for the RFR scorecards from the highest of 116.61 in the static approach to the highest being 92.87.

	Average Stability		
	ROS	**CBOS**	**SMOTE**
LR	34.89	40.65	36.92
PR	35.98	39.27	37.87
POI	50.63	51.70	44.26
LDA	44.12	42.49	44.18
ANN	79.90	77.41	68.61
SVM	59.98	63.47	49.44
RFR	92.87	69.88	79.90
SV	67.91	77.76	75.40

Table 7.5: Metric Average Stability Index DYNAMIC

Chapter 8

Conclusion and Recommendation

8.1 Conclusion

This study applied eight different modelling techniques for three sample datasets to develop credit scorecards and test the scorecards with unseen OOT data. The sample datasets were drawn from a population of loans which were granted from the period of 2013 to 2017 and the OOT data was drawn from 2018. Out of the eight modelling techniques used, three were ML techniques, four were classical statistical methods, and a survival modelling technique was used. The sampling techniques used comprised of two statistical and one machine learning technique.

For the BVT data and for all three sampling techniques, the scorecard with the most desirable performance is the one built using the RFR modelling technique. The metric values for the RFR scorecards are slightly different for the different samples, and based on only BVT results, the best sampling technique for RFR is SMOTE. The PR and LR scorecards show similar values for the performance metrics and ROS is the sampling technique for these two scorecards. The POI scorecard performs the best when using the ROS sample data, while LDA performs well when using the SMOTE data. The scorecard with the worst BVT performance is the one built with CBOS and the SVM technique.

The OOT results paint a slightly different picture from the BVT results. This study tested two approaches to implementing the scorecards in the OOT period. Based on the static approach, the scorecards that show a drop in performance heavily and are unstable are less desirable. This part of the study found that the LR and PR scorecard were the most stable

with the smallest drop in performance except in January, where almost all of the scorecards experience a drop in performance. This means that customers financially behave differently in January than in all other months and this may be due many factors, one of which might be their financial behaviour in the festive season. The RFR scorecard performed the worst in the OOT period, as it was the least stable and with the highest average change, even though it gave the best performance in the BVT. The drop in performance and the higher instability was seen in all the machine learning techniques.

The results from the dynamic approach (where the scorecards are updated as new data is introduced) are different from the static approach (where the scorecards are not updated with new data). The results show that the ML scorecards all improved their respective measures used in the OOT period. This is attributed to the fact that ML algorithms are supposed to "learn" from the data, as shown by the literature review. The SVM scorecard is the most improved, while all the SM scorecards have deteriorated. The ML scorecards are observed to be more unstable when compared to the SM scorecards. This is because as the scorecards improve, and they look more and more different from the BVT, which contributes to the high stability index.

8.2 Recommendation

This study shows that all aspects of scorecard development from how the data is sampled to the implementation are important to the overall output. Implementing ML scorecards using the static approach can yield results that are significantly different from the BVT stage. This is because the static approach does not allow the model to "learn" from the data. Based on the dynamic approach to implementing the scorecard, it is clear that the ML scorecards provide the best performance, as the scorecards are shown to improve in the OOT period. Based on the static approach, the statistical techniques should not be written off based only on BVT performance. As the OOT performance has shown, the ML techniques are not relatively stable when compared to the classical statistical techniques.

More performance measures that are more suited to the different techniques may need to be used in the future. The stability index used in this research was developed to primarily measure the change between time periods and seems to penalise the scorecard even if the scorecard improves. A better measure would also take into account stability, even in a gradient.

This type of study is also open to other measures of performance, like the F-score, misclassification rate, precision, and many more.

Appendix A

Appendix

A.1 Variable List

Varable	Description
acc_now_delinq	The number of accounts on which the borrower is now delinquent.
acc_open_past_24mths	Number of trades opened in past 24 months.
addr_state	The state provided by the borrower in the loan application
all_util	Balance to credit limit on all trades
annual_inc	The self-reported annual income provided by the borrower during registration.
annual_inc_joint	The combined self-reported annual income provided by the co-borrowers during registration
application_type	Indicates whether the loan is an individual application or a joint application with two co-borrowers
avg_cur_bal	Average current balance of all accounts

bc_open_to_buy	Total open to buy on revolving bankcards.
bc_util	Ratio of total current balance to high credit/credit limit for all bankcard accounts.
chargeoff_within_12_mths	Number of charge-offs within 12 months
collection_recovery_fee	post charge off collection fee
collections_12_mths_ex_med	Number of collections in 12 months excluding medical collections
delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years
delinq_amnt	The past-due amount owed for the accounts on which the borrower is now delinquent.
desc	Loan description provided by the borrower
dti	A ratio calculated using the borrower's total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower's self-reported monthly income.
dti_joint	A ratio calculated using the co-borrowers' total monthly payments on the total debt obligations, excluding mortgages and the requested LC loan, divided by the co-borrowers' combined self-reported monthly income
earliest_cr_line	The month the borrower's earliest reported credit line was opened
emp_length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.

emp_title	The job title supplied by the Borrower when applying for the loan.*
fico_range_high	The upper boundary range the borrower's FICO at loan origination belongs to.
fico_range_low	The lower boundary range the borrower's FICO at loan origination belongs to.
funded_amnt	The total amount committed to that loan at that point in time.
funded_amnt_inv	The total amount committed by investors for that loan at that point in time.
grade	LC assigned loan grade
home_ownership	The home ownership status provided by the borrower during registration or obtained from the credit report. Our values are: RENT, OWN, MORTGAGE, OTHER
il_util	Ratio of total current balance to high credit/credit limit on all instalment accounts
initial_list_status	The initial listing status of the loan. Possible values are – W, F
inq_fi	Number of personal finance inquiries
inq_last_12m	Number of credit inquiries in past 12 months
inq_last_6mths	The number of inquiries in past 6 months (excluding auto and mortgage inquiries)
instalment	The monthly payment owed by the borrower if the loan originates.
int_rate	Interest Rate on the loan
issue_d	The month which the loan was funded
last_credit_pull_d	The most recent month LC pulled credit for this loan
last_fico_range_high	The upper boundary range the borrower's last FICO pulled belongs to.

last_fico_range_low	The lower boundary range the borrower's last FICO pulled belongs to.
last_pymnt_amnt	Last total payment amount received
last_pymnt_d	Last month payment was received
loan_amnt	The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value.
loan_status	Current status of the loan
max_bal_bc	Maximum current balance owed on all revolving accounts
member_id	A unique LC assigned Id for the borrower member.
mo_sin_old_il_acct	Months since oldest bank instalment account opened
mo_sin_old_rev_tl_op	Months since oldest revolving account opened
mo_sin_rcnt_rev_tl_op	Months since most recent revolving account opened
mo_sin_rcnt_tl	Months since most recent account opened
mort_acc	Number of mortgage accounts.
mths_since_last_delinq	The number of months since the borrower's last delinquency.
mths_since_last_major_derog	Months since most recent 90-day or worse rating
mths_since_last_record	The number of months since the last public record.
mths_since_rcnt_il	Months since most recent instalment accounts opened
mths_since_recent_bc	Months since most recent bankcard account opened.
mths_since_recent_bc_dlq	Months since most recent bankcard delinquency
mths_since_recent_inq	Months since most recent inquiry.
mths_since_recent_revol_delinq	Months since most recent revolving delinquency.
next_pymnt_d	Next scheduled payment date

num_accts_ever_120_pd	Number of accounts that were over 120 or more days past due
num_actv_bc_tl	Number of currently active bankcard accounts
num_actv_rev_tl	Number of currently active revolving trades
num_bc_sats	Number of satisfactory bankcard accounts
num_bc_tl	Number of bankcard accounts
num_il_tl	Number of instalment accounts
num_op_rev_tl	Number of open revolving accounts
num_rev_accts	Number of revolving accounts
num_rev_tl_bal_gt_0	Number of revolving trades with balance >0
num_sats	Number of satisfactory accounts
num_tl_120dpd_2m	Number of accounts currently 120 days past due (updated in past 2 months)
num_tl_30dpd	Number of accounts currently 30 days past due (updated in past 2 months)
num_tl_90g_dpd_24m	Number of accounts 90 or more days past due in last 24 months
num_tl_op_past_12m	Number of accounts opened in past 12 months
open_acc	The number of open credit lines in the borrower's credit file.
open_acc_6m	Number of open trades in last 6 months
open_il_12m	Number of instalment accounts opened in past 12 months
open_il_24m	Number of instalment accounts opened in past 24 months
open_act_il	Number of currently active instalment trades
open_rv_12m	Number of revolving trades opened in past 12 months
open_rv_24m	Number of revolving trades opened in past 24 months

out_prncp	Remaining outstanding principal for total amount funded
out_prncp_inv	Remaining outstanding principal for portion of total amount funded by investors
pct_tl_nvr_dlq	Percent of trades never delinquent
percent_bc_gt_75	Percentage of all bankcard accounts >75% of limit.
policy_code	publicly available policy_code=1, new products not publicly available policy_code=2
pub_rec	Number of derogatory public records
pub_rec_bankruptcies	Number of public record bankruptcies
purpose	A category provided by the borrower for the loan request.
pymnt_plan	Indicates if a payment plan has been put in place for the loan
recoveries	post charge off gross recovery
revol_bal	Total credit revolving balance
revol_util	Revolving line utilisation rate, or the amount of credit the borrower is using relative to all available revolving credit.
sub_grade	LC assigned loan subgrade
tax_liens	Number of tax liens
term	The number of payments on the loan. Values are in months and can be either 36 or 60.
title	The loan title provided by the borrower
tot_coll_amt	Total collection amounts ever owed
tot_cur_bal	Total current balance of all accounts
tot_hi_cred_lim	Total high credit/credit limit
total_acc	The total number of credit lines currently in the borrower's credit file
total_bal_ex_mort	Total credit balance excluding mortgage
total_bal_il	Total current balance of all instalment accounts

total_bc_limit	Total bankcard high credit/credit limit
total_cu_tl	Number of finance trades
total_il_high_credit_limit	Total instalment high credit/credit limit
total_pymnt	Payments received to date for total amount funded
total_pymnt_inv	Payments received to date for portion of total amount funded by investors
total_rec_int	Interest received to date
total_rec_late_fee	Late fees received to date
total_rec_prncp	Principal received to date
total_rev_hi_lim	Total revolving high credit/credit limit
url	URL for the LC page with listing data.
verification_status	Indicates if income was verified by LC, not verified, or if the income source was verified
verified_status_joint	Indicates if the co-borrowers' joint income was verified by LC, not verified, or if the income source was verified
zip_code	The first 3 numbers of the zip code provided by the borrower in the loan application.
revol_bal_joint	Sum of revolving credit balance of the co-borrowers, net of duplicate balances
sec_app_fico_range_low	FICO range (high) for the secondary applicant
sec_app_fico_range_high	FICO range (low) for the secondary applicant
sec_app_earliest_cr_line	Earliest credit line at time of application for the secondary applicant
sec_app_inq_last_6mths	Credit inquiries in the last 6 months at time of application for the secondary applicant

