Student Declaration of Authorship

HERIOT WATT UNIVERSITY
UK | DUBAI | MALAYSIA

Course code and name:	F21MP Masters Project and Dissertation
Type of assessment:	**Individual**
Coursework Title:	Dissertation
Student Name:	Josh Yang
Student ID Number:	091514042

Declaration of authorship. By signing this form:

- **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section.

- I confirm that I have read, understood and followed the University's Regulations on plagiarism as published on the University's website, and that I am aware of the penalties that I will face should I not adhere to the University Regulations.

- I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on Academic Integrity and Plagiarism

Student Signature *(type your name):* *Josh Yang*

Date: *15/08/2022*

Copy this page and insert it into your coursework file in front of your title page.
For group assessment each group member must sign a separate form and all forms must be included with the group submission.

Your work will not be marked if a signed copy of this form is not included with your submission.

Heriot-Watt University

Masters Thesis

The Performance of Reject Inference Techniques in a Credit Decision Framework

Author:
Josh Aide Yang

Supervisor:
Dr. Theodoros Georgiou

A thesis submitted in fulfilment of the requirements for the degree of MSc. Data Science

in the

School of Mathematical and Computer Sciences

August 2022

Declaration of Authorship

I, Josh Aide YANG, declare that this thesis titled, 'The Performance of Reject Inference Techniques in a Credit Decision Framework' and the work presented in it is my own. I confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: Josh Aide Yang

Date: 15/08/2022

Abstract

In the finance industry, machine learning algorithms have been trained to score the risk of applicants for credit products such as credit cards or loans. Without necessary intervention, these algorithms are trained on data only from those applicants that were offered and accepted the credit from the lender. This can result in selection bias as the algorithm is only trained on the available data, approved customers, as there is no available information on those applicants that were rejected to determine if they would have been good or bad customers. Behaviour of new applicants may have changed and differ from those that the model has been trained on.

Reject inference has been proposed as a technique for improving the quality and robustness of a credit scoring framework by incorporating data from rejected applications. Several techniques have been proposed in the current literature with varying results and no clear standout.

This project investigates the applicability of reject inference and compares the performance of techniques by implementing a Random Forest (RF) model to predict the likelihood of issued peer-to-peer (P2P) loan defaults. Benchmark model performance results are reported on the accepted applicant data only. A selection of both statistical and machine learning reject inference techniques is then applied, and the synthesised dataset is fitted and re-evaluated on the same model to provide a comparison.

Analysis of the results confirms previous research findings that there is not a standout reject inference technique. Although, the statistical technique augmentation shows promising recall score, increase of over 17%, for the default class, vital for lenders.

Acknowledgments

First and foremost, I would like to thank my supervisor Dr. Theodoros Georgiou for supporting me throughout my trials and tribulations experienced during this project. I am grateful towards you for keeping me focussed and level-headed. I would also like to thank The Data Lab for funding my studies, without it I wouldn't be able to complete my MSc. Finally, I would like to pay special thanks to my mother for continuously supporting me in any way she possibly can. I will always love and cherish you.

Contents

List of Figures

List of Equations

Abbreviations

3WD	Three-way Decision
AR	Accept / Reject Model
AUC	Area Under the Curve
BIC	Bayesian Information Criterion
BN	Bayesian Network
CPD	Conditional Probability Distribution
DAG	Directed Acyclic Graph
FN	False Negatives
FP	False Positives
GB	Good / Bad Model
KNN	k-Nearest Neighbour
LR	Logistic Regression
MICE	Multiple Imputation by Chained Equations
MAR	Missing At Random
MCAR	Missing Completely At Random
MNAR	Missing Not At Random
P2P	Peer-to-peer
PCC	Percentage Correctly Classified
RF	Random Forest
ROC	Receiver Operating Characteristic
S3VM	Semi-supervised Support Vector Machine
SSL-EC3	Semi-supervised Learning Ensemble Learning Framework
STL	Self-Taught Learning
SVM	Support Vector Machine
TN	True Negatives
TP	True Positives
TTD	Through-The-Doo

1. Introduction

1.1. Overview

Data is never perfect. It is normal for raw data to be missing values (Luca & Desjardins, 2018). How we handle these missing values can affect the performance of our models and their development. It is especially pernicious when it comes to explaining how a model arrived at a decision.

Financial services use machine learning models, a program used to identify patterns in unseen data, to evaluate potential customers for a variety of products such as credit loans and insurance. The ability to discern differences in characteristics of good and bad applicants for these products is a critical feature of a good model (Banasik & Crook, 2007). These models are de facto trained on the known performance of the provider's customers (accepted applicants), such as defaulting on their credit. This leads to a difference in the samples used for retraining the model or training a new model and the samples pertaining to new applications (Li, et al., 2017). In other words, an inherent bias is introduced as the model has no knowledge of the characteristics of the applicants that were non-randomly rejected from the outset. In effect removed from the model's training dataset. Thus, predictions for new applicants are based on the parameters of the previously accepted applicants only and the model is considered stale (Crook & Banasik, 2004). To illustrate, if some of the applicants in the through-the-door (TTD) population, all applicants, had a characteristic such as "bankruptcy" and this feature was considered a derogatory characteristic by the evaluator, their application would likely be rejected. As a result, future models trained on the accepted population would not have any knowledge of the 'bankruptcy' characteristic, as there are no or a low number of instances with this predictive variable, thus will not apply sufficient weighting to this key discriminatory indicator of a most likely bad applicant (Henley, 1995).

To improve the quality of these classification models, it is worthwhile to consider this missing data and reduce the selection bias as it is typically not ignorable (Ha-Thu, 2016). The higher the rejection rate, the larger the impact of sample bias (Henley, 1995). Ideally, models should be built based on the total population of applicants, including rejected applications with their true outcome (typically unknown in practice), as complete information allows for better discriminating power between good and bad applicants (Henley, 1995). Reject inference has been proposed as a technique to infer the unknown performance of those applicants that were rejected and include this sample within the training set for subsequent iterations. Section 3.2. will provide a discussion on reject inference and the various techniques proposed in the literature. A more representative dataset of the customer pool should provide enhanced performance of the classification model (Anderson, 2019). New customers, whose characteristics may have evolved and changed in comparison to the current customer pool, referred to as population drift (Henley, 1995), will be evaluated more fairly and should result in more business.

1.2. Aims

This research project aims to:

> *Investigate the performance of a selection of reject inference techniques in a peer-to-peer (P2P) credit application framework.*

Firstly, by analysing a P2P lending dataset, we will be able to show support for the implementation of reject inference techniques in a credit application framework. Secondly, this research aims to provide an evaluation and analysis of a selection of reject inference technique's relative performance and appropriateness. Lastly, this research aims to identify and propose the principal technique within the credit application context.

1.3. Objectives

To realise the aims, the following objectives are outlined:

1. Review current literature on reject inference techniques.
2. Develop an appropriate credit application framework.
3. Develop reject inference techniques.
4. Apply reject inference techniques to appropriate datasets.
5. Benchmark classification performance of machine learning model.
6. Evaluate performance of the credit application framework with selected reject inference techniques.
7. Highlight key deviations in performance between benchmark and selected reject inference techniques.
8. Propose superior credit application framework.

1.4. Report Structure

The chapters of this report are structured as follows:

- **Chapter 2, Literature Review:** The foundation and concepts of the research are established based on previous work by scholars. This will provide research-based support for the analysis methodology proposed.
- **Chapter 3, Risk Analysis:** Any potential risks that will hamper or halt the project will be laid out.
- **Chapter 4, Discussion of Professional, Legal, Ethical and Social Issues:** A discussion is provided on the potential issues that can arise from this study.
- **Chapter 5, Methodology:** Based on the previous sections, we will outline the proposed methodology which is suitable for this study.
- **Chapter 6, Implementation:** We will discuss our implementation and any design choices so our study can be replicated.
- **Chapter 7, Results & Findings:** Presentation of the results and discussion of any initial findings

- **Chapter 8, Discussion:** The results will be analysed, and reference made to the previous literature.
- **Chapter 9, Limitations:** Any factors which have impacted the project and / or the results.
- **Chapter 10, Conclusion:** The research will be summarised, and key takeaways presented.
- **Chapter 11, Further Work:** Suggested extensions of this project.

2. Literature Review

2.1. Introduction

A literature review has been carried out to gain an understanding of reject inference and how performance is analysed. It has been identified that reject inference techniques can be broken down into two broad categories: statistical and machine learning. The following sections will explore the various reject inference techniques proposed in the current literature. Subsequently, the arguments for and against implementing reject inference will be presented. Finally, in this chapter, we will explain the different metrics used for evaluating the performance of classification models.

2.2. Reject Inference

Reject inference is a method for improving the quality and applicability of a dataset based on the use of parameters contained in rejected applications and their estimated performance. In general, no information is available on the performance of rejected applications thus it is uncertain whether those applicants would have been good or bad customers. Supplementary information about rejected applications may be available, for example acquired from a credit bureau, but it is not always realistic for all providers as it can be expensive or unreliable (Mancisidor, et al., 2020). The proposed benefit of determining those rejected applicants that were in fact good is to identify new customers at a lower potential risk.

Reject inference is a form of missing values treatment where the outcomes are grouped into one of three groups (Ha-Thu, 2016):

- Missing completely at random (MCAR)
- Missing not at random (MNAR)
- Missing at random (MAR)

MCAR indicates that applicants are accepted and rejected at random, independent of all their characteristics (Henley, 1995). Sample bias would not exist in this instance. MNAR are manual overrides of the model, such as a human loan officer altering the decision according to personal experience or business strategy. These potentially unobservable variables (human making decisions) make research difficult (Liu, et al., 2020). Additionally, cases where variables are missing or excluded, effecting the decision on an application, would be considered MNAR (Banasik & Crook, 2007). If it does not affect the decision then the missing values would be considered MAR. MAR is the primary missing data type addressed in previous literature as applications are classified based on an applicant's characteristics only and are accepted if they meet a certain threshold (Liu, et al., 2020).

The primary use for reject inference is within a credit framework. It is crucial for lenders to accurately predict the default risk (Turiel & Aste, 2020), inability to repay debt obligations (Mancisidor, et al., 2020), of any new applicant based on their profile by considering information such as marital status, age and employment status, and the requested credit. The predictive model will output a credit score based on these variables and if the score is above a predefined threshold, arbitrarily set by the lender, they will usually be approved for the credit. Although, previous research mainly focused on applying reject inference techniques to credit scoring models, the technique can be applied to various other contexts to good effect (Banasik & Crook, 2007).

By default, credit scoring models are built on datasets of only approved customers as the institution will only hold information on those applicants' credit statuses (Ehrhardt, et al., 2021; Shen, et al., 2020). This implies that any subsequent applicants for credit, the TTD population, will be scored by a model that has been trained on a dataset that is not representative. Therefore, such models are biased, detracting from the credibility of the scoring model.

Crook & Banasik (2004) confirmed this bias when comparing the performance of a model trained on accepts-only on the hold-out sample and TTD applicants test sample, there was a significant drop-off in the Area Under the Curve (AUC) metric, explained in Section 3.3.2, especially if there was a high cut-off rate. This error was referred to as 'accept analysis delusion'.

Although, conversely, Luca & Desjardins (2018) observed that the AUC for a model trained on accepts-only was higher when applying the sample of TTD applicants (0.7496) to the sample of accepts-only (0.7326). In the same study, when comparing results of a model trained on TTD applications (including rejects) with known performance (0.754) to the aforementioned model trained on accepts-only, as we can observe, the latter showed a slight degradation in the AUC metric when classifying unobserved TTD applications. This contrast in results may lead us to believe that performance benefits gained from reject inference are not guaranteed.

The following sections will delve into the reject techniques that have been proposed in the literature. First, we will discuss the statistical techniques.

2.2.1. Statistical Techniques

Statistical techniques will mathematically estimate the performance of the rejected sample and then include the estimation within the training set for the machine learning model. We will describe in detail assignment, augmentation, extrapolation, fuzzy augmentation, parcelling and reclassification.

2.2.1.1. Assignment

The most basic technique for handling the rejected sample is assignment. The simplest assignment is not an actual reject inference process and elects to either ignore the rejected sample completely or assigns all rejects to the bad class. We can also randomly assign the rejects into good and bad classes with a bad rate two to five times greater than in the accepted population strategy. This is referred to as proportional assignment.

2.2.1.2. Augmentation (Re-weighting)

Augmentation, also referred to as re-weighting in the literature (Luca & Desjardins, 2018; Ha-Thu, 2016), is a popular technique that involves artificially expanding the size of the dataset by

creating modified data. In terms of reject inference, it entails weighting the accepted applicants such as to obtain a distribution that fully accounts for the contribution of rejected applicants (Banasik & Crook, 2007). The first step is to build an estimation of the Accept / Reject (AR) model which will provide the probability that an applicant will be accepted as a customer. The model is trained on the TTD population which are labelled either accepted or rejected.

The training sample instances are then ordered by acceptance probability and split into equal intervals of equivalent probabilities. The acceptance cut-off should be set so that the actual number of accepts in the training set equals the number of accepts predicted by the model. Table 1 (Banasik & Crook 2007) clarifies the method as 1289 instances, with a minimum probability of 0.48605, equals 1289 actual accepts in the training set. Table 1 (Banasik & Crook 2007) shows that 167 instances were erroneously classified as accept by the AR model and thus 167 true accepts classified as rejects. As this method uses the actual accepts only, as we know their true performance, to represent the total population, a large emphasis is attributed to those 167 true accepts classified as rejects by the AR model.

Interval	*P*(Accept) range within interval	Total				Training proportion			Represented by accepts
		Good	Bad	Accepts	Rejects	Cases	Accepted	Weights	
1	.99997–1.0000	126	3	129	0	129	1.00000	1.00	129
2	.99587–.99997	109	20	129	0	129	1.00000	1.00	129
3	.98302–.99587	113	16	129	0	129	1.00000	1.00	129
4	.96095–.98302	113	13	126	3	129	.97674	1.02	129
5	.93144–.96095	116	10	126	3	129	.97674	1.02	129
6	.88551–.93144	101	19	120	8	128	.93750	1.07	128
7	.82116–.88551	100	15	115	14	129	.89147	1.12	129
8	.72150–.82116	83	10	93	36	129	.72093	1.39	129
9	.60282–.72150	73	11	84	45	129	.65116	1.54	129
10	.48605–.60282	66	5	71	58	129	.55039	1.82	129
Subtotal				1122	167	1289	.87044		1289
11	.35984–.48605	48	3	51	78	129	.39535	2.53	129
12	.24927–.35984	34	2	36	93	129	.27907	3.58	129
13	.16051–.24927	20	3	23	106	129	.17829	5.61	129
14	.10240–.16051	17	3	20	109	129	.15504	6.45	129
15	.00000–.10240	31	6	37	4604	4641	.00797	125.43	4641
Total				1289	5157	6446			6446

Table 1 - Augmentation illustration of an Individual Interval (Banasik & Crook 2007)

The key assumption of this method is that within each interval, *j*, the probability that an *accepted* applicant with score S_j is good equals the probability that a *rejected* applicant with score S_j is good (Henley, 1995).

$$P(g|S_j, A) = P(g|S_j, R)$$

Equation 1 – Augmentation Assumption

To calculate the weight for each interval, *j*, see Equation 2. The A_j accepts represent both the A_j and R_j cases.

$$W_{aug} = \frac{(R_j + A_j)}{A_j}$$

Equation 2 - Augmentation Interval Weight Equation

If we consider individual values, *i*, each accepted instance has a probability sampling weight of Equation 3. This is the inverse of the estimated probability of acceptance $P(A_i)$.

$$W_{aug}(i) = \frac{1}{P(A_i)}$$

Equation 3 - Augmentation Individual Weight Equation

This ratio will reweight the accepted applications so that the number of the weights will equal the total number of applications (Ha-Thu, 2016). Accepts which have relatively low probabilities of acceptance will have relatively high weights applied (Banasik & Crook, 2007). We expect these instances to have more similar characteristics to the rejected instances than those that have a high probability of acceptance. The purpose of augmentation is to focus attention towards more risky accepted instances (Banasik & Crook, 2007).