sec_app_mort_acc	Number of mortgage accounts at time of application for the secondary applicant
sec_app_open_acc	Number of open trades at time of application for the secondary applicant
sec_app_revol_util	Ratio of total current balance to high credit/credit limit for all revolving accounts
sec_app_open_act_il	Number of currently active instalment trades at time of application for the secondary applicant
sec_app_num_rev_accts	Number of revolving accounts at time of application for the secondary applicant
sec_app_chargeoff_within_12_mths	Number of charge-offs within last 12 months at time of application for the secondary applicant
sec_app_collections_12_mths_ex_med	Number of collections within last 12 months excluding medical collections at time of application for the secondary applicant
sec_app_mths_since_last_major_derog	Months since most recent 90-day or worse rating at time of application for the secondary applicant
hardship_flag	Flags whether or not the borrower is on a hardship plan
hardship_type	Describes the hardship plan offering
hardship_reason	Describes the reason the hardship plan was offered
hardship_status	Describes if the hardship plan is active, pending, cancelled, completed, or broken
deferral_term	Amount of months that the borrower is expected to pay less than the contractual monthly payment amount due to a hardship plan

hardship_amount	The interest payment that the borrower has committed to make each month while they are on a hardship plan
hardship_start_date	The start date of the hardship plan period
hardship_end_date	The end date of the hardship plan period
payment_plan_start_date	The day the first hardship plan payment is due. For example, if a borrower has a hardship plan period of 3 months, the start date is the start of the three-month period in which the borrower is allowed to make interest-only payments.
hardship_length	The number of months the borrower will make smaller payments than normally obligated due to a hardship plan
hardship_dpd	Account days past due as of the hardship plan start date
hardship_loan_status	Loan Status as of the hardship plan start date
orig_projected_additional_accrued_interest	The original projected additional interest amount that will accrue for the given hardship payment plan as of the Hardship Start Date. This field will be null if the borrower has broken their hardship payment plan.
hardship_payoff_balance_amount	The payoff balance amount as of the hardship plan start date
hardship_last_payment_amount	The last payment amount as of the hardship plan start date
disbursement_method	The method by which the borrower receives their loan. Possible values are: CASH, DIRECT_PAY
debt_settlement_flag	Flags whether or not the borrower, who has charged-off, is working with a debt-settlement company.
debt_settlement_flag_date	The most recent date that the Debt_Settlement_Flag has been set

settlement_status	The status of the borrower's settlement plan. Possible values are: COMPLETE, ACTIVE, BROKEN, CANCELLED, DENIED, DRAFT
settlement_date	The date that the borrower agrees to the settlement plan
settlement_amount	The loan amount that the borrower has agreed to settle for
settlement_percentage	The settlement amount as a percentage of the payoff balance amount on the loan
settlement_term	The number of months that the borrower will be on the settlement plan

Table A.1: All Variables

A.2 Information Value List

Variable	IV
loan_status	20.84441
total_rec_prncp	4.79205
recoveries	1.73125
collection_recovery_fee	1.61360
debt_settlement_flag	1.29107
settlement_status	1.16618
last_credit_pull_d	1.03485
settlement_term	0.95635
settlement_percentage	0.85785
debt_settlement_flag_date	0.84633
settlement_date	0.80742
instalment	0.76403
last_pymnt_d	0.62061
revol_bal	0.53106
last_pymnt_amnt	0.52132
avg_cur_bal	0.51822
emp_title	0.51129
bc_open_to_buy	0.50532

total_bal_il	0.43215
int_rate	0.41648
sub_grade	0.40143
total_il_high_credit_limit	0.38190
grade	0.36269
max_bal_bc	0.36061
dti	0.34549
annual_inc	0.32935
total_bal_ex_mort	0.30890
total_rec_int	0.25405
total_rec_late_fee	0.25310
tot_hi_cred_lim	0.23620
total_rev_hi_lim	0.21695
settlement_amount	0.21444
next_pymnt_d	0.19755
total_pymnt	0.19101
tot_cur_bal	0.18971
out_prncp_inv	0.18852
out_prncp	0.18615
total_pymnt_inv	0.18010
desc	0.17964
total_bc_limit	0.17826
title	0.16354
tot_coll_amt	0.16320
funded_amnt	0.14099
loan_amnt	0.14099
funded_amnt_inv	0.13779
bc_util	0.09989
acc_open_past_24mths	0.09695
zip_code	0.09363
revol_util	0.08836
mo_sin_old_rev_tl_op	0.08335
num_tl_op_past_12m	0.07940
earliest_cr_line	0.06692
mths_since_recent_bc	0.05798
mort_acc	0.05221
mo_sin_rcnt_tl	0.05135
mo_sin_rcnt_rev_tl_op	0.05041
mths_since_recent_inq	0.04577
mo_sin_old_il_acct	0.04395

hardship_status	0.04318
verification_status	0.04116
home_ownership	0.03923
inq_last_6mths	0.03920
hardship_dpd	0.03403
mths_since_rcnt_il	0.03401
hardship_payoff_balance_amount	0.03399
hardship_reason	0.03366
hardship_amount	0.03365
hardship_end_date	0.03348
payment_plan_start_date	0.03327
hardship_last_payment_amount	0.03316
open_rv_24m	0.03275
hardship_start_date	0.03233
hardship_loan_status	0.03227
pct_tl_nvr_dlq	0.03049
il_util	0.02970
percent_bc_gt_75	0.02877
hardship_length	0.02831
hardship_type	0.02831
deferral_term	0.02831
all_util	0.02495
orig_projected_additional_accrued_interest	0.02428
dti_joint	0.02309
inq_last_12m	0.02259
open_rv_12m	0.02127
num_actv_rev_tl	0.02007
emp_length	0.02001
open_acc_6m	0.01968
num_rev_tl_bal_gt_0	0.01962
purpose	0.01950
addr_state	0.01874
open_il_12m	0.01820
open_il_24m	0.01716
mths_since_last_record	0.01696
inq_fi	0.01587
delinq_amnt	0.01446
annual_inc_joint	0.01392
issue_d	0.01184
mths_since_last_major_derog	0.01107

total_acc	0.01131
initial_list_status	0.00978
mths_since_recent_bc_dlq	0.00931
mths_since_recent_revol_delinq	0.00842
mths_since_last_delinq	0.00817
pub_rec	0.00803
num_il_tl	0.00730
open_acc	0.00729
num_bc_tl	0.00716
num_rev_accts	0.00701
num_actv_bc_tl	0.00608
num_op_rev_tl	0.00674
num_sats	0.00637
pub_rec_bankruptcies	0.00589
num_accts_ever_120_pd	0.00524
year	0.00502
open_act_il	0.00465
num_bc_sats	0.00410
num_tl_120dpd_2m	0.00302
delinq_2yrs	0.00287
total_cu_tl	0.00265
hardship_flag	0.00262
pymnt_plan	0.00262
num_tl_90g_dpd_24m	0.00226
disbursement_method	0.00222
tax_liens	0.00175
collections_12_mths_ex_med	0.00141
acc_now_delinq	0.00043
num_tl_30dpd	0.00029
chargeoff_within_12_mths	0.00029
application_type	0.00002
verification_status_joint	0.00001
policy_code	0.00000
sec_app_chargeoff_within_12_mths	0.00000
revol_bal_joint	0.00000
id	0.00000
sec_app_mort_acc	0.00000
sec_app_earliest_cr_line	0.00000
url	0.00000
sec_app_collections_12_mths_ex_med	0.00000

sec_app_revol_util	0.00000
sec_app_open_act_il	0.00000
sec_app_num_rev_accts	0.00000
sec_app_mths_since_last_major_derog	0.00000
sec_app_inq_last_6mths	0.00000
sec_app_open_acc	0.00000
member_id	0.00000
term	0.00000

Table A.2: Full IV list

A.3 Variable clusters

Cluster	**Variable**	**RS_Own**	**RS_NC**	**RS_Ratio**
1	mort_acc_woe	0.59170	0.15903	0.48551
1	tot_cur_bal_woe	0.87699	0.13704	0.14254
1	avg_cur_bal_woe	0.81185	0.10576	0.21040
1	home_ownership_woe	0.59140	0.06920	0.43898
1	tot_hi_cred_lim_woe	0.84105	0.14129	0.18511
2	acc_open_past_24mths_woe	0.63677	0.20692	0.45800
2	mo_sin_rcnt_rev_tl_op_woe	0.69729	0.16476	0.36242
2	mths_since_recent_bc_woe	0.49683	0.10643	0.56310
2	num_tl_op_past_12m_woe	0.63391	0.25680	0.49259
2	mo_sin_rcnt_tl_woe	0.64698	0.14597	0.41335
3	mths_since_rcnt_il_woe	0.86180	0.18784	0.17017
3	open_il_12m_woe	0.88295	0.19474	0.14536
3	open_il_24m_woe	0.77047	0.21291	0.29161
4	total_rec_int_woe	0.62815	0.03684	0.38607
4	total_pymnt_inv_woe	0.96248	0.13927	0.04360
4	total_pymnt_woe	0.96248	0.13933	0.04359
4	total_rec_prncp_woe	0.88034	0.13339	0.13808
5	int_rate_woe	0.95111	0.11380	0.05517
5	sub_grade_woe	0.93997	0.11481	0.06781
5	purpose_woe	0.11485	0.01436	0.89804
5	grade_woe	0.94227	0.11107	0.06495
6	tot_coll_amt_woe	0.02793	0.01591	0.98778
6	revol_util_woe	0.77571	0.04598	0.23511
6	percent_bc_gt_75_woe	0.79690	0.06918	0.21819

6	bc_util_woe	0.86390	0.07036	0.14640
6	installment_woe	0.00561	0.18301	1.21713
7	funded_amnt_woe	0.98375	0.05214	0.01715
7	loan_amnt_woe	0.98375	0.05214	0.01715
7	funded_amnt_inv_woe	0.93133	0.04998	0.07228
8	pct_tl_nvr_dlq_woe	0.07005	0.00841	0.93783
8	total_bc_limit_woe	0.88304	0.12565	0.13377
8	total_rev_hi_lim_woe	0.86929	0.17627	0.15868
8	bc_open_to_buy_woe	0.56763	0.34985	0.66502
8	revol_bal_woe	0.59008	0.17646	0.49776
8	max_bal_bc_woe	0.24078	0.05147	0.80042
9	out_prncp_woe	0.95589	0.03570	0.04575
9	out_prncp_inv_woe	0.95588	0.03571	0.04575
9	last_pymnt_amnt_woe	0.21476	0.07908	0.85268
10	num_actv_rev_tl_woe	0.99254	0.12248	0.00850
10	num_rev_tl_bal_gt_0_woe	0.99254	0.12526	0.00853
11	total_bal_ex_mort_woe	0.84196	0.10892	0.17736
11	total_il_high_credit_limit_woe	0.84196	0.05216	0.16674
12	mths_since_last_major_derog_woe	0.05744	0.04533	0.98732
12	mths_since_recent_inq_woe	0.78874	0.15305	0.24944
12	inq_last_6mths_woe	0.78598	0.11013	0.24051
13	total_acc_woe	0.56590	0.07298	0.46828
13	mo_sin_old_il_acct_woe	0.55933	0.05599	0.46680
13	mo_sin_old_rev_tl_op_woe	0.50129	0.09227	0.54941
14	open_rv_24m_woe	0.74024	0.20901	0.32839
14	open_rv_12m_woe	0.84163	0.21817	0.20256
14	open_acc_6m_woe	0.61672	0.18608	0.47091
15	il_util_woe	0.66251	0.21329	0.42899
15	total_bal_il_woe	0.47738	0.14411	0.61061
15	all_util_woe	0.66041	0.11387	0.38323
16	inq_fi_woe	0.75132	0.12352	0.28372
16	inq_last_12m_woe	0.75132	0.17771	0.30242
17	annual_inc_woe	0.62231	0.19581	0.46965
17	emp_length_woe	0.62231	0.01597	0.38382
18	total_rec_late_fee_woe	1.00000	0.00873	-
19	dti_woe	0.54106	0.04838	0.48228
19	verification_status_woe	0.54106	0.05269	0.48447
20	mths_since_last_record_woe	1.00000	0.03810	-

Table A.3: Cluster Full details

A.4 OOT

A.4.1 Static approach

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.69264	0.68198	0.69102	0.61823	0.60293	0.59005	0.58132	0.57040	0.56556	0.55810	0.55098	0.54782	0.54109	0.53660	0.53253
PR	0.69260	0.64576	0.68284	0.60885	0.62847	0.63598	0.61675	0.60593	0.60174	0.61131	0.63877	0.62940	0.62472	0.61157	0.61558
POI	0.65720	0.68803	0.65981	0.57083	0.51378	0.59551	0.55278	0.54439	0.54412	0.56031	0.55725	0.52220	0.53594	0.51547	0.55550
LDA	0.68741	0.68858	0.68758	0.58547	0.58175	0.55152	0.56816	0.56935	0.56449	0.61128	0.56039	0.55024	0.59606	0.56109	0.61119
ANN	0.69266	0.74193	0.69525	0.62608	0.54935	0.54784	0.54153	0.61319	0.60025	0.57315	0.55637	0.61148	0.57247	0.55868	0.60691
SVM	0.61698	0.68815	0.62107	0.56309	0.50617	0.55512	0.54701	0.49760	0.57532	0.51601	0.57439	0.52463	0.54593	0.52691	0.49999
RFR	0.73454	0.73867	0.73508	0.54526	0.39320	0.52595	0.37958	0.52775	0.46411	0.48836	0.38084	0.38245	0.51383	0.42675	0.62312
SV	0.69084	0.65841	0.68451	0.53977	0.60775	0.56954	0.60637	0.60679	0.56360	0.50507	0.51508	0.54760	0.53074	0.55449	0.48062

Table A.4: ROS GINI OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.36037	0.34074	0.35704	0.31691	0.30900	0.29473	0.28822	0.28212	0.27879	0.27598	0.27406	0.27277	0.27126	0.27063	0.27022
PR	0.36008	0.34047	0.35675	0.31846	0.30864	0.29950	0.29887	0.28948	0.29307	0.28691	0.29354	0.28579	0.29502	0.28977	0.28073
POI	0.33363	0.32133	0.33146	0.25339	0.27445	0.22502	0.22529	0.19629	0.21706	0.19751	0.20477	0.18540	0.21370	0.19632	0.21627
LDA	0.35835	0.34519	0.35620	0.28168	0.29746	0.26935	0.27245	0.25653	0.25667	0.25199	0.23208	0.23561	0.24328	0.23082	0.24281
ANN	0.36062	0.36839	0.36151	0.24111	0.24237	0.22380	0.23492	0.21932	0.21236	0.22409	0.21660	0.19302	0.18673	0.20317	0.22206
SVM	0.31715	0.33976	0.31867	0.25145	0.25112	0.22306	0.17247	0.20755	0.19452	0.17517	0.20244	0.20490	0.20948	0.20365	0.18861
RFR	0.34820	0.35822	0.34970	0.28453	0.19659	0.23527	0.22462	0.17586	0.20601	0.16615	0.15483	0.14253	0.18262	0.11701	0.11647
SV	0.35938	0.37618	0.36133	0.28047	0.28600	0.24498	0.25270	0.20194	0.23596	0.21315	0.19852	0.21537	0.21761	0.24408	0.23391

Table A.5: ROS KS OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.84632	0.84099	0.84551	0.80912	0.80146	0.79503	0.79066	0.78520	0.78278	0.77905	0.77549	0.77391	0.77054	0.76830	0.76626
PR	0.84630	0.82288	0.84142	0.80443	0.81424	0.81799	0.80838	0.80296	0.80087	0.80566	0.81938	0.81470	0.81236	0.80579	0.80779
POI	0.82860	0.84401	0.82990	0.78542	0.75689	0.79775	0.77639	0.77219	0.77206	0.78015	0.77863	0.76110	0.76797	0.75773	0.77775
LDA	0.84370	0.84429	0.84379	0.79273	0.79088	0.77576	0.78408	0.78468	0.78224	0.80564	0.78019	0.77512	0.79803	0.78055	0.80560
ANN	0.84633	0.87096	0.84763	0.81304	0.77467	0.77392	0.77076	0.80659	0.80013	0.78658	0.77819	0.80574	0.78624	0.77934	0.80345
SVM	0.80849	0.84407	0.81053	0.78154	0.75309	0.77756	0.77350	0.74880	0.78766	0.75800	0.78720	0.76232	0.77296	0.76346	0.74999
RFR	0.86727	0.86934	0.86754	0.77263	0.69660	0.76297	0.68979	0.76388	0.73205	0.74418	0.69042	0.69123	0.75692	0.71338	0.81156
SV	0.84542	0.82921	0.84226	0.76989	0.80387	0.78477	0.80318	0.80339	0.78180	0.75254	0.75754	0.77380	0.76537	0.77725	0.74031

Table A.6: ROS AUC OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.81233	0.85242	0.81506	0.74163	0.72012	0.71455	0.71062	0.71023	0.70922	0.71035	0.70903	0.71027	0.70942	0.71091	0.71200
PR	0.81233	0.85242	0.81506	0.76214	0.72880	0.72792	0.73948	0.75848	0.74763	0.73204	0.73866	0.75164	0.75420	0.74829	0.74132
POI	0.73791	0.71945	0.73478	0.60533	0.60839	0.57024	0.62505	0.57737	0.61917	0.64083	0.62087	0.67551	0.63323	0.61089	0.64167
LDA	0.81066	0.74204	0.79300	0.69570	0.66370	0.62283	0.66100	0.62377	0.60913	0.64979	0.64639	0.69030	0.70569	0.68104	0.63079
ANN	0.81181	0.89593	0.81401	0.74216	0.62902	0.63100	0.63863	0.63600	0.62295	0.53126	0.59314	0.54818	0.58535	0.49740	0.52010
SVM	0.80866	0.78451	0.80508	0.57482	0.53229	0.62476	0.55219	0.48832	0.61847	0.62684	0.60701	0.52613	0.49427	0.59194	0.50156
RFR	0.96345	0.98516	0.97055	0.64541	0.46766	0.59492	0.47274	0.58544	0.58138	0.37583	0.34758	0.43828	0.52212	0.44487	0.53130
SV	0.80971	0.76600	0.80173	0.64280	0.59422	0.61872	0.57228	0.53419	0.65236	0.56600	0.62872	0.67542	0.59480	0.52765	0.63446

Table A.7: ROS ACCURACY OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.82013	0.80687	0.81845	0.73063	0.72968	0.73070	0.72926	0.72793	0.72870	0.73088	0.73023	0.72963	0.72805	0.73012	0.72943
PR	0.82015	0.80689	0.81847	0.74055	0.73297	0.74632	0.72934	0.76551	0.72966	0.74960	0.76322	0.74925	0.74820	0.72838	0.75389
POI	0.86439	0.94693	0.86993	0.76900	0.76934	0.77922	0.72798	0.73264	0.75601	0.68772	0.71973	0.71013	0.81418	0.72279	0.68636
LDA	0.82088	0.74955	0.80525	0.68878	0.63396	0.69435	0.66813	0.66160	0.61508	0.69914	0.62360	0.67039	0.64610	0.62125	0.69438
ANN	0.82111	0.82588	0.82181	0.64114	0.71766	0.62695	0.62095	0.73090	0.65147	0.62942	0.62258	0.72838	0.70739	0.65943	0.71187
SVM	0.81999	0.87529	0.82231	0.75120	0.68664	0.65929	0.72403	0.65904	0.73451	0.64433	0.74377	0.65337	0.73257	0.66414	0.67277
RFR	0.96320	0.88701	0.95006	0.58137	0.57052	0.68720	0.75814	0.63382	0.57003	0.52701	0.51879	0.62802	0.80285	0.81219	0.49993
SV	0.81305	0.81043	0.81259	0.58032	0.70816	0.65211	0.65369	0.66521	0.64837	0.65437	0.62179	0.62446	0.64689	0.69579	0.64982

Table A.8: ROS SPECIFICITY OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.6911903	0.64988	0.68368	0.60725	0.60437	0.60590	0.60658	0.60657	0.60433	0.60738	0.60642	0.60438	0.60737	0.60692	0.60523
PR	0.6911863	0.65351	0.68438	0.63913	0.62499	0.61530	0.63693	0.60499	0.61938	0.64100	0.61710	0.62948	0.62411	0.64114	0.62732
POI	0.6526124	0.69286	0.65484	0.58707	0.55067	0.54549	0.60520	0.59644	0.52736	0.54695	0.56338	0.58303	0.58617	0.56461	0.58207
LDA	0.6879575	0.68627	0.68771	0.55095	0.53412	0.54703	0.60093	0.54286	0.55235	0.58563	0.58315	0.57076	0.58941	0.54633	0.54456
ANN	0.6916793	0.69894	0.69236	0.58805	0.56651	0.53200	0.54101	0.56044	0.51222	0.50950	0.54005	0.57948	0.53087	0.57921	0.58422
SVM	0.6141589	0.58859	0.60933	0.60674	0.56079	0.53514	0.55626	0.53084	0.51678	0.56416	0.57263	0.51666	0.54222	0.58330	0.58592
RFR	0.7041473	0.76807	0.71006	0.38063	0.54728	0.35872	0.60135	0.41143	0.57441	0.55620	0.50045	0.53938	0.34596	0.35413	0.35646
SV	0.6884535	0.67556	0.68663	0.55903	0.51799	0.56602	0.51802	0.47350	0.57064	0.55957	0.50564	0.60344	0.56561	0.50929	0.60271

Table A.9: CBOS GINI OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.35495	0.36590	0.35627	0.32216	0.30872	0.29534	0.28695	0.27967	0.27611	0.27318	0.26980	0.26830	0.26721	0.26605	0.26545
PR	0.35508	0.36604	0.35640	0.32885	0.32419	0.31638	0.31375	0.30977	0.29819	0.29422	0.29516	0.28773	0.27804	0.28886	0.28626
POI	0.32918	0.33902	0.33019	0.28990	0.27136	0.26119	0.22168	0.23971	0.23366	0.22672	0.20071	0.20409	0.21668	0.20038	0.20038
LDA	0.35426	0.36069	0.35486	0.31906	0.28548	0.26089	0.25796	0.24128	0.24722	0.23176	0.23435	0.22056	0.22903	0.22863	0.21664
ANN	0.35547	0.34738	0.35418	0.24418	0.27766	0.21587	0.21512	0.23428	0.22598	0.18570	0.20563	0.22387	0.19536	0.18334	0.17383
SVM	0.31416	0.31910	0.31465	0.26984	0.20691	0.22799	0.17674	0.20503	0.19628	0.16653	0.19928	0.16649	0.19009	0.15673	0.16345
RFR	0.31154	0.34684	0.31163	0.21475	0.14457	0.19920	0.14216	0.11925	0.11580	0.11052	0.14790	0.14760	0.12338	0.15344	0.17013
SV	0.35345	0.34532	0.35180	0.29399	0.29722	0.25424	0.27965	0.23979	0.26690	0.25453	0.25183	0.25947	0.23280	0.23659	0.25295