The next step is to build a Good / Bad (GB) model and fit the weighted accept sample. This model should now have learned characteristics of the rejected applicants.

Augmentation assumes the missing values are MAR. In other words, the model can only be estimated if all explanatory variables are known and there are no manual interventions (Crook & Banasik, 2004).

Anderson (2019) description of augmentation is different to what we have just illustrated. In their study, augmentation first builds a model trained on the known performance of the accepted applicant data and then the rejected applicant sample is applied to determine their predicted probabilities. A cut-off threshold then determines if a rejected instance is good or bad. In our opinion, this description is in essence extrapolation, see the next Section 2.2.1.3. Our interpretation of augmentation is what has been described in other studies, reviewed above, and will be the process we follow.

2.2.1.3. Extrapolation (Hard Cut Off)

Extrapolation estimates the probability to be bad for each instance in the rejected sample based on a model that has been trained on the known characteristics and performance of the approved sample (Ha-Thu, 2016; Crook & Banasik, 2004; Henley, 1995). Like augmentation, section 2.2.1.2, the underlying assumption is the repayment behaviour of the rejected sample is the same as accepted sample, Equation 1. Using a cut-off value, extrapolation assigns rejects with scores below to the bad class and scores above to the good class (Anderson, 2019). The main challenge in this method is choosing the appropriate cut-off value. It has been suggested that the cut-off should be set so that the actual and predicted number of bad are equal (Crook & Banasik, 2004). This approach is also called the hard cut-off technique.

After the rejects have been classified, the entire population of accepts and rejects are fitted in the final model (Crook & Banasik, 2004).

Like augmentation, the presumption is the missing values are MAR and there no unobserved variables (Crook & Banasik, 2004).

2.2.1.4. Fuzzy Augmentation

The fuzzy augmentation method involves using the *rejected* sample with weight values that correspond to the probability of an instance being approved or rejected. Each reject is then duplicated into two observations and assigned a weighted probability of being good and bad, both derived from the original score (Anderson, 2019). The two probabilities must equal to 1 ($P(B) = 1 - P(G)$) (Luca & Desjardins, 2018). Accepted applications are given a weight equal to 1 (Ha-Thu, 2016). This synthesised sample, which represents the rejected applicants, is then aggregated to the accepted applicant sample (Banasik & Crook, 2007). A new model is then built on this new, representative data set.

2.2.1.5. Parcelling

Parcelling is a hybrid method encompassing augmentation and proportional assignment (Anderson, 2019). Parcels are created by binning the rejects' probabilities, generated using the original AR model, into intervals. Proportional assignment is then applied on each parcel with a probable bad rate which will be greater than the bad rate in the equivalent score interval of the accepted population. The suggested rate should be two to five times greater (Anderson, 2019). The final step is to fit a new model with the aggregated dataset (Mancisidor, et al., 2020).

2.2.1.6. Reclassification

Reclassification method adds to the observed bad sample, from the accepted applicant sample, a subset of rejects which have been scored by the estimated AR model as bad. No rejects scored as good are added. The rejects identified as bad are assigned a weight so that their contribution to the total bad sample is controlled (Luca & Desjardins, 2018). It is important to ensure that the inferred bad instances do not account for more than the observed bad instances. 30% is an advisable proportion of inferred to the total.

$$p = \frac{wx}{D_0 + wx'}$$

Equation 4 - Reclassification Proportion

Equation 4 notates this p proportion, where D_0 represents the total observed bad instances, x represents the number of rejected bad instances, and w is the calculated weight. This can be rearranged to give us the weight required to achieve the proportion desired:

$$w = \left(\frac{p}{1-p}\right) * \left(\frac{D_0}{x}\right)$$

Equation 5 - Reclassification Weight

The next step is to build a model with the combined weighted inferred bad sample and the accepted sample. This model should now have learned characteristics of the rejected applicants.

2.2.2. Machine Learning

Previous research has investigated the ability for machine learning models to use the rejected application sample and infer their performance to create a more representative dataset for future models to be trained on.

First and foremost, to ensure effective and comparable evaluation of the various machine learning reject inference techniques, the hyperparameters for each algorithm needs to be carefully set (Shen, et al., 2020). A grid search method was employed for hyperparameter optimisation by several studies (Li, et al., 2017; Shen, et al., 2020). In addition, the training and test samples used for validation need to be same. To evaluate models, Li, et al. (2017) repeated the experiment 50 times, resampling the dataset each time with replacement.

In the following sections, we will describe in detail Bayesian network, semi-supervised support vector machine, three-stage framework, and ensemble learning framework.

2.2.2.1. Bayesian Network

A Bayesian Network (BN) is a probabilistic graphical model for representing knowledge that consists of two parts. The first part is a directed acyclic graph (DAG) in which nodes represent characteristic variables and directed edges between nodes represent conditional dependencies of the random variables (Anderson, 2019). No loop or self-connection is allowed. This is referred to as the BN network structure. The second part is a set of conditional probability distributions (CPD), one for each node conditional on its parents. The BN is completely determined by both these parts (Anderson, 2019).

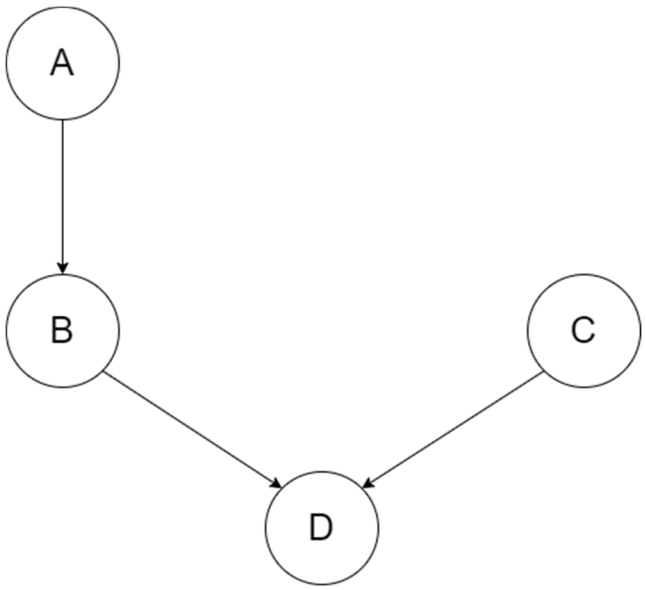

Figure 1 - Directed Acyclic Graph

For illustrative purposes, Figure 1 shows a basic DAG. Nodes A and C are independent of each other, as are B and C, thus if C occurs, our knowledge of A or B does not change. Node B is dependent on A since it is a child node and can be notated as P(B | A). D is similarly a child node and depends upon B and C. D is also considered conditionally independent of A which means if B occurs, A is now irrelevant to D. The notation for this relationship is P(D | B, A) = P(D | B).

The structure of the BN will be determined using the data from the accepted applicants (Anderson, 2019). Anderson (2019) implemented a score-based approach, which utilises the Bayesian Information Criterion (BIC), to optimise the network structure of the BN on the given data. The rejected applicants are then applied to the trained BN and scored. If the score is below a predetermined cut-off, it will be classified as bad otherwise it will be classified as good. To determine this cut-off value, Anderson (2019) used the default rate of the accepted-only credit datasets they had employed for their study. Their previous research confirmed this method as industry standard for assessing reject inference techniques.

The results of study by Anderson (2019) provided evidence of the better performance of the classifier when utilising a BN as a reject inference technique. Although, we note, that the study does not provide a benchmark model not fitted on a reject inferred dataset to compare the performance against.

2.2.2.2. Semi-supervised Support Vector Machine

Li, et al. (2017) proposed a semi-supervised support vector machine (S3VM) model to solve the reject inference problem and found it to improve the performance of the classification model implemented. SV3M is an extension of the support vector machine (SVM) machine learning model. This classifier separates the accepted and rejected instances in a multi-dimensional space that maximises the margin between two optimal hyperplanes. The hyperplane traverses through non-density regions of rejected applications allowing it to appropriately classify (Mancisidor, et al., 2020). In semi-supervised learning some of the instances in the dataset are labelled but not all, thus suitable for solving the reject inference problem. The model will learn hidden information from the rejected applicants. This advantage over the traditional, supervised SVM and logistic regression (LR) models, trained on only accepted applicants' information, was proven in the results with an overall accuracy of 91.4% compared to 89.1% (SVM) and 78.7% (LR).

2.2.2.3. Three-Stage Framework

Shen, et al. (2020) proposed a three-stage reject inference learning framework, Figure 2 - Three-Way Stage Framework (Shen, et al. 2020), which does not require any prior data distribution assumptions. The first stage is the three-way decision (3WD) theory which divides the rejected samples into positive, boundary and negative regions. This stage is important to handle the negative transfer problem, the inclusion of irrelevant negative samples.

The second stage is a special unsupervised transfer learning technique called self-taught learning (STL). The rejected samples in the positive region and the accepted sample are combined to train the STL model. The STL method only utilises the data's features, the label is disregarded, and that is why it is considered an unsupervised learning method. In practice, this stage assumes that inputs x_u and x_l are vectors for the rejected (unlabelled) and accepted (labelled) application samples. The STL method uses x_u to detect the inherent elements that comprise the credit sample. For example, it may identify certain strong correlations between the features, and therefore learn that most samples have many latent, unobserved, correlations. From this, it then learns to represent the samples in terms of the feature correlations rather than the raw feature values. The sample representations from the latent correlations act as a higher, more abstract, representation of the input. When this learned representation is applied to the accepted credit sample, x_l, a higher-level representation of the accepted credit samples is also attained, making it an easier supervised learning task (Shen, et al., 2020).

The third, and final, stage is the binary classification which is trained on the reconstructed training data features.

To evaluate the proposed framework, Shen, et al. (2020) verified the applicability with a personal credit dataset. Their proposed model outperformed the traditional statistical methods parcelling, augmentation and the S3VM (Li, et al., 2017) model.

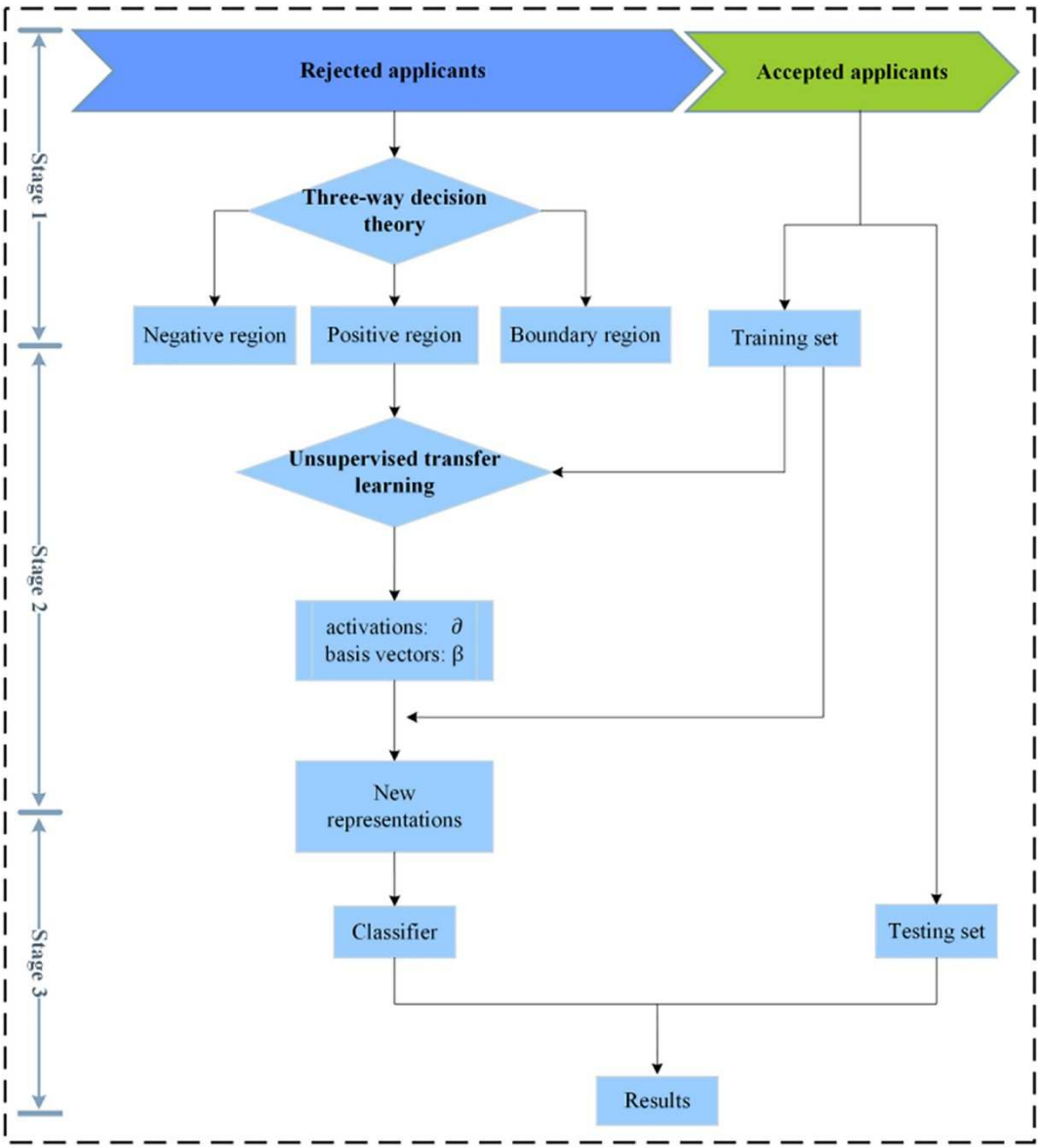

Figure 2 - Three-Way Stage Framework (Shen, et al. 2020)

2.2.2.4. Ensemble Learning Framework

Liu, et al. (2020) introduced a novel approach for addressing the reject inference problem which combines the unsupervised learning clustering algorithm with a supervised classifier and is then integrated into a global semi-supervised learning called ensemble learning framework, referred to as SSL-EC3. Similar to the S3VM technique (Li, et al., 2017), by using clustering methods, hidden relationships can be explored between accepted and rejected samples. In this work, an

EC3, ensemble classifier that merges both classification and clustering techniques (Chakraborty, 2017), model is built based on the accepted sample. The base classifiers (LR, k-Nearest Neighbour, SVM, decision tree and RF) and base clustering algorithms (KMeans) split the samples into two classes. The global SSL-EC3 receives these initial labels for the rejected applicants and uses the unlabelled samples to correct the classification. Each fusion of base classifier and base cluster algorithms showed a more stable and improved performance over an individual classifier. The RF and KMeans cluster ensemble were the standout performers.

2.2.3. Discussions on Reject Inference

There is a broad spectrum of opinions on the benefits of reject inference in the development of models: those who think that it is a vicious circle, where inferred performance of the rejects would be based on the approved but biased population, exaggerating the problem, and those who advocate the methodology as a valuable approach that benefits the model's performance and reduces selection bias. The former school of thought advocates that all pertinent information for modelling resides within the known performance of the accepted applications, and thus there is no use for the rejects, no matter the proportion size (Luca & Desjardins, 2018; Crook & Banasik, 2004).

Statistical methods which utilise augmentation and extrapolation are popular methods for addressing the MAR problem (Liu, et al., 2020). Despite their popularity, there are arguments against their benefits. Firstly, Shen, et al. (2020) and Anderson (2019) argue that the underlying assumption of these methods, Equation 1, that the rejected samples share the same repayment behaviour as the accepted sample may be untrue and is difficult to verify. Secondly, the statistical reject inference techniques which utilise thresholds, such as augmentation and extrapolation, require the modeller to have knowledge of the defined configuration value. This ambiguity detracts from the applicability of these techniques as it implies knowledge of the bad rate of the rejected population which is, generally, unknown. In addition, in general, the higher the cut-off score in the original AR model, the greater the deterioration in performance caused by augmentation which detracts from the applicability of the technique (Crook & Banasik, 2004). Finally, Anderson (2019) noted that there is an inherent problem in techniques which utilise

extrapolation. It is also not considered appropriate to train a supervised classification model using the accepted data to predict the probability of the rejected samples which extrapolation techniques do. The assumption that the distribution of characteristics for the rejected applicants is similar to the accepted is risky and does not hold true (Henley, 1995). Crook & Banasik (2004) study showed that the extrapolation technique results were virtually identical to a model trained on accepts-only and noted *"extrapolation appears to be both useless and harmless"*.

Ehrhardt, et al. (2021) results showed that not one statistical reject inference technique significantly outperformed another and noted that the methods are highly dependent on the data itself and / or the proportion of unlabelled observations. They concluded that due to the time-consuming task of implementing reject inference, the offsetting benefit is negligible unless there is significant information on the rejected applicants. Although, Henley (1995) argues that even if reject inference only adds a marginal benefit to the performance of a model, even the slightest improvement is a worthwhile endeavour because of reducing the rate of bad results in savings for the business.

If information loss due to selection bias is significant and cannot be neglected, reject inference must be recommended as a technique to resolve the imbalance (Ha-Thu, 2016). As highlighted in Section 2.2, reject inference uses the characteristics of the rejected population in the building of subsequent models to diminish the bias, and thus improve the predictive accuracy on the whole TTD population (Ehrhardt, et al., 2021).

Henley (1995) summarises that the reliability of reject inference is conditional on the availability of additional information, such as credit bureau information, or accepting the assumptions made, as we have spoken about.

2.2.4. Summary

Reject inference as a concept has been introduced to address the missing data problem. We have discussed the various statistical and machine learning methods that have been proposed in the literature.

From our literature review, it has been observed that statistical reject inference methods have a negligible benefit and can even produce inferior classification performance results for models (Banasik & Crook, 2007; Mancisidor, et al., 2020; Luca & Desjardins, 2018; Crook & Banasik, 2004). Whereas more sophisticated machine learning methods proposed exhibit increased model performance over statistical methods and make fewer presumptions (Anderson, 2019). Nevertheless, it is worthwhile implementing statistical methods to obtain a natural baseline to compare more advanced methods to (Henley, 1995).

2.3. Evaluation Metrics

2.3.1. Overview

To assess the performance of the classification model and conduct a comparative analysis, evaluation metrics are necessary. In the following section, we will present performance indicators used in the literature researched to analyse results.

2.3.2. Model Evaluation

To evaluate the accuracy of our classification models, the area under the curve (AUC) has been used extensively in prior research (Luca & Desjardins, 2018; Dastile, et al., 2020; Turiel & Aste, 2020). It is based on the receiver operating characteristic (ROC) curve and varies between 0 and 1 (Shen, et al., 2020). The ROC curve is a graph of the true positive rate (sensitivity), which is plotted on the x-axis, and the false positive rate (1-specificity), which is plotted on the y-axis (Anderson, 2019). Sensitivity and specificity are derived from the confusion matrix which consists of *True Positives (TP), True Negatives (TN), False Positives (FP)* and *False Negatives (FN)* (Dastile, et al., 2020). The confusion matrix is considered a useful tool to gauge the classification accuracy (Li, et al., 2017; Liu, et al., 2020).

Sensitivity, also called recall, is the fraction of correct positive predictions and can be calculated as per Equation 6. It is essentially the percentage of the actual positives the classifier was able to identify. This is a key metric for our models as predicting bad instances is more important than misclassifying a good instance (Turiel & Aste, 2020).

$$\frac{TP}{(FN + TP)}$$

Equation 6 - Recall

Specificity is the proportion of actual negative cases correctly identified and can be calculated as per Equation 7. It can be referred to as the recall of the negative class.

$$\frac{TN}{(TN + FP)}$$

Equation 7 – Specificity

Now that we have provided background, we can delve into the AUC scoring metric further. The AUC can be interpreted as the probability a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (Turiel & Aste, 2020). If the model's AUC result is or close to 0.5, it would be fair to describe the model as random and not intelligently classifying (Liu, et al., 2020). Closer to 1 is encouraging and is accurately classifying instances. Luca & Desjardins (2018) expects the AUC of a classifier to be in the range of 0.70 to 0.85, anything above this would usually indicate the model has overfitted to the data. We expect the AUC to be similar between the training and test samples, again to confirm overfitting has not occurred. Overfitting indicates a model cannot generalise to new data. Equation 8 represents the calculation for the AUC.

$$AUC = \frac{1}{2}\left(1 + \frac{TP}{TP+FN} - \frac{FP}{FP+TN}\right)$$

Equation 8 - Area Under the Curve

Dastile, et al. (2020) systematic review of literature in credit scoring determined that the most commonly used evaluation metric was *Percentage Correctly Classified (PCC),* Equation 9. Banasik & Crook (2007) presented their models' performances with this metric.

$$PCC = \frac{(TP + TN)}{(TP\ +\ FP\ +\ FN + TN)}$$

Equation 9 - Percentage Correctly Classified

Ehrhardt, et al. (2021) in their comparison of the performance of several reject inference techniques used the mean Gini index of their experiments. It measures risks differentiation ability of a given model (Liu, et al., 2020). The Gini index would measure the area between the cumulative frequency distribution of the bad applicants on the number of good applicants and the random distribution curve. A higher Gini score indicates that a model has a better ability to distinguish risks. In other words, a model with a higher Gini score means that the model can more effectively eliminate bad applicants (Liu, et al., 2020). Ha-Thu (2016) advises a score greater than 35% is considered satisfactory.

In addition to the previously mentioned AUC statistic, Shen, et al. (2020) employed the Kolmogorov-Smirnov statistic (KS), Equation 10. KS measures the maximum distance between the cumulative score distribution of bad $P(s|B)$ and good $P(s|G)$ (Dastile, et al., 2020; Ha-Thu, 2016). Similar to the Gini index, KS indicates the strength of a model to differentiate good applicants from bad. The higher the KS score, the better. Similarly, Ha-Thu (2016) advises a score greater than 35% is considered good.

$$KS = \max\ |P(s|G) - P(s|B)|$$

Equation 10 - Kolmogorov-Smirnov Statistic

2.4. Summary

We have discussed several metrics that are presented in the literature. For us to quantify the performance our models with and without reject inference, we will need to present similar statistics. We will discuss within our methodology, Section 5.6, those that we will be utilising.

3. Risk Analysis

Here we will discuss any potential risks to the project. It is important these are presented beforehand so that we can factor in delays and disruptions into our schedule, dampening their effect on the project.

Risk Description	Likelihood	Impact	Resolution
Data Availability	Low	High	Alternative data must be sourced, possibly synthesise own dataset
Insufficient time to complete project	Medium	High	Reduce project objectives
Insufficient computing resources	Medium	Medium	Access university resources
Onboarding with industry partner unsuccessful	Low	High	Pivot project to open data and build own models
Student is ill	Low	Low	Mitigating circumstance can be applied for.
Project plan changes	High	Low	Rearrange project to better suit new plan

Table 2 - Risk Management

4. Discussion of Professional, Legal, Ethical and Social Issues

4.1. Professional and Legal Issues

The project will follow and respect the British Computer Society (https://www.bcs.org/) Code of Conduct. Foremost, we will work to the upmost professional competence and integrity. All papers, code, or libraries will be referenced as appropriate and used under the terms of the publisher licences.

An employment contract has been signed with Royal London which includes protection of company's intellectual property and confidentiality. As part of The Data Lab MSc scholarship, we will be expected to work to the highest professional standard.

The dataset used is available on a public GitHub repository (https://github.com/jeremyDT/P2P-lending-with-AI). No personal or sensitive data is available in the chosen dataset.

4.2. Ethical and Social Issues

Potential for real-world models, with reject inference implemented, to potentially learn the characteristics of fraudulent and malicious credit applications. Models will then unfairly evaluate new applications with erroneous coefficients.

Another consideration is if it is ethically and socially acceptable for machines to decide on credit applications. Machines have no empathy towards a person's situation.

5. Methodology

5.1. Introduction

In this chapter, we will discuss the methodology of our research. We will describe the dataset, models, reject inference techniques and metrics that have been chosen to be developed and evaluated.

5.2. Development Environment

We opted to carry out our research in the programming language Python (https://www.python.org/), version 3.8.8, as it gives us access to several useful packages. The packages we have used can be viewed in Table 3. We used a Jupyter (https://jupyter.org/) notebook for our development environment.

Package	**Version**
imbalanced-learn	0.9.1
joblib	1.0.1
matplotlib	3.5.0
numpy	1.19.5
pandas	1.2.4
pgmpy	0.1.19
scikit-learn	1.1.1
seaborn	0.11.1

Table 3 - Python Packages

5.3. Dataset

To carry out our research, we have utilised an open dataset analysed in previous academic research and made available by Turiel & Aste (2020). The dataset contains information on Lending Club platform's P2P lending applications and accepted loans' performance from the period 2007 to 2017. P2P lending is defined as the act of lending to individuals or businesses through a centralised, online platform that matches lenders with borrowers (Turiel & Aste, 2020). The attractiveness for both parties is the lower transaction costs and availability of counterparties. For the lender, the return on investment is greater than a traditional savings account, although, with higher risk of loss of investment. For the borrower, loans can be drawn for a variety of reasons such as credit card or home related loans.

The data has been split into two datasets, one contains loan applications, with a target variable of rejected or accepted, and the other contains information on those loans that were accepted, with a target variable of charged-off or not. Charged-off usually means the account has been closed because the borrower has defaulted on their obligations. There are ≈15.5 million instances in the loan application dataset and ≈800,000 in the loan status dataset. There are ≈14.7 million rejected instances and the remaining are accepted, this equals the total instances in the accepted loan status dataset. This means that there is an acceptance rate of 5.24% and thus a reject rate of 94.76%. Of the accepted loans, 79.37% have been fully paid and 20.63% have defaulted.

	Loan Application	**Accepted Loan Status**
Features	6	21
Instances	15,554,529	814,643

Table 4 - Dataset Statistics

Originally the application dataset had 9 features and the accepted loan dataset had 150 features, but these were reduced by Turiel & Aste (2020) for their analysis to 6 features for the former and 21 for the latter. This was carried out to reduce the features to those only shared by both datasets. Turiel & Aste (2020) acknowledged that there is likely to be a loss of information for the models by removing features.

Features	Loan Application Dataset	Accepted Loan Dataset	Data Type
Debt to Income Ratio	✓	✓	Numerical
Employment Length	✓	✓	Numerical
Loan Amount	✓	✓	Numerical
Loan Purpose	✓	✓	Categorical
Loan Issue Date	✓	✓	Date Time
Term		✓	Numerical
Instalment		✓	Numerical
Home Ownership		✓	Categorical
Income Verification Status		✓	Categorical
Earliest Record Credit Line		✓	Numerical
Number of Open Credit Lines		✓	Numerical
Derogatory Public Records		✓	Numerical
Revolving Line Utilisation Rate		✓	Numerical
Total Number of Credit Lines		✓	Numerical
Number of Mortgages		✓	Numerical
Number of Bankruptcies		✓	Numerical
Logarithm of Annual Income		✓	Numerical
FICO Score		✓	Numerical
Log of Credit Revolving Balance		✓	Numerical
Application Type		✓	Categorical

Table 5 - Dataset Features

5.4. Model Development

In the following sections, a description of the models chosen for classifying the samples will be provided. The purpose of this section is not to extensively analyse the benefits, flaws, or suitability of each model, but simply to provide sufficient background and understanding.

5.4.1. Application Classifier

In line with the results of Turiel & Aste (2020), we will implement a binomial logistic regression (LR) algorithm for classifying applications into dichotomous classes, accept or reject. LR estimates the probability of a binary outcome based on a set of independent variables. Independence between the predictor variables is a key assumption of the model.

The model is trained on a set of feature vectors, $x^{(i)}$, and their classifications, $y^{(i)} = 0\ or\ 1$, by finding the set of parameters that minimises the difference between the predicted label, $\hat{y}^{(i)}$, and the actual label, $y^{(i)}$. The model parameters are the components of a vector, *w,* and a constant, *b,* which relate a given input feature vector to the predicted logit, the function associated with the standard logistic distribution, *z*. The hypothesis is provided in Equation 11.

$$z = w^T x + b$$

Equation 11 - Logistic Regression Hypothesis

The sigmoid function, Equation 12, then takes the output, *z*, and maps the real values into bounded valued between 0 and 1, a probability estimate. The probability estimate is then mapped to a class depending on a threshold. This threshold can be set to any value between 0 and 1.

$$S(z) = \frac{1}{1 + e^{-z}}$$

Equation 12 - Sigmoid Activation Function

5.4.2. Accepted Loans Classifier

For our analysis, we must also build a model that will be able to classify the accepted loans into good (non-default) and bad (default) classes. From our research, the favourable model for tabular datasets is ensemble models based on decision trees (Grinsztajn, et al., 2022). Other research has proposed deep learning architectures, such as a deep neural network, for classification and have shown promising results (Turiel & Aste, 2020). Although, for our study, we have opted to

implement a random forest. The research by Grinsztajn, et al. (2022) evidenced that a random forest model built for classification of tabular data had stable and high accuracy performance over several random search iterations.

A random forest is a collection of unpruned decision trees and is an extension of the bootstrap aggregation. Bootstrapping involves selecting random samples from the training dataset with replacement and fitting a decision tree on each. The advantage of ensemble methods, over an individual model, is the reduction in bias and variance in predictions. The main feature of a random forest is that not all the data characteristics are used and instead a random subset is chosen for each bootstrap sample. This increases the independence of each decision tree in the forest as the information observed is different. The predictions from each decision tree are aggregated and, in the case of classification problems, the majority vote generates a single prediction (Chen, et al., 2004). The benefit of this method is that there is generally better performance than a single well-tuned decision tree.

5.5. Reject Inference Techniques

In the following sections we will describe the reject inference techniques we will implement in our study, as guided by our review of the current literature. Our intention was to study all techniques researched, although, as discussed in Section 9, we have instead selected every statistical technique except reclassification and only the machine learning technique Bayes Network for our research.

5.5.1. Simple Assignment

For the reject inference technique simple assignment, we will assign all rejected instances in the application dataset to the default (Charged Off = 1) class. The classified rejected instances will then be appended to the accepted loan dataset with common features, alternatively, reduced features. The synthesised dataset will then be fitted to the relevant GB model.