Table A.10: CBOS KS OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.84560	0.82494	0.84184	0.80363	0.80218	0.80295	0.80329	0.80328	0.80217	0.80369	0.80321	0.80219	0.80369	0.80346	0.80262
PR	0.84559	0.82676	0.84219	0.81957	0.81249	0.80765	0.81846	0.80249	0.80969	0.82050	0.80855	0.81474	0.81205	0.82057	0.81366
POI	0.82631	0.84643	0.82742	0.79354	0.77534	0.77274	0.80260	0.79822	0.76368	0.77348	0.78169	0.79151	0.79308	0.78230	0.79103
LDA	0.84398	0.84313	0.84386	0.77547	0.76706	0.77351	0.80046	0.77143	0.77618	0.79282	0.79157	0.78538	0.79470	0.77317	0.77228
ANN	0.84584	0.84947	0.84618	0.79402	0.78326	0.76600	0.77051	0.78022	0.75611	0.75475	0.77002	0.78974	0.76543	0.78961	0.79211
SVM	0.80708	0.79430	0.80466	0.80337	0.78039	0.76757	0.77813	0.76542	0.75839	0.78208	0.78632	0.75833	0.77111	0.79165	0.79296
RFR	0.85207	0.88404	0.85503	0.69032	0.77364	0.67936	0.80068	0.70571	0.78720	0.77810	0.75023	0.76969	0.67298	0.67706	0.67823
SV	0.84423	0.83778	0.84332	0.77952	0.75900	0.78301	0.75901	0.73675	0.78532	0.77978	0.75282	0.80172	0.78280	0.75464	0.80136

Table A.11: CBOS AUC OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.78542	0.76198	0.78089	0.69738	0.67183	0.66215	0.66090	0.66008	0.66217	0.66165	0.65909	0.66025	0.65864	0.66010	0.66210
PR	0.78543	0.76200	0.78090	0.73470	0.69849	0.71010	0.67749	0.66618	0.67763	0.67411	0.67071	0.67455	0.66500	0.67376	0.69800
POI	0.74015	0.79410	0.74568	0.62140	0.60116	0.59822	0.60740	0.56349	0.57456	0.53755	0.56198	0.53642	0.55327	0.61260	0.59929
LDA	0.78484	0.78063	0.78439	0.65021	0.58062	0.60389	0.63920	0.58297	0.56406	0.57264	0.61143	0.64524	0.59477	0.59988	0.59929
ANN	0.78532	0.75472	0.78039	0.60491	0.61489	0.55286	0.48398	0.45527	0.43845	0.43087	0.46063	0.55968	0.50876	0.51996	0.58763
SVM	0.78200	0.86145	0.78912	0.63655	0.47384	0.51002	0.53736	0.62541	0.60852	0.51049	0.49193	0.45885	0.44577	0.47623	0.57894
RFR	0.97259	0.97988	0.97520	0.48718	0.42247	0.36639	0.43085	0.40128	0.49973	0.58718	0.57779	0.58070	0.50481	0.54273	0.38075
SV	0.78240	0.82155	0.78660	0.68520	0.66016	0.63628	0.62505	0.66806	0.67144	0.63534	0.60882	0.62341	0.61906	0.64296	0.63334

Table A.12: CBOS ACCURACY OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.79484	0.73318	0.78262	0.68141	0.68055	0.67903	0.68203	0.68184	0.68168	0.68017	0.67959	0.68193	0.67880	0.67983	0.68141
PR	0.79489	0.73323	0.78267	0.67855	0.68614	0.71847	0.70512	0.68152	0.69197	0.71116	0.68733	0.71951	0.70938	0.71471	0.69221
POI	0.82766	0.84525	0.82999	0.72231	0.68835	0.70830	0.66365	0.62832	0.68990	0.68636	0.65801	0.67114	0.63572	0.67356	0.70847
LDA	0.79440	0.86750	0.79670	0.74765	0.72186	0.74575	0.71865	0.68318	0.72795	0.71382	0.71540	0.73789	0.65795	0.68783	0.71595
ANN	0.79632	0.83915	0.80065	0.65293	0.68799	0.65412	0.67076	0.66066	0.68210	0.63279	0.66686	0.68823	0.65890	0.72903	0.66515
SVM	0.79056	0.79990	0.79164	0.67806	0.62776	0.60922	0.66926	0.62818	0.69494	0.60354	0.68794	0.70211	0.61267	0.64753	0.61634
RFR	0.97562	0.96683	0.97435	0.62639	0.61368	0.87188	0.74927	0.65074	0.76571	0.70850	0.59719	0.82677	0.83471	0.68917	0.63849
SV	0.78506	0.74898	0.77732	0.66695	0.61770	0.66538	0.67746	0.65757	0.58323	0.54820	0.60634	0.69336	0.64768	0.56053	0.69903

Table A.13: CBOS SPECIFICITY OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.697548	0.72873	0.69990	0.63517	0.63680	0.63516	0.63729	0.63606	0.63758	0.63685	0.63560	0.63716	0.63676	0.63544	0.63470
PR	0.697531	0.68835	0.69645	0.63471	0.65021	0.66062	0.65697	0.67209	0.65884	0.66634	0.63076	0.66632	0.64009	0.64673	0.65737
POI	0.662521	0.65950	0.66210	0.58091	0.56558	0.57655	0.59586	0.56261	0.57043	0.61737	0.53370	0.61980	0.60619	0.58178	0.54797
LDA	0.693854	0.68890	0.69296	0.63549	0.59262	0.63238	0.62731	0.58398	0.57503	0.61802	0.59739	0.57915	0.56225	0.62574	0.62169
ANN	0.697819	0.70152	0.69826	0.58538	0.58617	0.56721	0.62347	0.53787	0.61291	0.55284	0.54909	0.59073	0.58194	0.62074	0.62391
SVM	0.625321	0.69244	0.63163	0.62890	0.57408	0.56673	0.55849	0.57726	0.62938	0.63308	0.58637	0.60400	0.57671	0.57242	0.62043
RFR	0.787333	0.84878	0.79329	0.52931	0.35695	0.48120	0.39738	0.41565	0.57958	0.49424	0.42687	0.46936	0.48906	0.37429	0.42074
SV	0.695308	0.74656	0.70172	0.65582	0.59128	0.61167	0.55588	0.60907	0.63815	0.53286	0.54318	0.60820	0.57977	0.56327	0.59664

Table A.14: SMOTE GINI OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.35973	0.36400	0.36031	0.32393	0.31238	0.29777	0.29089	0.28429	0.27975	0.27581	0.27385	0.27145	0.27027	0.27015	0.26890
PR	0.36019	0.36447	0.36077	0.33085	0.31299	0.31394	0.30793	0.30879	0.30419	0.29150	0.29744	0.28550	0.28095	0.28548	0.29104
POI	0.33481	0.31463	0.33008	0.26366	0.27165	0.22876	0.21224	0.20980	0.19237	0.21288	0.21738	0.22768	0.21061	0.21292	0.20936
LDA	0.35897	0.37289	0.36076	0.31411	0.30070	0.29023	0.26587	0.25777	0.25939	0.25842	0.24370	0.25573	0.25461	0.25381	0.25610
ANN	0.36082	0.34450	0.35727	0.23965	0.26484	0.20561	0.24316	0.23336	0.20641	0.20062	0.21590	0.21841	0.19109	0.22380	0.22483
SVM	0.32538	0.32345	0.32503	0.27962	0.27424	0.25784	0.25433	0.24541	0.24752	0.23962	0.23256	0.23858	0.23259	0.22616	0.22607
RFR	0.31499	0.31959	0.31569	0.20046	0.18795	0.15896	0.12346	0.11363	0.15015	0.17153	0.15502	0.16328	0.17812	0.14674	0.11870
SV	0.35814	0.37702	0.36039	0.32646	0.28501	0.26643	0.24970	0.26918	0.24705	0.26982	0.26451	0.25151	0.25837	0.23723	0.26049

Table A.15: SMOTE KS OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.84877	0.86436	0.84995	0.81758	0.81840	0.81758	0.81865	0.81803	0.81879	0.81843	0.81780	0.81858	0.81838	0.81772	0.81735
PR	0.84877	0.84418	0.84822	0.81736	0.82510	0.83031	0.82848	0.83604	0.82942	0.83317	0.81538	0.83316	0.82004	0.82336	0.82869
POI	0.83126	0.82975	0.83105	0.79045	0.78279	0.78827	0.79793	0.78130	0.78521	0.80869	0.76685	0.80990	0.80309	0.79089	0.77398
LDA	0.84693	0.84445	0.84648	0.81775	0.79631	0.81619	0.81366	0.79199	0.78752	0.80901	0.79869	0.78957	0.78112	0.81287	0.81085
ANN	0.84891	0.85076	0.84913	0.79269	0.79309	0.78360	0.81174	0.76894	0.80646	0.77642	0.77454	0.79537	0.79097	0.81037	0.81196
SVM	0.81266	0.84622	0.81582	0.81445	0.78704	0.78336	0.77925	0.78863	0.81469	0.81654	0.79319	0.80200	0.78836	0.78621	0.81021
RFR	0.89367	0.92439	0.89665	0.76465	0.67848	0.74060	0.69869	0.70782	0.78979	0.74712	0.71343	0.73468	0.74453	0.68714	0.71037
SV	0.84765	0.87328	0.85086	0.82791	0.79564	0.80583	0.77794	0.80454	0.81908	0.76643	0.77159	0.80410	0.78988	0.78164	0.79832

Table A.16: SMOTE AUC OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.76561	0.78910	0.76797	0.69427	0.67206	0.66687	0.66450	0.66381	0.66434	0.66571	0.66404	0.66374	0.66392	0.66552	0.66368
PR	0.76557	0.78907	0.76794	0.70769	0.67944	0.66562	0.70005	0.68996	0.67012	0.66542	0.68613	0.70606	0.70559	0.69121	0.69895
POI	0.75640	0.75832	0.75673	0.59571	0.63561	0.55040	0.61254	0.62858	0.55739	0.58195	0.56004	0.62205	0.62752	0.64197	0.64143
LDA	0.76419	0.74495	0.76082	0.67799	0.64216	0.58130	0.63290	0.63833	0.58880	0.63592	0.62750	0.58968	0.57675	0.57510	0.59006
ANN	0.76533	0.77994	0.76775	0.64051	0.50392	0.47575	0.59394	0.48383	0.44472	0.53783	0.48794	0.50799	0.58743	0.47384	0.53813
SVM	0.75981	0.70439	0.74924	0.53886	0.61790	0.56063	0.49134	0.54315	0.46249	0.57948	0.61390	0.63786	0.58439	0.52730	0.48725
RFR	0.96565	0.94370	0.96114	0.61801	0.43914	0.49042	0.34311	0.51415	0.35815	0.51381	0.42963	0.40389	0.48430	0.45114	0.49709
SV	0.75968	0.76472	0.76055	0.68225	0.60110	0.53793	0.63627	0.56850	0.59344	0.60363	0.65362	0.64374	0.60358	0.65826	0.57352