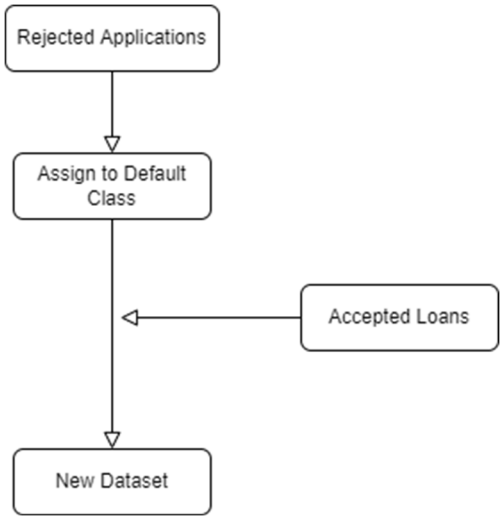

Figure 3 - Simple Assignment Process Flow

5.5.2. Random Assignment

For random assignment, we need to calculate the percentage of default instances in the accepted loan dataset. The result is then multiplied to set a higher rate of default to the rejected instances. Based on this calculated rate, the rejected sample is randomly split with the portion assigned to the default class given the relevant class label and the remaining sample assigned to the non-default class. Both portions and the common feature accepted loan dataset is then concatenated and fitted to the relevant GB model.

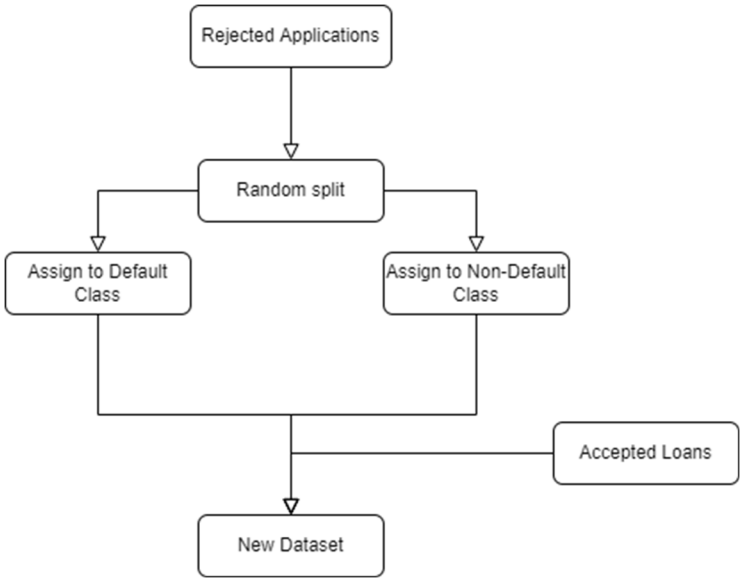

Figure 4 - Random Assignment Process Flow

5.5.3. Augmentation

Augmentation design was guided by (Banasik & Crook, 2007), as discussed in Section 2.2.2.2. The application dataset, including both accepted and rejected instances, is sorted by the probability score generated by the estimation of the AR model. The dataset is then split into intervals and for each interval the weight is calculated using the formula in Equation 2. The rejects are then dropped from each interval as it is only the accepts which we are augmenting. This is important as it allows us to disregard the rejects and we can utilise the full feature accepted loan dataset. Finally, we concatenate all the intervals together, drop any duplicates and merge with the accepted training dataset. We drop duplicates as we have no use for multiple instances as the merge will apply the relevant information to any instances that match the accepted loans in the application dataset.

The weighted accept-only dataset can then be fitted to the full feature model and the common feature model.

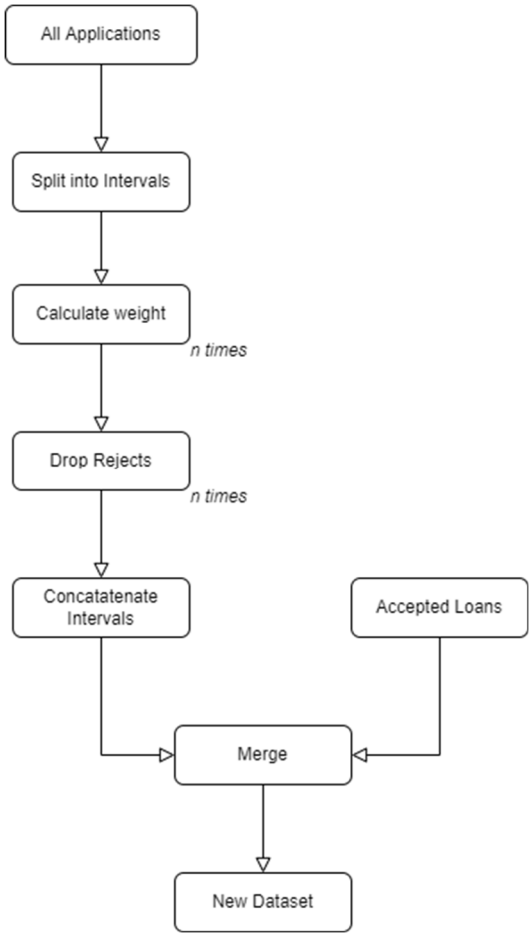

Figure 5 - Augmentation Process Flow. n times indicates the number of intervals and repetition of process step

5.5.4. Parcelling

Parcelling initially follows a similar process flow to augmentation but also includes a similar function to assignment. The application instances are sorted by score and split into two new subsets, rejected and accepted. Both subsets are then split into intervals. By sorting by score initially, each interval should have similar scores, regardless of its application status. The default rate is retrieved from the first accepted loan interval, factored and then the rejected interval is split on this new rate. All those samples within the reject split are assigned to the default class and the remaining instances are assigned to the non-default class. Finally, the rejects, both default and non-default samples, and the accepted loan dataset is concatenated. This process is repeated for each interval.

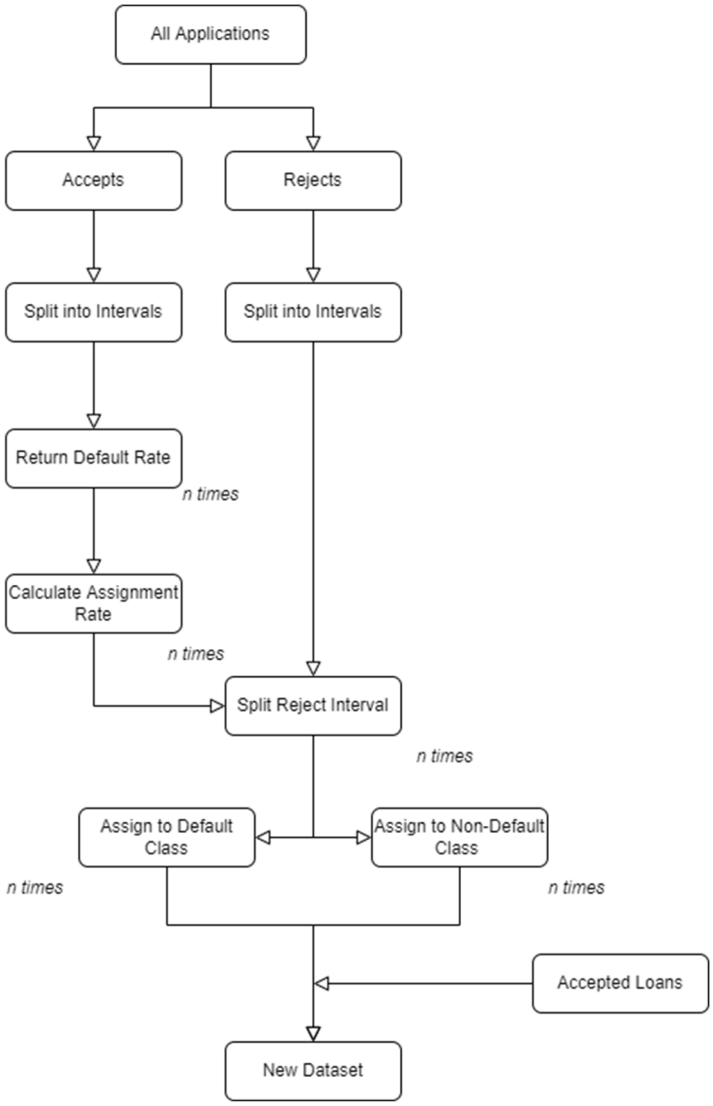

Figure 6 - Parcelling Process Flow. n times indicates the number of intervals and repetition of process step

5.5.5. Extrapolation

Extrapolation utilises the model fitted on the accepted loans to predict the default class probability for the rejected applications. We then find the actual default rate (18.30%) in the accepted loan dataset and use this rate as the threshold for the rejected sample's class assignment, as advised by Anderson (2019). Any instance probability above or equal the rate will be assigned to the default class and any below will be assigned to the non-default class. The accepted loan dataset is then appended to the now classified rejected dataset.

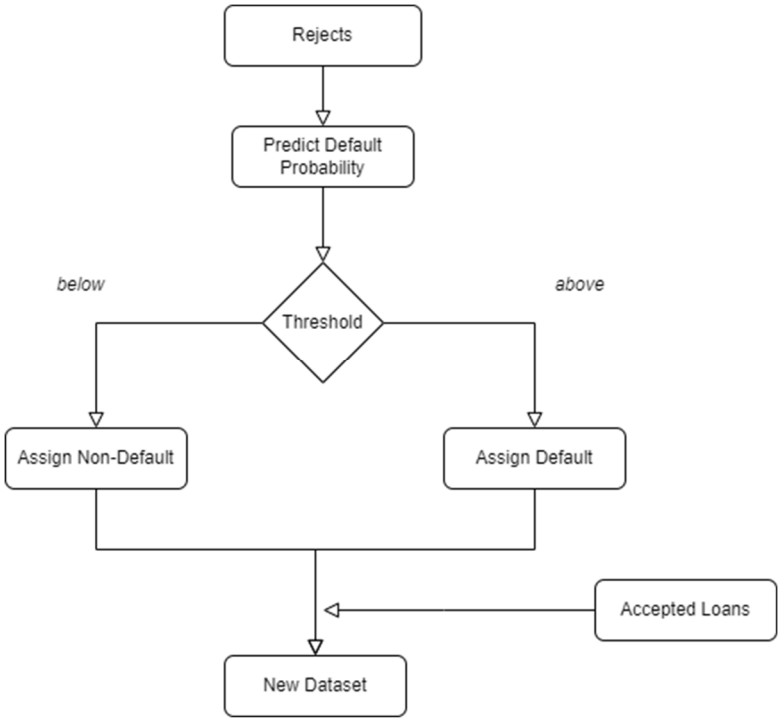

Figure 7 - Extrapolation Process Flow

5.5.6. Fuzzy Augmentation

The process for fuzzy augmentation begins with duplicating the rejected sample. The sample of rejects that will be assigned to the non-default class are assigned the complement of the score from the AR model as its weight. The sample of rejects that will be assigned to the default class are assigned the AR score as its weight. The accepted loans are assigned a weight of 1. The synthesised dataset is then created from the duplicated rejected samples and the accepted sample.

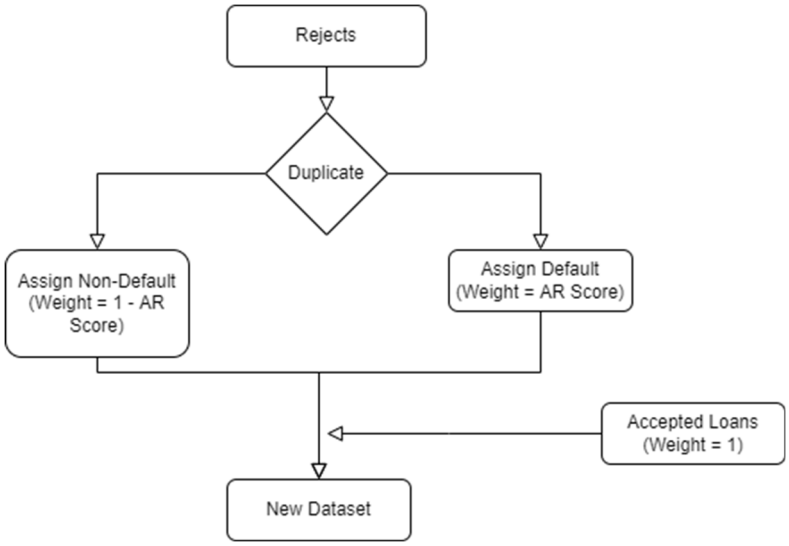

Figure 8 - Fuzzy Augmentation Process Flow

5.5.7. Bayesian Network

The BN methodology design has been strongly influenced by Anderson (2019). The first requirement of the BN setup is to bin the numerical features of the accepted loan dataset. We then fit the shared, numerical features in the reject sample on the defined bins.

We will exhaustively search for the optimal DAG structure for the training dataset. Each possible DAG is fitted on the data and the structure returning the lowest BIC score will be implemented.

The optimal structure is then provided to the BN and the CPDs for each variable in the given training data can be estimated.

Now the BN has been built, we can feed in the pre-processed rejected instances to determine the probability of default. Those rejects below the threshold, the default rate of the accepted loan dataset (Anderson, 2019), will be assigned to the default class, otherwise the non-default class. A new dataset is then synthesised with the joining of the inferred performance of the reject sample and the accepted loan dataset, ready to be fitted into the AR model.

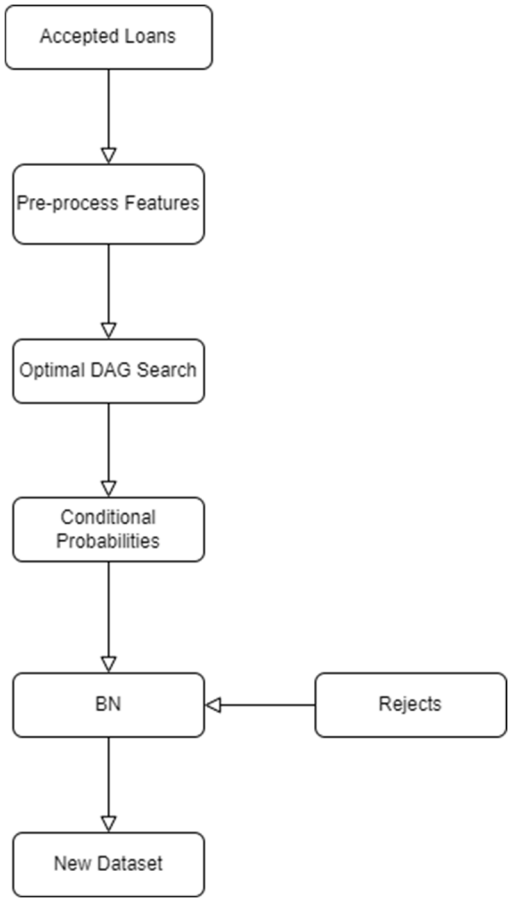

Figure 9 - Bayesian Network Process Flow

5.6. Results

To analyse and compare the performance of our models with and without reject inference, we must report metrics. For our model's performance evaluation, we have selected to report the confusion matrix, AUC, PCC, specificity and recall metrics. Each iteration of our model will be evaluated on the same test set, as mentioned in Section 2.2.2, for fair comparison of each iteration of our model. We will also report the mean 10 cross-validated, explained in Section 0, AUC value of the training set. This will indicate if an iteration has overfitted to the training dataset. The AR model fitted on the known performance of the accepted loans only will be defined as our benchmark model to compare with as this would be the status quo.