Table A.17: SMOTE ACCURACY OOT Static

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.77618	0.74055	0.76882	0.67898	0.67883	0.68159	0.68061	0.68202	0.68128	0.68015	0.67916	0.68206	0.67956	0.68129	0.68196
PR	0.77623	0.74059	0.76886	0.70670	0.70049	0.71822	0.69583	0.71955	0.67540	0.70873	0.71025	0.71172	0.68430	0.68648	0.71544
POI	0.79010	0.76801	0.78627	0.64410	0.62120	0.60554	0.69380	0.67815	0.63637	0.66019	0.67450	0.68808	0.64277	0.59641	0.61948
LDA	0.77356	0.75269	0.76945	0.63143	0.64637	0.65430	0.63842	0.63660	0.63423	0.61665	0.65591	0.63083	0.67498	0.63697	0.65503
ANN	0.77804	0.79440	0.78028	0.67133	0.59238	0.63398	0.70479	0.67378	0.63233	0.62276	0.70415	0.68466	0.65452	0.66507	0.70600
SVM	0.76553	0.81317	0.77216	0.68042	0.62750	0.64909	0.62103	0.70401	0.69999	0.68602	0.69715	0.63738	0.67623	0.69782	0.63146
RFR	0.96675	0.91903	0.95634	0.68093	0.74859	0.79605	0.84086	0.57724	0.77044	0.81136	0.57582	0.80288	0.70758	0.76247	0.83669
SV	0.76134	0.80818	0.76408	0.62094	0.62880	0.58165	0.65554	0.69479	0.63905	0.59174	0.59982	0.64308	0.64258	0.66492	0.67817

Table A.18: SMOTE SPECIFICITY OOT Static

A.4.2 Second approach

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.69264	0.68198	0.69102	0.61823	0.66853	0.57813	0.56743	0.58720	0.59060	0.58303	0.57895	0.59737	0.62286	0.65804	0.67583
PR	0.69260	0.64576	0.68284	0.60885	0.60961	0.55267	0.57212	0.60622	0.59726	0.56648	0.62245	0.58691	0.59438	0.57431	0.56985
POI	0.65720	0.68803	0.65981	0.57083	0.54632	0.52787	0.48589	0.52122	0.52699	0.52432	0.56309	0.49501	0.49422	0.54779	0.55928
LDA	0.68741	0.68858	0.68758	0.58547	0.57469	0.52917	0.50659	0.52689	0.55685	0.53775	0.51651	0.55656	0.54903	0.63481	0.64838
ANN	0.69266	0.74193	0.69525	0.62608	0.66189	0.55991	0.69052	0.60924	0.66763	0.62756	0.71766	0.73288	0.78294	0.72995	0.77883
SVM	0.61698	0.68815	0.62107	0.56309	0.66865	0.69614	0.68193	0.67626	0.71763	0.69284	0.71227	0.74854	0.78680	0.81030	0.83205
RFR	0.73454	0.73867	0.73508	0.54526	0.75180	0.59332	0.63535	0.56183	0.56527	0.56887	0.59028	0.68448	0.65895	0.74569	0.67977
SV	0.69084	0.65841	0.68451	0.53977	0.53964	0.61192	0.62278	0.53269	0.64840	0.55507	0.56449	0.70042	0.73671	0.64682	0.66663

Table A.19: ROS GINI OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.36037	0.34074	0.35704	0.31691	0.34269	0.30031	0.30100	0.30003	0.30686	0.31508	0.31309	0.31135	0.34579	0.33884	0.32107
PR	0.36008	0.34047	0.35675	0.31846	0.31955	0.30327	0.28892	0.31569	0.30356	0.31246	0.33389	0.34139	0.35978	0.32123	0.36587
POI	0.33363	0.32133	0.33146	0.25339	0.27249	0.24388	0.22143	0.22418	0.22428	0.22390	0.23320	0.21386	0.27321	0.24200	0.27087
LDA	0.35835	0.34519	0.35620	0.28168	0.29072	0.25887	0.26175	0.26514	0.29008	0.25102	0.24954	0.28697	0.29779	0.31009	0.33346
ANN	0.36062	0.36839	0.36151	0.24111	0.36299	0.30558	0.30980	0.37833	0.35971	0.34166	0.39405	0.37652	0.37329	0.38253	0.39040
SVM	0.31715	0.33976	0.31867	0.25145	0.30735	0.30306	0.34842	0.34137	0.33874	0.33498	0.35617	0.37991	0.37947	0.39657	0.40681
RFR	0.34820	0.35822	0.34970	0.28453	0.29544	0.36979	0.27548	0.37215	0.30619	0.38191	0.34846	0.34349	0.34680	0.36855	0.37623
SV	0.35938	0.37618	0.36133	0.28047	0.29437	0.28318	0.35016	0.27748	0.33569	0.27713	0.32118	0.31428	0.33526	0.32400	0.35081

Table A.20: ROS KS OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.84632	0.84099	0.84551	0.80912	0.83427	0.78907	0.78372	0.79360	0.79530	0.79151	0.78948	0.79868	0.81143	0.82902	0.83791
PR	0.84630	0.82288	0.84142	0.80443	0.80480	0.77633	0.78606	0.80311	0.79863	0.78324	0.81122	0.79345	0.79719	0.78716	0.78492
POI	0.82860	0.84401	0.82990	0.78542	0.77316	0.76394	0.74294	0.76061	0.76350	0.76216	0.78155	0.74751	0.74711	0.77390	0.77964
LDA	0.84370	0.84429	0.84379	0.79273	0.78735	0.76458	0.75330	0.76345	0.77842	0.76887	0.75825	0.77828	0.77451	0.81741	0.82419
ANN	0.84633	0.87096	0.84763	0.81304	0.83095	0.77996	0.84526	0.80462	0.83381	0.81378	0.85883	0.86644	0.89147	0.86497	0.88941
SVM	0.80849	0.84407	0.81053	0.78154	0.83433	0.84807	0.84097	0.83813	0.85882	0.84642	0.85613	0.87427	0.89340	0.90515	0.91602
RFR	0.86727	0.86934	0.86754	0.77263	0.87590	0.79666	0.81768	0.78092	0.78263	0.78444	0.79514	0.84224	0.82947	0.87284	0.83988
SV	0.84542	0.82921	0.84226	0.76989	0.76982	0.80596	0.81139	0.76634	0.82420	0.77754	0.78225	0.85021	0.86836	0.82341	0.83331

Table A.21: ROS AUC OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.81233	0.85242	0.81506	0.74163	0.80361	0.69044	0.70376	0.69151	0.66428	0.66025	0.70404	0.75177	0.71332	0.77258	0.74452
PR	0.81233	0.85242	0.81506	0.76214	0.74561	0.68456	0.69192	0.70285	0.67243	0.72120	0.68692	0.65593	0.64658	0.77660	0.76770
POI	0.73791	0.71945	0.73478	0.60533	0.53998	0.51137	0.54286	0.50138	0.50596	0.56792	0.52348	0.49209	0.44830	0.48348	0.55808
LDA	0.81066	0.74204	0.79300	0.69570	0.63719	0.62063	0.60506	0.60519	0.63676	0.60684	0.62145	0.62358	0.56526	0.54730	0.54453
ANN	0.81181	0.89593	0.81401	0.74216	0.62444	0.61207	0.78202	0.82639	0.83888	0.85929	0.77967	0.87420	0.87312	0.92407	0.91644
SVM	0.80866	0.78451	0.80508	0.57482	0.71093	0.79101	0.86584	0.76967	0.78273	0.86718	0.88468	0.92520	0.92589	0.95512	0.98873
RFR	0.96345	0.98516	0.97055	0.64541	0.94722	0.83628	0.73214	0.82432	0.81071	0.90746	0.86815	0.95354	0.93488	0.92542	0.91596
SV	0.80971	0.76600	0.80173	0.64280	0.56410	0.68543	0.69028	0.78932	0.74789	0.83462	0.66495	0.77058	0.85476	0.87182	0.93528

Table A.22: ROS ACCURACY OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.82013	0.80687	0.81845	0.73063	0.79123	0.68298	0.69911	0.68782	0.68524	0.69270	0.72642	0.72681	0.72000	0.68812	0.75121
PR	0.82015	0.80689	0.81847	0.74055	0.74960	0.72242	0.66763	0.73689	0.69944	0.67986	0.68392	0.72393	0.74942	0.76157	0.80109
POI	0.86439	0.94693	0.86993	0.76900	0.71940	0.63462	0.62388	0.61335	0.68972	0.63240	0.57906	0.59912	0.70683	0.58709	0.66433
LDA	0.82088	0.74955	0.80525	0.68878	0.62932	0.62625	0.57361	0.62969	0.62287	0.61177	0.54797	0.54118	0.66752	0.55685	0.65293
ANN	0.82111	0.82588	0.82181	0.64114	0.63967	0.65625	0.71295	0.75177	0.80339	0.72109	0.89547	0.80342	0.84493	0.83513	0.86445
SVM	0.81999	0.87529	0.82231	0.75120	0.79516	0.76153	0.88998	0.87986	0.93387	0.92485	0.96259	0.95302	0.96854	0.89546	0.96585
RFR	0.96320	0.88701	0.95006	0.58137	0.81446	0.87042	0.82166	0.91296	0.95643	0.79604	0.92393	0.93701	0.92849	0.93564	0.96731
SV	0.81305	0.81043	0.81259	0.58032	0.62603	0.62733	0.76799	0.64889	0.65409	0.72410	0.70017	0.67531	0.63462	0.64538	0.64562

Table A.23: ROS SPECIFICITY OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.6911903	0.64988	0.68368	0.60725	0.65558	0.56028	0.56112	0.55025	0.57438	0.58788	0.61513	0.60416	0.57348	0.60920	0.69821
PR	0.6911863	0.65351	0.68438	0.63913	0.60514	0.56685	0.55500	0.56208	0.58383	0.56919	0.54584	0.64636	0.57023	0.58837	0.59922
POI	0.6526124	0.69286	0.65484	0.58707	0.54742	0.48329	0.53423	0.53954	0.48750	0.47855	0.55616	0.59242	0.60304	0.51601	0.60921
LDA	0.6879575	0.68627	0.68771	0.55095	0.55777	0.49770	0.55544	0.54955	0.55921	0.53922	0.55685	0.58611	0.51005	0.53810	0.62430
ANN	0.6916793	0.69894	0.69236	0.58805	0.71580	0.64633	0.63614	0.61265	0.60004	0.61427	0.71240	0.75375	0.70590	0.77818	0.79681
SVM	0.6141589	0.58859	0.60933	0.60674	0.67498	0.57345	0.61528	0.66476	0.67371	0.66654	0.71875	0.74628	0.75476	0.76888	0.79494
RFR	0.7041473	0.76807	0.71006	0.38063	0.62954	0.52022	0.57413	0.62177	0.59922	0.55585	0.58684	0.59231	0.66965	0.69765	0.68849
SV	0.6884535	0.67556	0.68663	0.55903	0.45191	0.59124	0.52883	0.52638	0.52635	0.65547	0.64212	0.53174	0.57809	0.50882	0.61728

Table A.24: CBOS GINI OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.35495	0.36590	0.35627	0.32216	0.34886	0.30284	0.30058	0.30067	0.29040	0.30035	0.28097	0.31310	0.30187	0.30058	0.30118
PR	0.35508	0.36604	0.35640	0.32885	0.31912	0.30224	0.28485	0.30254	0.29731	0.30645	0.27852	0.31505	0.31702	0.30256	0.29653
POI	0.32918	0.33902	0.33019	0.28990	0.23812	0.24809	0.25085	0.25467	0.23957	0.23301	0.26714	0.26590	0.25051	0.27880	0.25601
LDA	0.35426	0.36069	0.35486	0.31906	0.30426	0.28010	0.28451	0.27280	0.27471	0.28433	0.30150	0.27387	0.25463	0.27866	0.28954
ANN	0.35547	0.34738	0.35418	0.24418	0.31515	0.27814	0.31209	0.31479	0.37124	0.30917	0.38082	0.33321	0.37751	0.38517	0.37997
SVM	0.31416	0.31910	0.31465	0.26984	0.28931	0.28600	0.32881	0.31445	0.33148	0.32561	0.33813	0.35637	0.37238	0.37718	0.39145
RFR	0.31154	0.34684	0.31163	0.21475	0.23642	0.30946	0.24046	0.27341	0.31308	0.26453	0.27819	0.28587	0.31932	0.34662	0.31006
SV	0.35345	0.34532	0.35180	0.29399	0.29456	0.30496	0.28939	0.31636	0.34926	0.32840	0.27935	0.36526	0.37601	0.36563	0.30280

Table A.25: CBOS KS OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.84560	0.82494	0.84184	0.80363	0.82779	0.78014	0.78056	0.77512	0.78719	0.79394	0.80756	0.80208	0.78674	0.80460	0.84910
PR	0.84559	0.82676	0.84219	0.81957	0.80257	0.78342	0.77750	0.78104	0.79192	0.78460	0.77292	0.82318	0.78511	0.79419	0.79961
POI	0.82631	0.84643	0.82742	0.79354	0.77371	0.74165	0.76712	0.76977	0.74375	0.73928	0.77808	0.79621	0.80152	0.75801	0.80460
LDA	0.84398	0.84313	0.84386	0.77547	0.77889	0.74885	0.77772	0.77477	0.77961	0.76961	0.77842	0.79306	0.75502	0.76905	0.81215
ANN	0.84584	0.84947	0.84618	0.79402	0.85790	0.82317	0.81807	0.80633	0.80002	0.80714	0.85620	0.87688	0.85295	0.88909	0.89841
SVM	0.80708	0.79430	0.80466	0.80337	0.83749	0.78673	0.80764	0.83238	0.83686	0.83327	0.85938	0.87314	0.87738	0.88444	0.89747
RFR	0.85207	0.88404	0.85503	0.69032	0.81477	0.76011	0.78707	0.81089	0.79961	0.77792	0.79342	0.79616	0.83482	0.84882	0.84425
SV	0.84423	0.83778	0.84332	0.77952	0.72595	0.79562	0.76442	0.76319	0.76318	0.82774	0.82106	0.76587	0.78904	0.75441	0.80864

Table A.26: CBOS AUC OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.78542	0.76198	0.78089	0.69738	0.75308	0.65620	0.64800	0.65122	0.63081	0.61697	0.60892	0.69315	0.69136	0.69355	0.66160
PR	0.78543	0.76200	0.78090	0.73470	0.70258	0.66388	0.67420	0.67196	0.70113	0.65107	0.63292	0.72544	0.65396	0.74475	0.74636
POI	0.74015	0.79410	0.74568	0.62140	0.61476	0.57237	0.53040	0.55647	0.58743	0.65246	0.57584	0.51920	0.48022	0.54385	0.61354
LDA	0.78484	0.78063	0.78439	0.65021	0.63117	0.60727	0.61256	0.55803	0.58621	0.55689	0.53742	0.64266	0.56823	0.57655	0.57025
ANN	0.78532	0.75472	0.78039	0.60491	0.74328	0.78956	0.71233	0.65058	0.70594	0.79965	0.86266	0.86065	0.89221	0.82537	0.85934
SVM	0.78200	0.86145	0.78912	0.63655	0.68643	0.78049	0.85429	0.76780	0.81228	0.90305	0.92337	0.90948	0.94981	0.97974	0.95642
RFR	0.97259	0.97988	0.97520	0.48718	0.81256	0.76707	0.78604	0.82763	0.81564	0.95185	0.91546	0.95646	0.96082	0.91256	0.97533
SV	0.78240	0.82155	0.78660	0.68520	0.59720	0.74966	0.65360	0.67501	0.77824	0.67856	0.77564	0.82450	0.81096	0.94077	0.95673

Table A.27: CBOS ACCURACY OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.79484	0.73318	0.78262	0.68141	0.74733	0.64084	0.63635	0.63509	0.65800	0.65091	0.63478	0.68285	0.70439	0.69486	0.64593
PR	0.79489	0.73323	0.78267	0.67855	0.70182	0.66728	0.66623	0.67279	0.61552	0.63115	0.63125	0.64261	0.73176	0.73933	0.64876
POI	0.82766	0.84525	0.82999	0.72231	0.65721	0.62998	0.57007	0.55440	0.57145	0.64446	0.64150	0.67500	0.65321	0.58350	0.52854
LDA	0.79440	0.86750	0.79670	0.74765	0.65013	0.65739	0.60830	0.58159	0.62936	0.65466	0.55977	0.55489	0.56434	0.58335	0.60714
ANN	0.79632	0.83915	0.80065	0.65293	0.71580	0.77290	0.74350	0.71482	0.69594	0.87629	0.82116	0.84562	0.80905	0.88870	0.88822
SVM	0.79056	0.79990	0.79164	0.67806	0.73709	0.68729	0.84132	0.85595	0.80141	0.89503	0.87755	0.91796	0.93695	0.96935	0.98919
RFR	0.97562	0.96683	0.97435	0.62639	0.83562	0.87546	0.91152	0.89429	0.95624	0.91857	0.91359	0.89565	0.94844	0.96040	0.98670
SV	0.78506	0.74898	0.77732	0.66695	0.60062	0.62702	0.71771	0.76098	0.76970	0.76915	0.83481	0.75763	0.63527	0.88397	0.91091

Table A.28: CBOS SPECIFICITY OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.69754	0.72873	0.69990	0.63517	0.68905	0.60305	0.59558	0.59198	0.61881	0.61612	0.62559	0.66365	0.68286	0.67113	0.65845
PR	0.69753	0.68835	0.69645	0.63471	0.62029	0.58052	0.57083	0.59456	0.55755	0.57643	0.63007	0.58613	0.54260	0.56885	0.56664
POI	0.66252	0.65950	0.66210	0.58091	0.52652	0.47555	0.43971	0.46602	0.48769	0.49749	0.41582	0.47168	0.51643	0.47858	0.50026
LDA	0.69385	0.68890	0.69296	0.63549	0.54417	0.51051	0.42910	0.44844	0.43366	0.43230	0.49728	0.45933	0.43589	0.53880	0.51701
ANN	0.69781	0.70152	0.69826	0.58538	0.62645	0.69376	0.70986	0.62063	0.61896	0.75891	0.69132	0.70042	0.78859	0.75079	0.82988
SVM	0.62532	0.69244	0.63163	0.62890	0.71353	0.73638	0.71666	0.77137	0.80424	0.79731	0.82564	0.85527	0.87404	0.88937	0.90887
RFR	0.78733	0.84878	0.79329	0.52931	0.62725	0.77688	0.67812	0.66942	0.72782	0.66005	0.77106	0.79494	0.81775	0.83483	0.87569
SV	0.69531	0.74656	0.70172	0.65582	0.56830	0.59737	0.61441	0.55189	0.66772	0.66232	0.58489	0.62567	0.75619	0.72287	0.78132

Table A.29: SMOTE GINI OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.35973	0.36400	0.36031	0.32393	0.35160	0.29866	0.29535	0.29061	0.30074	0.28906	0.30370	0.29342	0.32559	0.32144	0.33381
PR	0.36019	0.36447	0.36077	0.33085	0.32641	0.29418	0.29138	0.31716	0.32000	0.28363	0.28018	0.28818	0.32773	0.28658	0.33686
POI	0.33481	0.31463	0.33008	0.26366	0.26072	0.23909	0.23284	0.24969	0.23498	0.21349	0.25988	0.24977	0.22291	0.20844	0.20338
LDA	0.35897	0.37289	0.36076	0.31411	0.24715	0.23256	0.22785	0.23256	0.24688	0.23787	0.24420	0.23879	0.24022	0.23881	0.23971
ANN	0.36082	0.34450	0.35727	0.23965	0.33789	0.32274	0.30586	0.37667	0.30730	0.35344	0.38991	0.36923	0.40626	0.37427	0.39860
SVM	0.32538	0.32345	0.32503	0.27962	0.34560	0.30652	0.32299	0.36713	0.35967	0.38398	0.39719	0.38318	0.40190	0.41698	0.42702
RFR	0.31499	0.31959	0.31569	0.20046	0.24712	0.27100	0.31651	0.25208	0.26764	0.28203	0.27140	0.29883	0.33215	0.31369	0.32340
SV	0.35814	0.37702	0.36039	0.32646	0.30594	0.35588	0.31204	0.37101	0.32616	0.39832	0.31850	0.31913	0.30073	0.40294	0.37726

Table A.30: SMOTE KS OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.84877	0.86436	0.84995	0.81758	0.84453	0.80152	0.79779	0.79599	0.80941	0.80806	0.81279	0.83182	0.84143	0.83557	0.82922
PR	0.84877	0.84418	0.84822	0.81736	0.81015	0.79026	0.78541	0.79728	0.77878	0.78822	0.81504	0.79306	0.77130	0.78442	0.78332
POI	0.83126	0.82975	0.83105	0.79045	0.76326	0.73778	0.71986	0.73301	0.74384	0.74875	0.70791	0.73584	0.75821	0.73929	0.75013
LDA	0.84693	0.84445	0.84648	0.81775	0.77208	0.75525	0.71455	0.72422	0.71683	0.71615	0.74864	0.72966	0.71795	0.76940	0.75851
ANN	0.84891	0.85076	0.84913	0.79269	0.81323	0.84688	0.85493	0.81032	0.80948	0.87946	0.84566	0.85021	0.89430	0.87540	0.91494
SVM	0.81266	0.84622	0.81582	0.81445	0.85677	0.86819	0.85833	0.88568	0.90212	0.89866	0.91282	0.92763	0.93702	0.94469	0.95444
RFR	0.89367	0.92439	0.89665	0.76465	0.81362	0.88844	0.83906	0.83471	0.86391	0.83003	0.88553	0.89747	0.90888	0.91741	0.93784
SV	0.84765	0.87328	0.85086	0.82791	0.78415	0.79868	0.80720	0.77594	0.83386	0.83116	0.79244	0.81284	0.87809	0.86144	0.89066