6. Implementation

6.1. Data Pre-processing

After describing the datasets in Section 5.3, we should now proceed to discuss the pre-processing required so they are in a suitable condition to be efficiently analysed by the models. Section 6.1.1. will discuss the steps undertaken for the Lending Club loan application dataset. Section 6.1.2 will discuss the Lending Club accepted loans dataset.

6.1.1. Lending Club Loan Application Dataset

For the Lending Club loan application dataset, the first step is to remove any applications that were after 01/01/2016. Turiel & Aste (2020) in their analysis noticed that a significant of defaults had not been reported yet, see Figure 10 (Turiel & Aste 2020), and their status remained good. Without this filtering, bias would be introduced if they remained in the dataset. Although, this may not appear to apply to the application dataset as the class labels are rejected or accepted, with no relation to default, it is important that the datasets are aligned as we do not want the models to learn information about instances which do not exist in the accepted loans dataset. The impact on the dataset is large as 9,203,528 (59.17%) instances are removed. Subsequently, we reduce the application dataset further to remove any instances that are beyond the latest date in the accepted loan training dataset. The reason we do this, similar to the previous filtering, is so that when we are including the rejected sample in the training set, the models are not learning prior knowledge of samples in the test set, referred to as information leakage (Turiel & Aste, 2020). The impact on the dataset is a reduction of 2,201,655 (34.67%). The final count of instances in the application dataset is 4,149,346 instances.

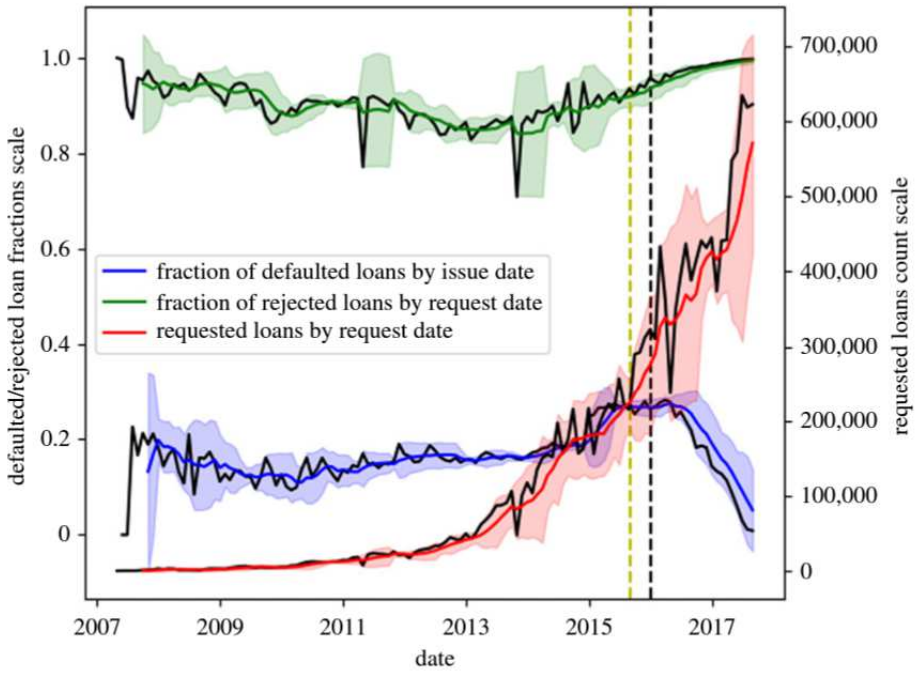

Figure 10 - Time Series Plot of Datasets. Data on the right of the vertical black dotted line was excluded from the analysis (Turiel & Aste 2020)

The next step is to consider the Loan Purpose column as it is the only categorical type in the dataset. There are 69,723 unique values in the Loan Purpose column but only 24 that have over 19,000 cases each. This threshold was specified by Turiel & Aste (2020) but there was no explanation provided for setting the threshold to 19,000. From our analysis, the value threshold could have been set lower at 15,000, for example, with the same outcome. Nevertheless, the impact on the sample count is a loss of 217,598 cases (5.24%). Within those 24 unique values there are several duplicates, for example 'debt_consolidation' and 'Debt consolidation'. After merging, we have 13 unique values. Finally, we create a dummy binary variable for each unique value in the Loan Purpose category. This is a necessary step as machine learning models struggle to interpret categorical variables. We reverse the dummy variables once the model has been built.

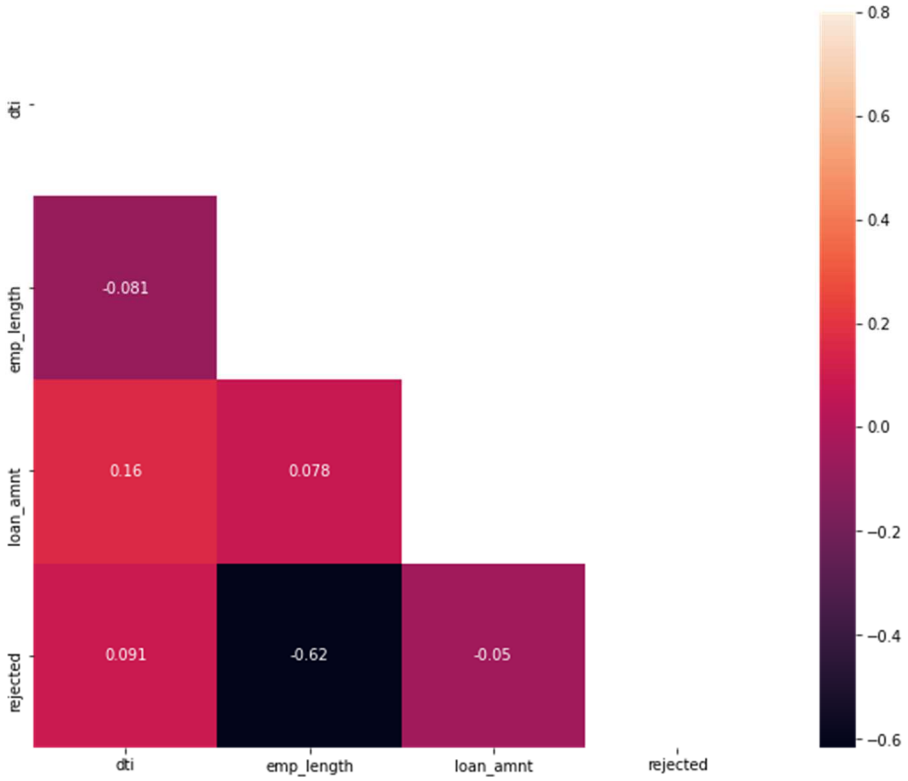

Figure 11 - Application Dataset Feature Correlation Map

We did not split the dataset into training and test as the only use for this model is to score each application. In reality this would be an old model which would either be replaced or retrained with the GB model. There are a total of 3,931,748 instances, 3,432,231 (87.30%) are rejected.

Finally, we setup a pipeline, to streamline the additional pre-processing of the data and the training of the model. The benefit of implementing a pipeline is the pre-processing is built in and will not alter the data. We will require the application dataset in its original format in later stages of the study. The first step in the pipeline is to impute null values. It is important to handle null values as models cannot interpret missing values otherwise. There exist 65,291 null values in the Employment Length column. We imputed the mean value for the missing values. We chose this option for imputation instead of more sophisticated techniques such as the k-Nearest Neighbour (KNN) or Multiple Imputation by Chained Equations (MICE) as the impact on our analysis should be limited as it effects only 1.66% of the instances. Next step is to standardise the numeric features by removing the mean and scaling the unit variance. To expand on the latter,

this means dividing all values in the column by the standard deviation. Machine learning models may behave poorly if individual features are not normally distributed.

6.1.2. Lending Club Accepted Loans Dataset

The pre-processing for the accepted loan dataset followed a similar approach to the application dataset. The accepted loans dataset is filtered to remove all loans after 2016, refer to Section 6.1.1. Turiel & Aste (2020) ring-fenced the most recent loans to test the performance of the models developed, while the earlier loans were used to train. Their reasoning for this strategy was to replicate the human process of learning by experience. Therefore, we split the dataset into training and test subsets by the date of application (Loan Issue Date). The oldest 80% are assigned to the training set and the remaining 20% to the test set. The test set remained the same for all iterations of evaluation. This is important so each model observes the same unseen data, otherwise our results would be incomparable and meaningless. The Loan Issue Date category is then dropped from both sets as it is irrelevant to the model.

To align the possible values in the Loan Purpose category in both datasets, we removed the instances that had ‘educational’ and ‘renewable energy’ as a value in this category. The reason why this value is not available in the applications dataset is because the number of instances is below the arbitrary 15,000 threshold, as explained in Section 6.1.1. The impact of this removal is minimal as there are only 323 (0.0005%) ‘educational’ instances and 467 (0.0007%) ‘renewable_energy’ instances. We are confident this will not impact our results and findings.

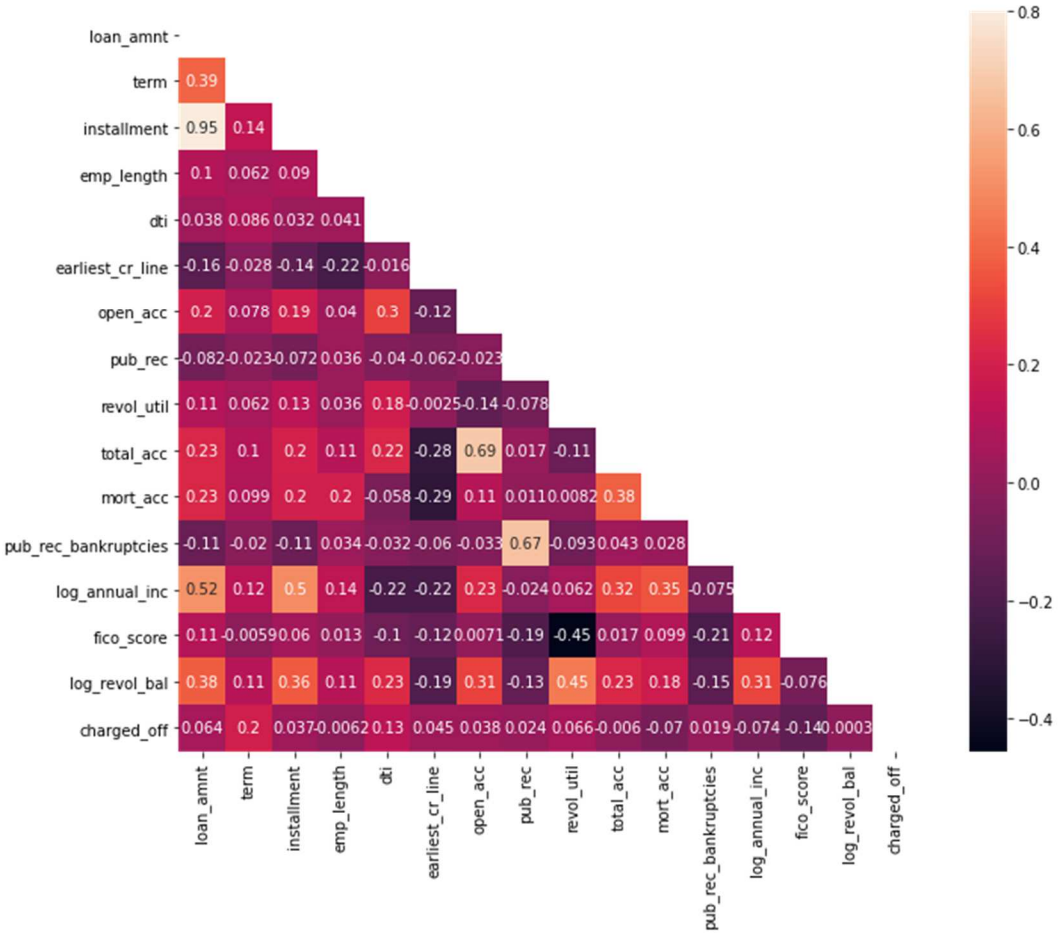

Figure 12 - Accepted Loans Dataset Feature Correlation Map

Similar to the applications dataset, we must create dummy variables for the categorical columns, see Table 5. We had to manually add an additional dummy variable column into the training set for the Application Type value 'Joint' as there are no instances that exist in the training set but there does in the test set. All values for this synthetic column were set to zero. Without this manual intervention, models would not be able interpret this feature when utilising the test set to evaluate.

The next step is to treat the NaN (not a number) values so that our models can perform on our dataset. For imputing the missing values, we have opted to fill them in with the mean value of the column, similar to the application dataset. While this method is better than simply removing the instances with missing values, it would not be our preferred method as there are more sophisticated methods such as KNN and MICE. Although, these latter methods require far more

computational resources than is available to us. There are total of 77,369 (12%) of instances in the dataset which have at least one missing value. This may appear significant and lead to bias but for individual features, the impact is less than 8% and thus imputation should not invalidate our analysis.

We scaled all the numerical columns using the same process for the application dataset, as described in Section 6.1.1. Although, as we split the dataset, the scaler is fitted on the training set alone and then applied to both the training and test sets, no information about the test set is contained in the scaler which could be leaked to the model (Turiel & Aste, 2020). The same scaler is applied to the rejected sample whenever a technique calls for it.

The breakdown for the accepted loans training set is a total of 499,517 instances, 18.30% are defaults. This matches exactly with the application dataset as there are 499,517 rejected instances thus, we are confident the pre-processing steps taken has aligned the datasets. The test set contains 146,571 instances, 26.79% defaults.

6.2. Model Development

In the following sections we will discuss the development of both classification models we have built and provide justification for the settings applied.

6.2.1. Accept / Reject Model

For the logistic regression model, built to classify applications into accept and reject classes, we have used the sklearn model LogisticRegression. The non-default hyperparameters are defined in Table 6.

Parameter	**Setting**
Class Weight	Balanced

Max Iterations	1000
Tolerance	0.001
Warm Start	True
Solver	Broyden–Fletcher–Goldfarb–Shanno

Table 6 - Logistic Regression Selected Hyperparameters

We performed a 10 cross-validated exhaustive grid search, like previous research, over the solver hyperparameter setting and, for our dataset, the Broyden–Fletcher–Goldfarb–Shanno algorithm was optimal. The solver parameter is the chosen algorithm used to optimise the problem. The remaining hyperparameters we did not optimise as these were selected purposely. For the class weight parameter, this was set to be balanced which adjusts the weights inversely proportional to the class frequencies thus making it cost sensitive and penalises misclassification of the minority class (Chen, et al., 2004). This is important as our dataset is highly imbalanced. Max iterations defines the number of iterations that the solver can take to converge. We have set it high to not constrain results. Tolerance defines the stopping criteria for the algorithm. Warm start has been set to true which allows the model to reuse the solution from the previous call.

Now that the AR model has been estimated we can retrieve the probability score for each instance in the application dataset.

6.2.2. Good / Bad Model

For our implementation of a random forest classifier, we have used the BalancedRandomForestClassifier which is available in the imbalanced-learn Python package. The special feature of this random forest is it randomly undersamples the majority class in each bootstrap sample to balance the number of classes before it grows each tree (Chen, et al., 2004). It is important that we handle skewed class distributions as there is a significant chance, due to the stochastic nature of the algorithm, that a bootstrap sample contains few or zero instances of the minority class, resulting in poor performance in predicting this class. This is particularly relevant for this classification problem as our primary aim is to identify those loans which are

predicted to default. To alleviate this issue, the algorithm draws a bootstrap sample from the minority class and then randomly draws the same number of instances from the majority class.

We have had to build two random forest classifiers and run concurrent experiments as several of the reject inference techniques developed use the application dataset and others, such as augmentation, use the accepted dataset only. For those techniques which use the former, we had to reduce the number of columns in the accepted loans dataset to obtain common features with the application dataset, see Table 5. If we did not build an additional random forest on the reduced number of columns, the synthesised dataset would not fit to the original model. The side-effect of having to reduce the number of characteristics is that relevant information is lost and features that are strong indicators of the class label will likely be removed. For example, if we look at Figure 12, we can observe that Fico Score is strongly negatively correlated with the class label, 'Charged Off', but this will be lost with the reduction of features to match the application dataset. We will discuss this further in Section 9.

The following sections will discuss the two random forest classifiers in detail.

6.2.2.1. All Features

Firstly, we built a random forest classifier on all predictor variables available in the accepted loan dataset.

To determine the optimal hyperparameters, we carried out a cross-validated, see Section 0 for an explanation, random search of over several parameter options. We opted for a random search rather than a grid search as there were a large number of parameter combinations for an exhaustive search that we deemed unnecessary to devote computing resources towards. In addition, as the purpose of this study is not to determine the optimal model to apply to this dataset, so it was not vital to test every combination. Random search tests a certain number of combinations that are selected randomly thus it may not provide the optimal set of hyperparameters. The benefit is observed in the computation expense. Table 7 provides the hyperparameter search space. Grinsztajn, et al. (2022) noted that the Random Forest

hyperparameter that requires the most tuning, in terms of performance, is the maximum depth of the decision tree. We experimented with a random assortment of depths, ranging from 5 up to 1000, and found a distribution around 12 to be a suitable depth in terms of AUC performance for our model. For the other hyperparameters, the settings chosen for the search space were led by our intuition after several iterations of experiments. The hyperparameters not included in the search space remained in the default setting.

Parameter	**Setting**
Number of Estimators	10, 50, 100, 200
Criterion	Gini, Entropy
Max Depth	5, 10, 12, 15, 20
Max Features	Log2, Square Root

Table 7 – Random Forest Grid Search Parameter Space

The random search determined the hyperparameter settings for the random forest classifier with all features as shown in Table 8.

Parameter	**Setting**
Criterion	Gini
Max Depth	5
Max Features	Square Root
Number of Estimators	50

Table 8 – Full Feature Random Forest Selected Hyperparameters

The criterion parameter determines the function used to measure the quality of a branch split. Max depth defines the maximum depth of a tree within the forest. Max features indicates the number of features to consider when deciding the best split. Finally, the number of estimators is the number of trees in the forest.

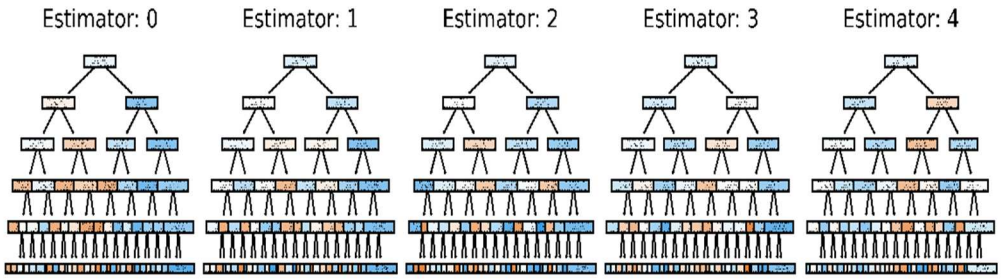

Figure 13 - First Five Decision Trees in Random Forest built on Full Feature Dataset. Presents the Variation in Trees as Different Colours Indicate the Class Labels and the Shades Indicate the Gini Strength.

6.2.2.2. Matched Features

For the random forest classifier fitted on the accepted loans dataset with features reduced to match those in the application dataset, Table 9 presents the optimal hyperparameter settings as determined by the random search. The same process as discussed in Section 6.2.2.1 was applied to this model, with the same parameter setting search space as well, see Table 7. There are variations in the parameters, and this would be expected as the dataset has experienced changes.

Parameter	**Setting**
Criterion	Entropy
Max Depth	10
Max Features	Log2
Number of Estimators	50

Table 9 - Common Feature Random Forest Chosen Hyperparameter Settings

6.3. Reject Inference Techniques

The implementation of the reject inference techniques, set out in Section 5.5, follow a similar pre-processing path once the methodology has been executed. Depending on whether the reject

inference technique utilises the application or accepted loan dataset determines the scaler[1] the missing values that require imputation and the columns that need converted into dummy variables. We first convert the categorical columns into dummy variables. Then we impute the mean value of the column for any missing values. This step is contentious as depending on the synthesized dataset, the values may be different, for example the mean value (after scaling) of the column Employment Length for the processed set used in the simple assignment technique is -1.321 whereas for augmentation the same column mean value is -1.407. We opted to impute the missing values on the synthesized dataset, as opposed to the original datasets, as we wanted to minimize the impact of the mean values on the models. If we imputed the values in the original datasets then when we came to synthesize the dataset after performing reject inference, then the engineered values may affect the model's interpretation as the required imputation would have different mean values between the two datasets. Lastly, we scale the values, as discussed in Section 6.1.1.

There are several techniques that require arbitrary values to determine the split of default class assignment. For random assignment we factored the default rate by 3.5 as it was mean value of the advised rate in Section 2.2.1.1. The default rate applied to the rejected instances is 64.05% of the sample. Our implementation of augmentation requires the dataset to be split and we chose 10 intervals. Parcelling used the same values, 3.5 for the rate factor and 10 intervals. Extrapolation, as described in the Section 5.4.5, used the actual default rate in the accepted loan dataset as the threshold for classifying the rejected instances. This threshold resulted in 88 instances being assigned to the default class and the remaining 3,432,143 assigned to non-default class.

Reject Inference Technique	**Training Set (Default %)**	**Test Set (Default %)**
Simple Assignment	3,931,748 (89.62%)	146,571 (26.79%)
Random Assignment	3,931,748 (58.24%)	146,571 (26.79%)
Augmentation	499,517 (18.3%)	146,571 (26.79%)
Parcelling	3,931,748 (58.24%)	146,571 (26.79%)

[1] While writing this report we arrived at the conclusion that it is unnecessary to use two distinct scalers as the values in the columns are the same in both datasets. We confirmed this by checking the mean values for the columns

Extrapolation	3,931,748 (2.33%)	146,571 (26.79%)
Fuzzy Augmentation	7,363,979 (47.85%)	146,571 (26.79%)
Bayesian Network	3,931,748 (44.87%)	146,571 (26.79%)

Table 10 - Training and Test Set Breakdown for Each Reject Inference Technique

6.3.1. Bayesian Network

For our implementation of BN as a reject inference methodology, we first binned the numerical features as BN as previous research has suggested. As we had to use the shared feature accepted loans dataset with fewer features, only 3 columns had to be processed, see Table 5. The reason we had to use this dataset is when predicting the class of the rejects, the input features must match with the model's fitted features. All numerical columns were binned evenly, 'Debt-to-Income' and 'Loan Amount' were split into 4 bins whereas 'Employment Length' was split into 3. The data type of each of these columns had to be cast to string. For the last bin, the largest values, for each feature, we changed to the value to greater than, for example, the last bin in the 'Debt-to-Income' column was 22.91 to 39.99 so we dropped the 39.99 and replaced it with '<'. The reason we did this is because there are values in the rejected dataset that are larger than the maximum values observed in the accepted loans dataset.

The rejected sample now had to be pre-processed in a similar manner. Initially, though, we must impute the missing values, which applied to the column Employment Length only, and we used the mean of the training set as our imputation value. The reason we altered our imputation method for this technique was because we did not want the BN's probability calculation to be affected by engineered values, instead minimize the impact.

Now that the data has been prepared, we must determine the structure of the BN. Similar to Anderson (2019), each possible DAG structure was scored using BIC. The optimal structure, with the lowest BIC score, for our dataset is shown in Figure 14. As we can observe this is indeed acyclical as no loop or self-connection is present. Nodes 'Charged Off', 'Purpose', 'Employment Length; and 'Debt-to-Income' all share a common parent and consequently are conditionally dependent on the 'Loan Amount' node. Interestingly, the optimal structure defines

'Charged Off' as independent of 'Purpose' and 'Employment Length', they have no influence on the class label. 'Debt-to-Income' node is a descendent of 'Loan Amount' and 'Charged Off' so, again, has no influence on the class label and our knowledge does not change for any value in this node. The most important relationship, the determinant of the class label, is notated as P('Charged Off' | 'Loan Amount').

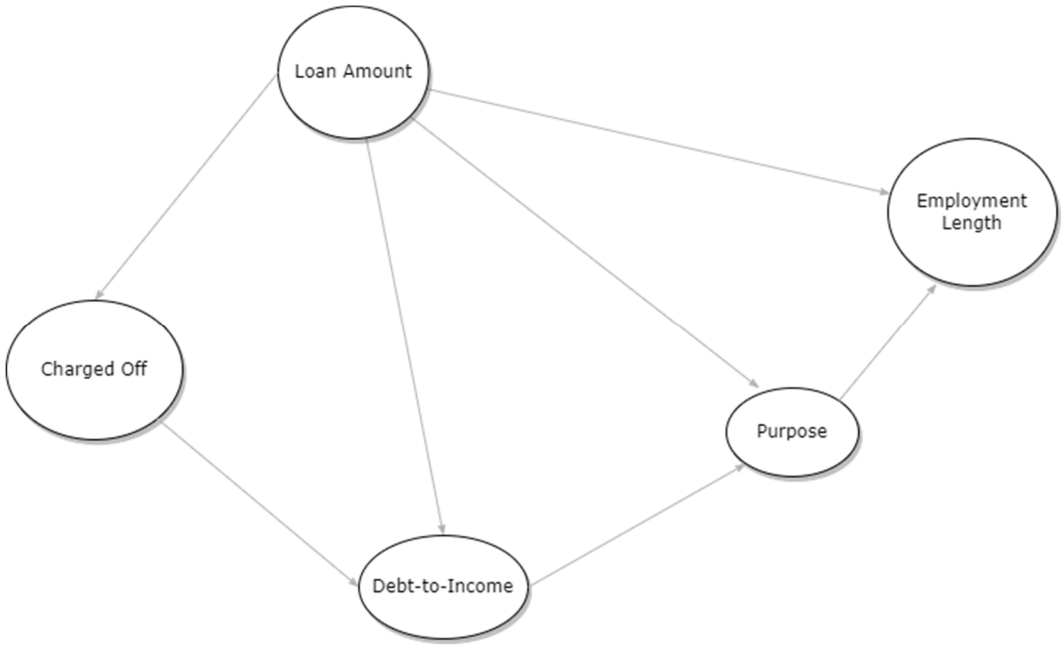

Figure 14 - Optimal Bayesian Network Structure

Since the structure of the BN has been established, we now need to calculate the CPDs of the variables. To do this we utilise the Bayesian Dirichlet equivalent uniform prior in our estimations of the probability distributions of the variables. Appendix Exhibit 1 provides the calculated CPDs. The implementation of the BN is now complete as all the components required are established.

7. Results & Findings

For all our training results we have cross validated the AUC 10 times. Cross validation splits the dataset into *k* folds, in this case 10, and trains the model on *k*-1 folds, with the remaining fold set aside to evaluate the model. The test fold will compute a score. The process is then repeated for each *k* fold until all splits have been evaluated. All scores are collected, and the mean reported. We also note the standard deviation to understand the distribution[2].

We then evaluate the model with the test set and report the AUC, recall for both classes (specificity is recall for the class label 1), PCC, and the confusion matrix, see Table 11.

Figure 15 displays the confusion matrix for the LR Application dataset. The predicted label on the x-axis is what the algorithm has predicted. 'ACCEPT' is the binary outcome 0 and 'REJECT' is the opposite outcome 1. We also display Figure 16, where 'NON-DEFAULT' is the binary outcome 0 and 'DEFAULT' is the complement 1. Thus, TP is the top left quadrant, FN the top right quadrant, FP the bottom left quadrant and TN the bottom right quadrant.

[2] We noticed that when we wrote the results to CSV and opened the file in Excel, we lost precision in each of the cross-validated fold's AUC score for the training set. For example, the first training AUC score for the LR Application Dataset is 0.9109895276561297 but Excel automatically rounded it to 0.91. This is a known issue with Excel which we were unaware of prior to saving and analysing the results. This conversion will have affected our standard deviation calculation, for example, the SD of the training set results for the BN should be 6.7% whereas with rounding the SD reported is 7.23%.

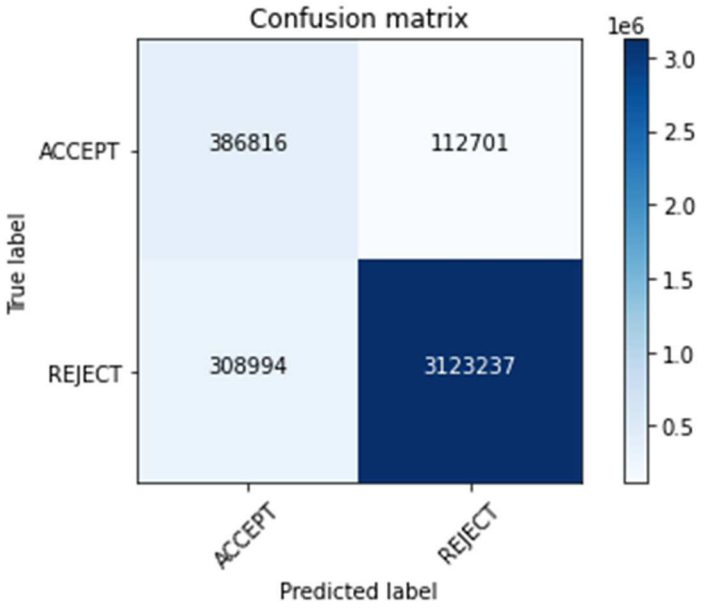

Figure 15 - LR Application Dataset Confusion Matrix

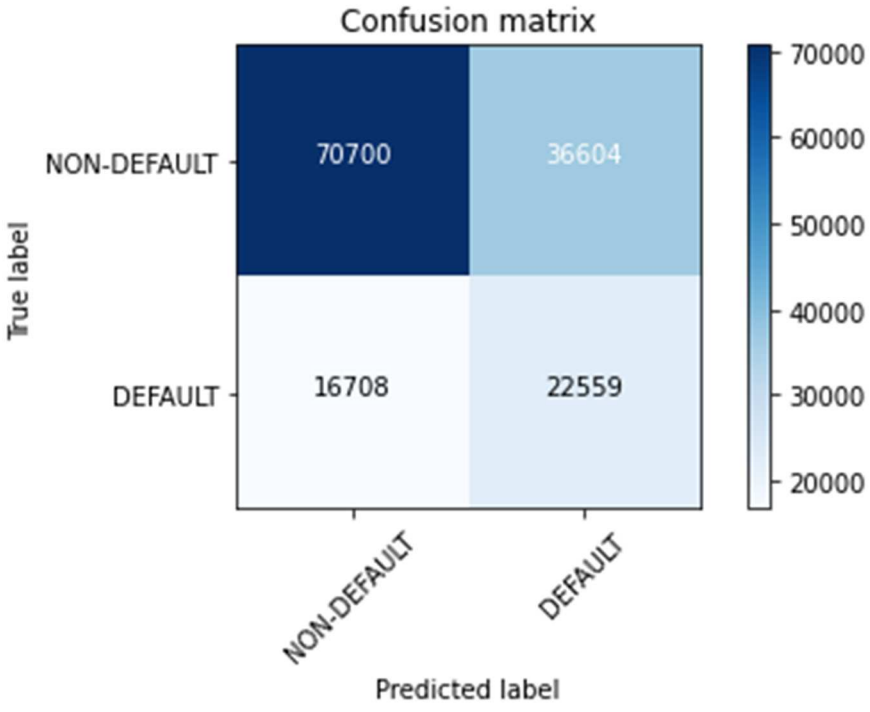

Figure 16 - Full RF (Accepts Only) Confusion Matrix

Model	Mean Train AUC (%)	SD Train AUC (%)	Test AUC	Recall	Specificity	PCC	TP	FN	FP	TN
LR Application Dataset	91.20	1.48	91.49	77.44	91.00	89.27	386,816	112,701	308,994	3,123,237
Full RF - Accepts Only	68.90	1.52	61.67	65.89	57.45	63.63	70,700	36,604	16,708	22,559
Full RF - Augmentation	67.90	2.73	62.24	50.02	74.46	56.57	53,669	53,635	10,027	29,240
Reduced RF - Accepts Only	62.60	7.62	58.26	54.08	62.44	56.32	58,028	49,276	14,749	24,518
Reduced RF - Simple Assignment	94.40	5.56	50.32	91.28	9.36	69.33	97,947	9,357	35,593	3,674
Reduced RF - Random Assignment	56.30	16.74	58.26	54.08	62.44	56.32	58,028	49,276	14,749	24,518
Reduced RF - Augmentation	60.60	1.58	57.91	46.57	69.24	52.64	49,969	57,335	12,077	27,190
Reduced RF - Parcelling	38.50	28.90	50.64	94.60	6.67	71.04	101,505	5,799	36,646	2,621
Reduced RF - Extrapolation	91.60	13.99	49.89	8.68	91.10	30.76	9,314	97,990	3,496	35,771
Reduced RF - Fuzzy Augmentation	49.90	5.84	50.50	90.56	10.44	69.10	97,174	10,130	35,166	4,101
Reduced RF - Bayesian Network	97.50	7.23	56.50	68.18	44.83	61.92	73,156	34,148	21,665	17,602

Table 11 – Results

In the following sections we will initially discuss the AR model and then the experiments with the GB model.