Table A.31: SMOTE AUC OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.76561	0.78910	0.76797	0.69427	0.75118	0.64612	0.64898	0.62388	0.61216	0.63606	0.60657	0.63984	0.64609	0.68181	0.66121
PR	0.76557	0.78907	0.76794	0.70769	0.70604	0.63304	0.66366	0.65911	0.62546	0.62615	0.65730	0.66150	0.65384	0.60654	0.67103
POI	0.75640	0.75832	0.75673	0.59571	0.57004	0.57186	0.59661	0.51790	0.51253	0.49361	0.51787	0.59397	0.49091	0.56538	0.61060
LDA	0.76419	0.74495	0.76082	0.67799	0.59380	0.57574	0.54099	0.54034	0.51207	0.53000	0.52882	0.52289	0.51208	0.51387	0.51621
ANN	0.76533	0.77994	0.76775	0.64051	0.72774	0.62143	0.60320	0.64025	0.77324	0.78926	0.75564	0.77999	0.76433	0.80160	0.87582
SVM	0.75981	0.70439	0.74924	0.53886	0.68818	0.66693	0.78982	0.73653	0.83463	0.80446	0.81315	0.86622	0.89354	0.91677	0.93512
RFR	0.96565	0.94370	0.96114	0.61801	0.89261	0.83532	0.85875	0.77666	0.79512	0.88480	0.92047	0.83701	0.98593	0.98944	0.98313
SV	0.75968	0.76472	0.76055	0.68225	0.61184	0.62785	0.71608	0.73897	0.64019	0.77873	0.82882	0.66842	0.71301	0.78523	0.92623

Table A.32: SMOTE ACCURACY OOT Dynamic

	Development			OOT											
	Built	**Validation**	**Test**	**201801**	**201802**	**201803**	**201804**	**201805**	**201806**	**201807**	**201808**	**201809**	**201810**	**201811**	**201812**
LR	0.77618	0.74055	0.76882	0.67898	0.73948	0.63578	0.61947	0.62096	0.63364	0.64302	0.65113	0.64293	0.69027	0.61022	0.70218
PR	0.77623	0.74059	0.76886	0.70670	0.67728	0.62257	0.64434	0.60287	0.67386	0.60801	0.63340	0.68773	0.69967	0.62748	0.75310
POI	0.79010	0.76801	0.78627	0.64410	0.59289	0.55302	0.56261	0.53144	0.61116	0.59147	0.60021	0.54714	0.55290	0.50876	0.54920
LDA	0.77356	0.75269	0.76945	0.63143	0.51893	0.48004	0.49857	0.52385	0.48772	0.49392	0.50302	0.50635	0.50672	0.50375	0.50397
ANN	0.77804	0.79440	0.78028	0.67133	0.82949	0.72871	0.87229	0.90353	0.83631	0.93586	0.87435	0.93359	0.91695	0.90027	0.94526
SVM	0.76553	0.81317	0.77216	0.68042	0.76000	0.77742	0.82393	0.82617	0.91225	0.83878	0.92917	0.96341	0.96013	0.98546	0.95613
RFR	0.96675	0.91903	0.95634	0.68093	0.79261	0.84252	0.81692	0.96296	0.91063	0.80903	0.98439	0.86026	0.90562	0.94892	0.97403
SV	0.76134	0.80818	0.76408	0.62094	0.64475	0.64857	0.66428	0.65003	0.58115	0.63213	0.73980	0.77041	0.83910	0.67999	0.75133

Table A.33: SMOTE SPECIFICITY OOT Dynamic

Appendix B

Python Code

B.1 Code for Sampling

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from deslib.des.meta_des import METADES
from sklearn.datasets import make_classification
from imblearn.over_sampling import SMOTE
from sklearn.cluster import DBSCAN
from imblearn.over_sampling import RandomOverSampler

from clover.over_sampling import ClusterOverSampler
from imblearn.over_sampling import SMOTE
from collections import Counter
from numpy import where
from sklearn.datasets import make_classification
```

```
from matplotlib import pyplot
from sklearn.model_selection import train_test_split
from itertools import permutations,combinations


#################################################################
#################################################################
```

B.2 Code for Modelling

B.2.1 Logistic Regression

```
from sklearn.linear_model import LogisticRegression
import scorecardpy as sc
import pandas as pd
import numpy as np
import csv
import itertools


lr = LogisticRegression(penalty='l1', C=0.9, solver='saga', n_jobs=-1)
lr.fit(X_built, y_built)

card_lr = sc.scorecard(bins, lr, X_built.columns)


# predicted probability
built_pred_lr = lr.predict_proba(X_built)[:,1]
valid_pred_lr = lr.predict_proba(X_valid)[:,1]
test_pred_lr = lr.predict_proba(X_test)[:,1]

built_score_lr = sc.scorecard_ply(built, card_lr, print_step=0)
valid_score_lr = sc.scorecard_ply(valid, card_lr, print_step=0)
test_score_lr = sc.scorecard_ply(test, card_lr, print_step=0)

```

```
24
25 built_perf = sc.perf_eva(y_built, built_pred_lr, title = "built" )
26 valid_perf = sc.perf_eva(y_valid, valid_pred_lr, title = "valid" )
27 test_perf = sc.perf_eva(y_test, test_pred_lr, title = "test" )
```

B.2.2 Probit Regression

```
1 from sklearn.linear_model import LogisticRegression
2 import scorecardpy as sc
3 import pandas as pd
4 import numpy as np
5 import csv
6 import itertools
7
8 #Special case of a Logistic Regression
9 probit = LogisticRegression(random_state=0, multi_class='multinomial',
10          penalty='none', solver='newton-cg').fit(X_built, y_built)
11
12 # predicted probability
13 built_pred_pr = probit.predict_proba(X_built)[:,1]
14 valid_pred_pr = probit.predict_proba(X_valid)[:,1]
15 test_pred_pr = probit.predict_proba(X_test)[:,1]
16
17 card_pr = sc.scorecard(bins, probit, X_built.columns)
18
19
20 built_score_pr = sc.scorecard_ply(built, card_pr, print_step=0)
21 valid_score_pr = sc.scorecard_ply(valid, card_pr, print_step=0)
```

```
test_score_pr = sc.scorecard_ply(test, card_pr, print_step=0)

built_perf_pr = sc.perf_eva(y_built, built_pred_pr, title = "built" )
valid_perf_pr  = sc.perf_eva(y_valid, valid_pred_pr, title = "valid" )
test_perf_pr  = sc.perf_eva(y_test, test_pred_pr, title = "test" )
```

B.2.3 Poisson Regression

```

import scorecardpy as sc
import pandas as pd
import numpy as np
import csv
import itertools
from scipy import stats
from sklearn import linear_model


poi = linear_model.PoissonRegressor()

poi.fit(X_built, y_built)
poi.score(X_built, y_built)


built_pred_poi= np.exp(poi._linear_predictor(X_built)+ poi.intercept_)
        /np.exp(-(np.exp(poi.predict(X_built))))
test_pred_poi = np.exp(poi._linear_predictor(X_valid)+ poi.intercept_)
        /np.exp(-(np.exp(poi.predict(X_valid))))
```

```
test_pred_poi = np.exp(poi._linear_predictor(X_test)+ poi.intercept_)
        /np.exp(-(np.exp(poi.predict(X_test))))


card_poi = sc.scorecard_poi(bins, clf, X_built.columns)


built_score_poi = sc.scorecard_ply(built, card_poi, print_step=0)
valid_score_poi = sc.scorecard_ply(valid, card_poi, print_step=0)
test_score_poi = sc.scorecard_ply(test, card_poi, print_step=0)


built_perf_poi= sc.perf_eva(y_built, built_pred_poi, title = "built" )
valid_perf_poi  = sc.perf_eva(y_valid, valid_pred_poi, title = "valid" )
test_perf_poi  = sc.perf_eva(y_test, test_pred_poi, title = "test" )
```

B.2.4 Linear Discriminant Analysis

```

import scorecardpy as sc
import pandas as pd
import numpy as np
import csv
import itertools
from scipy import stats
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis


```

```
lda = LinearDiscriminantAnalysis().fit(X_built, y_built)


built_pred_lda = lda.predict_proba(X_built)[:,1]
valid_pred_lda = lda.predict_proba(X_valid)[:,1]
test_pred_lda = lda.predict_proba(X_test)[:,1]



card_lda = sc.scorecard(bins, lda, X_built.columns)

built_score_lda = sc.scorecard_ply(built, card_lda, print_step=0)
valid_score_lda = sc.scorecard_ply(valid, card_lda, print_step=0)
test_score_lda = sc.scorecard_ply(test, card_lda, print_step=0)



built_perf_lda = sc.perf_eva(y_built, built_pred_lda, title = "built" )
valid_perf_lda  = sc.perf_eva(y_valid, valid_pred_lda, title = "valid" )
test_perf_lda  = sc.perf_eva(y_test, test_pred_lda, title = "test" )
```

B.2.5 Artificial Neural Networks

```

from sklearn.neural_network import MLPClassifier
from sklearn.datasets import make_classification
from sklearn.model_selection import built_test_split
import scorecardpy as sc
```

```
import pandas as pd
import numpy as np
import csv
import itertools
from scipy import stats


ann = MLPClassifier(random_state=3,hidden_layer_sizes=1,
  activation='logistic', max_iter=300).fit(X_built, y_built)


built_pred_ann = ann.predict_proba(X_built)[:,1]
valid_pred_ann = ann.predict_proba(X_valid)[:,1]
test_pred_ann = ann.predict_proba(X_test)[:,1]


card_ann= sc.scorecard_ann(bins, ann, X_built.columns)

built_score_ann = sc.scorecard_ply(built, card_ann, print_step=0)
valid_score_ann = sc.scorecard_ply(valid, card_ann, print_step=0)
test_score_ann = sc.scorecard_ply(test, card_ann, print_step=0)


built_perf_ann = sc.perf_eva(y_built, built_pred_ann, title = "built" )
valid_perf_ann  = sc.perf_eva(y_valid, valid_pred_ann, title = "valid" )
test_perf_ann  = sc.perf_eva(y_test, test_pred_ann, title = "test" )
```

B.2.6 Support Vector Machines

```
import numpy as np
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
import scorecardpy as sc
import pandas as pd
import csv
import itertools
from scipy import stats

svm = SVC(kernel='linear').fit(X_built, y_built)



built_pred_svm= svm.predict(X_built)
valid_pred_svm =svm.predict(X_valid)
test_pred_svm =svm.predict(X_test)

card_svm= sc.scorecard(bins, svm, X_built.columns)

built_perf_svm = sc.perf_eva(y_built, built_pred_svm, title = "built" )
valid_perf_svm = sc.perf_eva(y_valid, valid_pred_svm, title = "valid" )
test_perf_svm  = sc.perf_eva(y_test, test_pred_svm, title = "test" )
```

B.2.7 Random Forest Regression

```
from sklearn.ensemble import RandomForestClassifier
import pydot
import scorecardpy as sc
import pandas as pd
import csv
import itertools
from scipy import stats

rfr = RandomForestClassifier(n_estimators=10, bootstrap = True,
  max_features = 'sqrt').fit(X_built, y_built)

built_pred_rfr = rfr.predict(X_built)
valid_pred_rfr = rfr.predict(X_valid)
test_pred_rfr = rfr.predict(X_test)


card_rfr= sc.scorecard_rfr(bins, rfr, X_built.columns)


built_perf_rfr = sc.perf_eva(y_built, built_pred_rfr, title = "built" )
valid_perf_rfr = sc.perf_eva(y_valid, valid_pred_rfr, title = "valid" )
test_perf_rfr  = sc.perf_eva(y_test, test_pred_rfr, title = "test" )
```

B.2.8 Survival Modelling