7.1. Accept / Reject Model

The results of the AR model trained on the full application dataset reported a mean AUC score of 91.2% with 1.48% standard deviation. The recall scores were 91% for rejected instances and 77.44% for accepted instances. The PCC score is encouraging with 89.27%. The reason for this may be the imbalanced dataset, although, it is certainly more important to a credit institution to

identify those applications that should be rejected rather than accepted. Nevertheless, the scores are satisfactory enough to proceed with the study.

7.2. Good / Bad Model

The main results of the GB model with and without the reject inference techniques applied are reported in the following sections.

The benchmark score for the GB model built on the accepted loan applications only, with all predictor variables available, reported a mean AUC score of 68.9%, 1.52% standard deviation. Evaluating the GB model on the test set produced AUC of 61.67%, recall for default loans of 57.45% and recall for non-default loans of 65.68%. The PCC is 63.63%. The discrepancy between the AUC scores observed shows that there is an element of overfitting. The recall scores are low which indicates that the model is unable to identify defaulting loans efficiently.

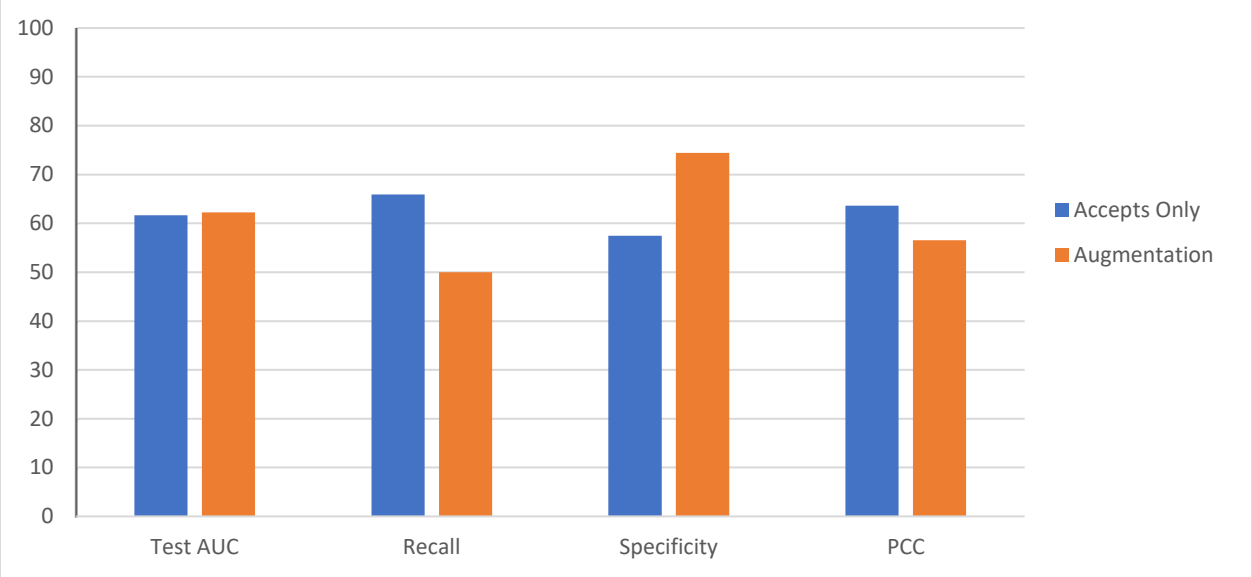

Figure 17 - Comparison of Full Feature GB Model with and without Augmentation

For the GB model built on the reduced features matching those available in the application dataset, we have observed a mean AUC score of 62.6%, standard deviation of 7.62%, for the training set. In terms of the prediction on the test set, the AUC score is 58.26%, recall for default

loans is 62.44%, recall for non-default loans is 54.08% and PCC is 56.32%. In comparison to the results for the GB model fitted on the full feature set, there is an expected drop-off in the AUC score and the PCC with less information available to discriminate, although, not as large as one would expect. The rise in the standard deviation of the AUC scores is concerning as it indicates that the model is not intelligently classifying. The recall scores are, interestingly, switched in terms of performance for correctly identifying classes. In terms of overall classification performance, enrichment of the models could improve the scores thus there is certainly scope for reject inference.

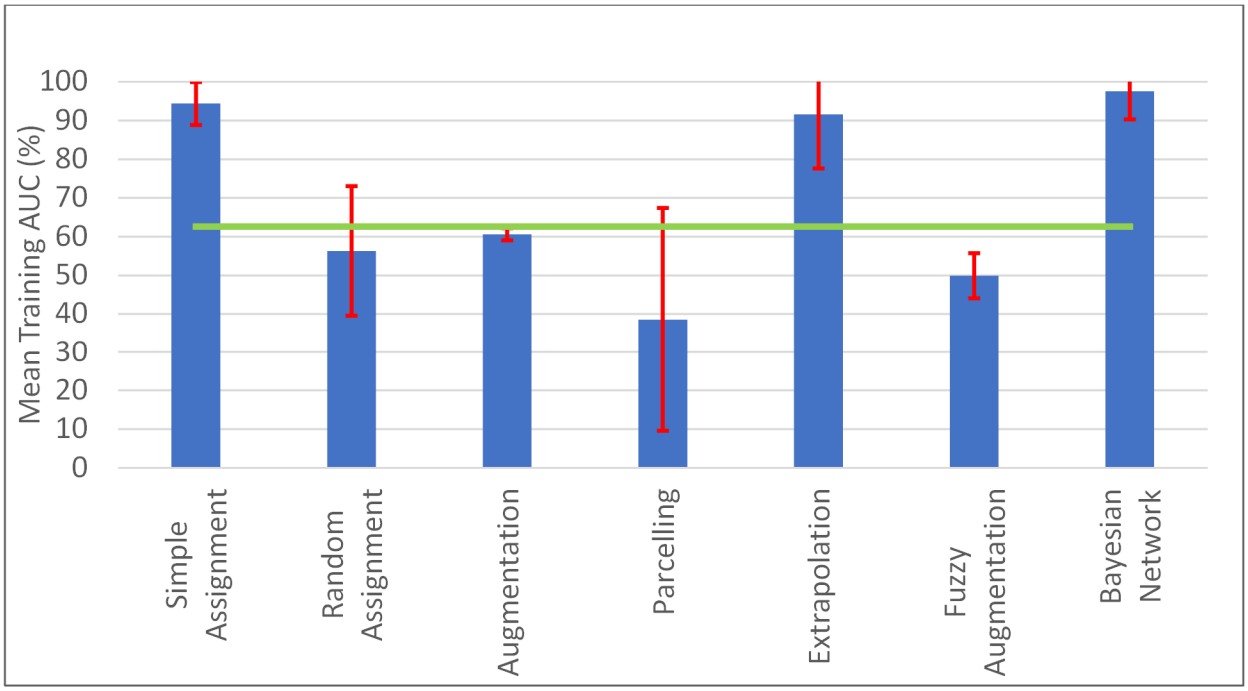

Figure 18 - GB Model Mean Training AUC Scores. Blue bars - Each Reject Inference Technique. Red Whiskers – Standard Deviation. Green Line – Benchmark Model (Accepts-Only)

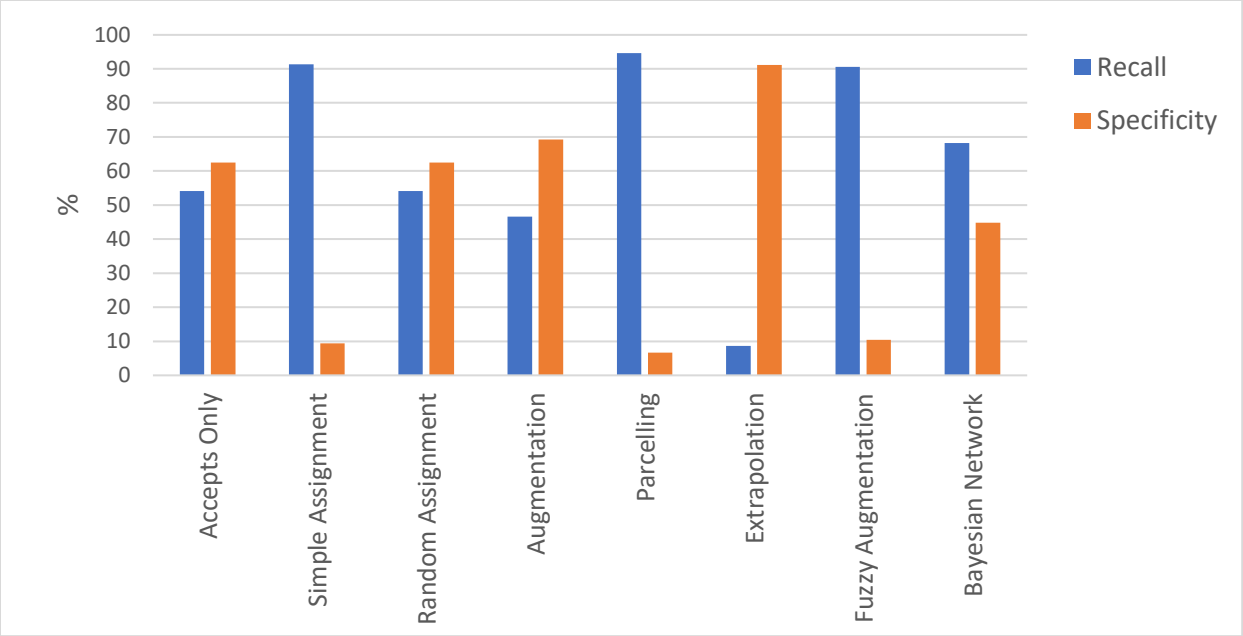

Figure 19 - GB Model Recall & Specificity Scores

7.2.1. Simple Assignment

The results of the GB model when assigning all rejects in the application dataset to the default class reported a mean cross-validated AUC of 94.9%, standard deviation of 5.56%, on the training dataset. The test AUC score was 50.32%, recall for the default class 9.36%, recall for the non-default class 91.28% and PCC is 69.33%. There is clear evidence of overfitting to the training dataset with such a large disparity between the training and test AUC score and the recall scores between the dichotomous classes.

7.2.2. Random Assignment

The Random assignment reject inference technique returned a mean cross-validated AUC score of 56.3%, standard deviation 16.74%, on the training set. The evaluation results reported 58.26% AUC, 62.44% recall for the default class, 54.08% for the non-default class and 56.32% PCC.

The standout statistic is the standard deviation of the AUC score for the training set. It evidences high volatility which leads to unreliable models.

7.2.3. Augmentation

Augmentation on the full feature accepted loan dataset reported a mean cross-validated AUC score on the training set of 67.9%, standard deviation 2.73%. The evaluation of the test set provided an AUC of 62.24%, recall on the default class of 74.46%, recall on the non-default class of 50.02% and PCC 56.57%. The disparity between the recall scores lends itself to the additional weighting applied to instances with similar characteristics to the rejected sample.

Augmentation on the common features between the accepted and application dataset reported a mean cross-validated AUC score on the training set of 60.6%, standard deviation 1.58%. Evaluating the model on the test set reported an AUC of 57.91%, recall on the default class of 69.24%, recall on the non-default class of 46.57% and PCC 52.64%.

7.2.4. Parcelling

The parcelling technique produced a mean cross-validated AUC score of 38.5%, standard deviation of 28.9%, on the training dataset. The evaluation of the model on the test set produced an AUC score of 50.64%, recall on the default class of 6.67%, recall on the non-default class of 94.6% and PCC 71.04%. All metrics are poor for this technique except when identifying non-default loans and a reasonable explanation for this is the model has predicted the class label 0 (non-default) 94.26% of the time.

7.2.5. Extrapolation

The mean cross-validated AUC training set with extrapolated rejects is 91.6%, standard deviation of 13.99%. The evaluation metrics are as follows: AUC 49.89%, recall on the default

class 91.1%, recall on the non-default class 8.68% and PCC 30.76%. Reverse to parcelling, the model with extrapolated rejects appears to be predicting the vast majority of observations as default which explains the high recall score for the default class.

7.2.6. Fuzzy Augmentation

Results for fuzzy augmentation are 49.9%, standard deviation 5.84%, for the mean cross-validated AUC score on the training set. Evaluation on the test set produced an AUC 50.5%, recall on the default class 10.44%, recall on the non-default class 90.56% and PCC 69.10%.

7.2.7. Bayesian Network

The results of BN reject inference implementation classified reported a mean cross-validated AUC score of 97.50%, standard deviation 7.23%. The evaluation performance reported an AUC of 56.50%, recall on the default class of 44.83%, recall on the non-default class of 68.18%, and PCC of 61.92%. The margin between the training and test AUC indicates the model overfitted to the training dataset.

8. Discussion

The aim of our research is to understand the performance of a selection of reject inference techniques in a credit application framework. Through our experiments we were able to report results that we will analyse in this section and reflect on with our literature review.

Overall classification performance of our models is weak according to previous research (Liu, et al., 2020; Luca & Desjardins, 2018). The best performing GB model in terms of AUC were the models' trained on the full feature dataset, marginally above 60%.