```

from lifelines import CoxPHFitter
import scorecardpy as sc
```

```
import pandas as pd
import numpy as np
import csv
import itertools
from scipy import stats

cph = CoxPHFitter()
cph.fit(built_woe,duration_col= 'week',  event_col='gb_ind')


built_pred_sv = cph.predict_survival_function(built_woe1).T
valid_pred_sv = cph.predict_survival_function(valid_woe1).T
test_pred_sv = cph.predict_survival_function(test_woe1).T

card_sv= sc.scorecard_sm(bins, cph, X_built.columns)


built_perf_sm = sc.perf_eva(y_built, built_pred_sm, title = "built" )
valid_perf_sm = sc.perf_eva(y_valid, valid_pred_sm, title = "valid" )
test_perf_sm = sc.perf_eva(y_test, test_pred_sm, title = "test" )
```

References

Abdou, H. A. and Pointon, J. (2011). Credit scoring, statistical techniques and evaluation criteria: a review of the literature. *Intelligent systems in accounting, finance and management*, 18(2-3):59–88.

Act, N. C. (2006). Government Gazette Republic of South Africa. *Cape Town*, 489(28619):1–10.

Al Ghayab, H. R., Li, Y., Abdulla, S., Diykh, M., and Wan, X. (2016). Classification of epileptic eeg signals based on simple random sampling and sequential feature selection. *Brain informatics*, 3(2):85–91.

Alfat, L., Rizkinia, M., Sari, R. F., and Romano, D. M. (2019). Feature selection of credit score factor based on smartphone usage using mcfs. In *2019 4th Technology Innovation Management and Engineering Science International Conference (TIMES-iCON)*, pages 1–5.

Ali, H., Salleh, M., Hussain, K., Ullah, A., Ahmad, A., and Naseem, R. (2019). A review on data preprocessing methods for class imbalance problem. pages 390–397.

Altland, H. (2012). Regression analysis: Statistical modeling of a response variable. *Technometrics*, 41:367–368.

Anderson, R. (2020). *Who Developed The First Credit Scoring Model?* `https://www.quora.com/Who-developed-the-first-credit-scoring-model-1`[Accessed: May 2020].

Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., and Vanthienen, J. (2003). Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the operational research society*, 54(6):627–635.

Basel Accord, I. (2004). Basel accord committee on banking supervision: International convergence of capital measurement and capital standards: A revised framework.

Benesty, J., Chen, J., Huang, Y., and Cohen, I. (2009). Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer.

Bijak, K. and Thomas, L. C. (2012). Does segmentation always improve model performance in credit scoring? *Expert Systems with Applications*, 39(3):2433–2442.

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

Bücker, M., Szepannek, G., Biecek, P., Gosiewska, A., and Staniak, M. (2019). Transparency of machine learning models in credit scoring.

Cai, D., Zhang, C., and He, X. (2010). Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, page 333–342, New York, NY, USA. Association for Computing Machinery.

Casin, P. (2018). Categorical multiblock linear discriminant analysis. *Journal of Applied Statistics*, 45(8):1396–1409.

Cateni, S., Colla, V., and Vannucci, M. (2017). A fuzzy system for combining filter features selection methods. *International Journal of Fuzzy Systems*, 19(4):1168–1180.

Chasco, C., Aroca, P., and Anselin, L. (2019). Probit models for grouped-data migration flows: A theoretical note. *Economia*, 42(84):1–8.

Chawla, N., Bowyer, K., Hall, L., and Kegelmeyer, W. (2011). Smote: Synthetic minority over-sampling technique. *arXiv*, pages arXiv–1106.

Chimedza, C. and Marimo, M. (2017). Survival analysis of bank loans in the presence of long-term survivors. *South African Statistical Journal*, 51(1):199–216.

Crone, S. F. and Finlay, S. (2012). Instance sampling in credit scoring: An empirical study of sample size and balancing. *International Journal of Forecasting*, 28(1):224–238.

Cruz, R. M. O., Hafemann, L. G., Sabourin, R., and Cavalcanti, G. D. C. (2020). Deslib: A dynamic ensemble selection library in python. *Journal of Machine Learning Research*, 21(8):1–5.

Das, A. (2019). *Oversampling to remove class imbalance using SMOTE.* `https://medium.com/@asheshdas.ds/oversampling-to-remove-class-imbalance - using -smote-94d5648e7d35`[Accessed: Jun 2020].

Dong, G., Lai, K. K., and Yen, J. (2010). Credit scorecard based on logistic regression with random coefficients. *Procedia Computer Science*, 1(1):2463–2468.

Doori, M. and Beyrouti, B. (2014). Credit scoring model based on back propagation neural network using various activation and error function. *IJCSNS International Journal of Computer Science and Network Security*, 14(3):16–24.

Douzas, G., Bacao, F., and Last, F. (2018). Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Information Sciences*, 465:1–20.

Drezner, Z., Turel, O., and Zerom, D. (2008). A modified kolmogorov-smirnov test for normality. *University Library of Munich, Germany, MPRA Paper*, 39.

Ghodselahi, A. and Amirmadhi, A. (2011). Application of artificial intelligence techniques for credit risk evaluation. *International Journal of Modeling and Optimization*, 1(3):243.

Gillani, R. and Zubair, S. M. (2016). An improved bat algorithm with artificial neural networks for classification problems.

Habibzadeh, F., Habibzadeh, P., and Yadollahie, M. (2016). On determining the most appropriate test cut-off value: The case of tests with continuous results. *Biochemia Medica*, 26:297–307.

Hand, D. J. and Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: a review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 160(3):523–541.

Henley, W. E. (1995). *Statistical aspects of credit scoring*. Open University (United Kingdom).

Hooman, A., Marthandan, G., Yusoff, W. F. W., Omid, M., and Karamizadeh, S. (2016). Statistical and data mining methods in credit scoring. *The Journal of Developing Areas*, 50(5):371–381.

Hussain, A., Khan, M., Rehman, S. U., and Khattak, A. (2019). Credit scoring model for retail banking sector in Pakistan. *Journal of Managerial Sciences*, 14(4):153–161.

Ij, H. (2018). Statistics versus machine learning. *Nat Methods*, 15(4):233.

Investopedia, T. I. (2020). *Credit: What everyone should know?* `https://www.investopedia.com/terms/c/credit.asp`[Accessed: Jun 2020].

Jing, C., Ling, X., Dun-Hu, L., and Jin, X. (2017). Effect analysis of resampling techniques on the performance of customer credit scoring models. *DEStech Transactions on Computer Science and Engineering*, (csma).

Junior, L. M., Nardini, F. M., Renso, C., and de Macêdo, J. A. F. (2019). On combining dynamic selection, sampling, and pool generators for credit scoring. In *MLDM (2)*, pages 443–457.

Karlis, D. and Rahmouni, M. (2007). Analysis of defaulters' behaviour using the poisson-mixture approach. *IMA Journal of Management Mathematics*, 18(3):297–311.

Lemaître, G., Nogueira, F., and Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5.

Lessmann, S., Baesens, B., Seow, H.-V., and Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, 247(1):124–136.

Lund, B. and Brotherton, D. (2013). Information value statistic, mwsug 2013. In *Proceedings*.

Man, R. (2014). Survival analysis in credit scoring: A framework for PD estimation. Master's thesis, University of Twente.

MathWorks (2020). *About Credit Scorecards*. `https://www.mathworks.com/help/finance/about-credit-scorecards.html`[Accessed: Jun 2020].

Mercadier, M. and Lardy, J.-P. (2019). Credit spread approximation and improvement using random forest regression. *European Journal of Operational Research*, 277(1):351–365.

Michie, D., Spiegelhalter, D. J., and Taylor, C. C. (1994). Machine learning, neural and statistical classification.

Nehrebecka, N. (2018). Predicting the default risk of companies. Comparison of credit scoring models: LOGIT vs Support Vector Machines. *Econometrics*, 22(2):54–73.

Pacelli, V. and Azzollini, M. (2011). An artificial neural network approach for credit risk management. *Journal of Intelligent Learning Systems and Applications*, 3(02):103.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pérez-Ortiz, M., Gutiérrez, P. A., Tino, P., and Martínez, C. (2015). Oversampling the minority class in the feature space. *IEEE Transactions on Neural Networks and Learning Systems*, 27:1–1.

Ratnasari, D., Nazir, F., Husein Zilullah Toresano, L. O., Pawiro, S., and Soejoko, D. (2016). The correlation between effective renal plasma flow (erpf) and glomerular filtration rate (gfr) with renal scintigraphy 99m tc-dtpa study. *Journal of Physics: Conference Series*, 694:012062.

Ren, D., Hou, M., and Li, H. (2013). A study of research and application of credit scoring model based on probit model. In *The 19th International Conference on Industrial Engineering and Engineering Management*, pages 1–13. Springer.

Saitoh, F. (2016). Predictive modeling of corporate credit ratings using a semi-supervised random forest regression. In *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 429–433. IEEE.

Segal, M. (2004). Machine learning benchmarks and random forest regression. *Technical Report, Center for Bioinformatics & Molecular Biostatistics, University of California, San Francisco*.

Siddiqi, N. (2017). *Intelligent credit scoring: Building and implementing better credit risk scorecards*. John Wiley & Sons.

Tian, Y., Xiao, Z., Wang, H., Peng, X., Guan, L., et al. (2017). Influence of the sampling period and time resolution on the pm source apportionment: Study based on the high time-resolution data and long-term daily data. *Atmospheric Environment*, 165:301–309.

TransUnion (2020). *Driving greater predictive power to help you harness macroeconomic changes*. `https://www.transunion.co.za/product/empirica`[Accessed: May 2020].

Van Gestel, I. T., Baesens, B., Garcia, I. J., and van Dijcke, P. (2003). A support vector machine approach to credit scoring. In *FORUM FINANCIER-REVUE BANCAIRE ET FINANCIAIRE BANK EN FINANCIEWEZEN-*, pages 73–82. UNKNOWN.

Wang, G., Hao, J., Ma, J., and Jiang, H. (2011). A comparative assessment of ensemble learning for credit scoring. *Expert systems with applications*, 38(1):223–230.

West, D. (2000). Neural network credit scoring models. *Computers & Operations Research*, 27(11-12):1131–1152.

Yang, J., Gong, J., Tang, W., Shen, Y., Liu, C., and Gao, J. (2019). Delineation of urban growth boundaries using a patch-based cellular automata model under multiple spatial and socio-economic scenarios. *Sustainability*, 11(21):6159.

Yao, J.-R. and Chen, J.-R. (2019). A new hybrid support vector machine ensemble classification model for credit scoring. *Journal of Information Technology Research (JITR)*, 12(1):77–88.

Zeng, G. (2013). Metric divergence measures and information value in credit scoring. *Journal of Mathematics*, 2013.

Zeng, G. (2020). On the confusion matrix in credit scoring and its analytical properties. *Communications in Statistics-Theory and Methods*, 49(9):2080–2093.

Zhang, J., Chen, W., Gao, M., and Shen, G. (2017). K-means-clustering-based fiber nonlinearity equalization techniques for 64-qam coherent optical communication system. *Opt. Express*, 25(22):27570–27580.

Zhu, B., Baesens, B., Backiel, A., and vanden Broucke, S. K. L. M. (2018). Benchmarking sampling techniques for imbalance learning in churn prediction. *Journal of the Operational Research Society*, 69(1):49–65.