In terms of the reject inference techniques, firstly, we analyse the benchmark results of the GB model fitted on all the features with the only reject inference technique which utilises the accepted loan dataset, augmentation. We can observe that, as Banasik & Crook (2007) concluded, that the reject inference technique can be detrimental to classification performance (63.63% to 56.57%). Although, if we analyse other metrics such as the recall, which is a key model performance indicator as the priority for lenders is to identify potential defaulting loans (Turiel & Aste, 2020), we have observed that the model with augmentation can identify the default class significantly better than the benchmark model (57.45% to 74.46%). The reason for this is the additional weight assigned to those instances that are similar to the rejected instances. It would be interesting to understand if the research by Banasik & Crook (2007) also observed an increase in recall for the default class by implementing augmentation as we are unable to extract that information from their study. However, if implementing augmentation, the assumption, Equation 1, must be acknowledged.

The benchmark results for the GB model fitted to the accepts only dataset with features in common with the application dataset showed a clear downward trend in performance across the majority of the metrics against the full feature set GB model. Similarly, if we analyse the results of the models fitted on augmented data, there is a drop-off of ≈5% in every evaluation metric between the two models. This is to be expected as models have lost strong predictive information, for example, the FICO score, as discussed in Section 6.2.2. Although, we observe an uptick in the recall performance of the default class on the test set in the benchmark model.

The reason for this is that the variable Loan Amount is a key indicator of the class label, also shown by the structure of the BN. With the full feature set, other variables effect the model's performance by introducing noise when identifying default instances.

If we compare the performance of the benchmark GB model fitted on the common feature set with all the reject inference techniques implemented, we agree with Ehrhardt, et al. (2021) that there is no standout. Not one of the models', with reject inference implemented, evaluation AUC score exceeds the benchmark. Techniques such as extrapolation, fuzzy augmentation, parcelling, and simple assignment show a very large discrepancy between the recall scores for each class. There should not be an issue of the model simply selecting the majority class as the RF implemented undersamples in each bootstrap sample to accommodate the imbalanced dataset. Rather the model appears to be unintelligently choosing the same class in the majority of predictions when these techniques have been implemented. Examining the AUC confirms the non-existent discriminatory power as each iteration reports a similar (≈50%) score which is considered very poor performance, it may as well be flipping a coin to decide which class to assign to.

Random assignment and the benchmark model report exactly the same scores for the evaluation metrics. This provides evidence that the loss of predictor variables detracts from the model's ability to perform. We have eluded that the key predictor information will most likely be lost by reducing the number of characteristics, and this is evidence of that. By adding more random samples without any concern for the appropriateness of the class labels, the performance neither increases nor decreases. This confirms that the loss of predictor variables, in essence, randomises the data as there is now no sense in the class label.

Initially, our expectation was that the BN machine learning technique would be the standout method, and out-perform the benchmark models and the statistical reject inference techniques. As the results show, performance is overall mediocre and does not out-perform any model across all metrics except mean training AUC. The reason the training AUC is exemplary (97.50%) is that the structure and CPDs of the BN is built on the accepted loan dataset. The rejected sample's class label is predicted by the built BN and when fitting the GB model, the additional samples

reinforce the model's predictions rather than adding to the complexity of the model. This is confirmed by the poor evaluation metrics on the unseen data. This is an example of the dangerous assumption, spoken about in Section 2.2.3, that the rejected sample shares the same characteristic distribution of he accepted loans (Henley, 1995; Crook & Banasik, 2004). Even, the proponent of the BN, Anderson (2019), argued against the use of methods which used the characteristics of the accepted sample to infer the performance of the rejected sample. Yet, our implementation, guided by their research, did exactly that as the structure and CPDs are all based on the accepted sample.

These BN evaluation results were disappointing, and we were keen to understand the reasons behind the underwhelming performance of the BN. One possible explanation is the binning of continuous predictor variables leads to loss of information for the model. We chose to pre-process our data in this matter as suggested by Anderson (2019). By categorising, we are assuming the relationship between the predictor variable and outcome is flat within intervals. If the continuous predictor variables had natural grouping, such as legal age to drive or boiling point, then it would be intuitive and insights from the results would be meaningful. In this study, that is not the case, and the groupings are arbitrary, simply an even split. A more sophisticated method for handling continuous variables may provide better performance of this reject inference technique. A final remark on our implementation of the BN is that the model should have been fitted on balanced data. This likely had an impact on the structure and the CPDs. As we used an imbalanced dataset to setup the BN, the CPDs weight the skewed class favourably as it has observed more instances.

9. Limitations

This project encountered limitations that have had an effect on its the final outcomes. The sections below provide a summative discussion of these limitations.

9.1. Access to Real Data

This project was agreed to be undertaken in association with an industry partner as part of The Data Lab MSc programme. The author was successful in applying for The Data Lab MSc programme which had several benefits including an arranged summer funded placement within the industry. The project offered was to be supported by Royal London, a mutual life insurance and pension provider company, and the purpose of the research was to create a proof of concept for the applicability of reject inference in a life insurance framework. This included the sharing of code and real-life insurance datasets for testing and performing evaluation. This would allow us to gain industrial experience and deliver a project with real world implications.

However, due to the lengthy background checks, the author was not onboarded with the industry partner (Royal London) as an employee until the 3rd of August 2022. This was a requirement as we would have access to sensitive data and intellectual property. As there was not sufficient time before the deadline, for this reason, the work presented here has been evaluated using the open and available P2P loan datasets (see Section 5.3).

Working with a real dataset within life insurance context has been added as future work (see Section 11.1).

9.2. Computational Resources

This project was limited by the computational resources available to the author. Certain aspects of the research could have been improved, for example, hyperparameter tuning and increasing the cross validation, such as Li, et al. (2017) who undertook 50 fold cross validation. Several of

the techniques use arbitrary values, for example, random assignment factors the default rate in the training set by a specified value or the number of intervals in parcelling. These values should be tuned. Decisions had to be made on the critical areas of the research that required devoted computing resources. As a result of this, full confidence in the performance and results of our study cannot be assured.

9.3. Handling of Null Values

As noted in our introduction, missing values are to be expected and this research was not any different. Null or missing values in the datasets need to be handled so models can interpret the values. There are numerous solutions to this problem, each with their own advantages and disadvantages. For this study, we used the mean value of the column to impute any missing values which is not a particularly sophisticated method. Alternatives were offered in Section 6.1.1, such as KNN and MICE algorithms, although these have their drawbacks too, namely computational requirements. We also considered simply removing all instances that had null values but the number of instances (77,137) in the accepted loan dataset we would lose was too significant (11.93%) to consider this action. While we do not anticipate this will have a profound effect on our findings, it is worth highlighting as a limitation.

10. Conclusion

The aim of this research is to "*Investigate the performance of a selection of reject inference techniques in a peer-to-peer (P2P) credit application framework"*. The literature was first reviewed (Objective 1 – Chapter 2) and we were able to determine the reject inference techniques that we should implement. An appropriate credit application framework (Objective 2 – Chapter 5, Section 5.4) and reject inference techniques (Objective 3 – Chapter 5, Section 5.5) were then developed. The reject inference techniques were then applied to the appropriate datasets (Objective 4 – Chapter 6, Section 6.3) to create a new dataset. The models were refitted and evaluated. Results were then collated and presented (Objective 5 & 6 – Chapter 7, Section 7.2). A discussion was subsequently presented (Objective 7 – Chapter 8) and notable findings examined. Although, a new, better credit application framework could not be proposed (Objective 8) as the results of our study did not outright support a reject inference technique.

In conclusion of our research, we were able to meet the aim set out and a thorough investigation of selected reject inference techniques was complete in a P2P credit application framework. Our research provides evidence that there is a benefit to implementing the reject inference technique augmentation if the purpose of credit models is to identify default loans rather than overall classification performance. Care needs to be taken when using reject inference and that the assumptions are understood, otherwise models can amplify the bias with inclusion of inferred performance of the rejected sample. We hope the results of our study can be used to inform future researchers on the application of reject inference techniques in a credit application framework and the need for rich data to observe the benefits. The research also offered the industry possibly valuable techniques to improve the credit application framework if the upmost concern is the identification of defaults.

11. Further Work

There are several areas where this research can be extended. This chapter will discuss these possibilities in the use of real data, additional reject inference techniques, and alternative application contexts.

11.1. Real Data

After the discussion in Section 9, real data was not available for our research project. To really understand and evaluate the performance of the reject inference techniques developed in this study, real data would have been extremely beneficial. The P2P open data which we have utilised had already been pruned by Turiel & Aste (2020) for their research. We had to further reduce the number of features in the accepted loan dataset to match those in the application dataset which resulted in a loss of predictive information for the models. Real data would provide richer insights and the ability for models to identify patterns.

11.2. Reject Inference

Due to the time constraints, discussed in Section 9, we did not have the opportunity to implement all the reject inference techniques researched. We were unable to research the applicability of the statistical method reclassification or the machine learning methods S3VM, ensemble learning framework and the three-stage framework. Further work could implement these additional techniques to study their performance.

11.3. Alternative Contexts

As mentioned by Banasik & Crook (2007), reject inference could be applied to a wide variety of contexts. All the literature reviewed examined reject inference in a credit application setting.

This research continue that theme with a P2P lending dataset. Further work could investigate the performance of reject inference in alternative contexts such as insurance.

12. References

Anderson, B., 2019. Using Bayesian networks to perform reject inference. *Expert Systems with Applications,* Volume 137, pp. 349 - 356.

Banasik, J. & Crook, J., 2007. Reject inference, augmentation, and sample selection. *European Journal of Operational Research,* 183(3), pp. 1582 - 1594.

Chakraborty, T., 2017. *EC3: Combining Clustering and Classification for Ensemble Learning.* s.l., IEEE, pp. 781 - 786.

Chen, C., Liaw, A. & Breiman, L., 2004. *Using Random Forest to Learn Imbalanced Data,* s.l.: University of California, Berkeley.

Crook, J. & Banasik, J., 2004. Does reject inference really improve the performance of application scoring models?. *Journal of Banking & Finance,* 28(4), pp. 857 - 874.

Dastile, X., Celik, T. & Potsane, M., 2020. Statistical and machine learning models in credit scoring: A systematic literature survey. *Applied Soft Computing,* Volume 91.

Ehrhardt, A. et al., 2021. Reject Inference Methods in Credit Scoring. *Jorunal of Applied Statistics,* 48(13-15), pp. 2734 - 2754.

Grinsztajn, L., Oyallon, E. & Varoquaux, G., 2022. Why do tree-based models still outperform deep learning on tabular data?.

Ha-Thu, N., 2016. *Reject inference in application scorecards: evidence from France,* s.l.: University of Paris Nanterre, EconomiX.

Henley, W. E., 1995. *Statistical aspects of credit scoring.* s.l.:The Open University.

Liu, Y., Li, X. & Zhang, Z., 2020. A new approach in reject inference of using ensemble learning based on global semi-supervised framework. *Future Generation Computer Systems,* Volume 109, pp. 382 - 391.

Li, Z. et al., 2017. Reject inference in credit scoring using Semi-supervised Support. *Expert Systems With Applications,* Volume 74, pp. 105 - 114.

Luca, S. & Desjardins, 2018. *Evaluation of Different Approaches to Reject Inference: a case study in Credit Risk.* Denver, SAS.

Mancisidor, R. A., Kampffmeyer, M., Aas, K. & Jenssen, R., 2020. Deep Generative Models for Reject Inference in Credit Scoring. *Knowledge-Based Systems,* Volume 196.

Shen, F., Zhao, X. & Kou, G., 2020. Three-stage reject inference learning framework for credit scoring using unsupervised transfer learning and three-way decision theory. *Decision Support Systems,* 137(113366).

Turiel, J. D. & Aste, T., 2020. Peer-to-peer loan acceptance and default prediction with artificial intelligence. *Royal Society Open Science,* 7(6).

13. Appendix

13.1. Exhibit 1

Conditional probability distributions of the Bayesian Network implemented.

Loan Amount								
loan_amnt_cat((12000.0, 19750.0])	0.232273							
loan_amnt_cat((499.999, 8000.0])	0.281237							
loan_amnt_cat((8000.0, 12000.0])	0.237125							
loan_amnt_cat(19750.0 <)	0.249365							

Charged Off Variable								

loan_amnt_cat	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat(19750.0 <)				
charged_off(0)	0.796952603	0.852519944	0.824603153	0.788319388				
charged_off(1)	0.203047397	0.147480056	0.175396847	0.211680612				

Purpose Variable								
loan_amnt_cat	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])
purpose	purpose(car)	purpose(credit_card)	purpose(debt_consolidation)	purpose(home_improvement)	purpose(house)	purpose(major_purchase)	purpose(medical)	purpose(moving)
emp_length_cat((-0.001, 4.0])	0.452322464	0.367744213	0.347179009	0.329471202	0.467366011	0.436155479	0.438723546	0.465404518
emp_length_cat((4.0, 9.0])	0.282722572	0.309474957	0.310313562	0.31564228	0.291674528	0.322982111	0.284776586	0.309100088
emp_length_cat(9.0 <)	0.264954964	0.32278083	0.342507428	0.354886517	0.240959461	0.24086241	0.276499868	0.225495393
loan_amnt_cat	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])
purpose	purpose(other)	purpose(small_business)	purpose(vacation)	purpose(wedding)	purpose(car)	purpose(credit_card)	purpose(debt_consolidation)	purpose(home_improvement)

emp_length_cat((-0.001, 4.0])	0.373517148	0.445477136	0.316140587	0.534650461	0.459958698	0.43567922	0.417512363	0.300895564
emp_length_cat((4.0, 9.0])	0.305002299	0.324415337	0.367718827	0.303349931	0.3111357	0.330685506	0.324882317	0.345444838
emp_length_cat(9.0 <)	0.321480554	0.230107527	0.316140587	0.161999607	0.228905602	0.233635275	0.25760532	0.353659598
loan_amnt_cat	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((499.999, 8000.0])
purpose	purpose(house)	purpose(major_purchase)	purpose(medical)	purpose(moving)	purpose(other)	purpose(small_business)	purpose(vacation)	purpose(wedding)
emp_length_cat((-0.001, 4.0])	0.472845686	0.440606686	0.378726067	0.593365525	0.400750158	0.49340594	0.33116592	0.568585991
emp_length_cat((4.0, 9.0])	0.313181549	0.305494228	0.323393319	0.258281597	0.316479127	0.281157768	0.356710434	0.283839432
emp_length_cat(9.0 <)	0.213972765	0.253899086	0.297880614	0.148352878	0.282770715	0.225436291	0.312123646	0.147574577
loan_amnt_cat	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])
purpose	purpose(car)	purpose(credit_card)	purpose(debt_consolidation)	purpose(home_improvement)	purpose(house)	purpose(major_purchase)	purpose(medical)	purpose(moving)
emp_length_cat((-0.001, 4.0])	0.441139955	0.401991934	0.387619515	0.306550535	0.490164046	0.427057463	0.388644795	0.565574507

emp_length_cat((4.0, 9.0])	0.318060729	0.321847214	0.317046545	0.32667888	0.319610646	0.312878335	0.309424508	0.264338608
emp_length_cat(9.0 <)	0.240799316	0.276160852	0.29533394	0.366770585	0.190225308	0.260064201	0.301930697	0.170086885
loan_amnt_cat	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)
purpose	purpose(other)	purpose(small_business)	purpose(vacation)	purpose(wedding)	purpose(car)	purpose(credit_card)	purpose(debt_consolidation)	purpose(home_improvement)
emp_length_cat((-0.001, 4.0])	0.387311737	0.484364312	0.41430068	0.556696335	0.440235032	0.329456016	0.312419922	0.325594286
emp_length_cat((4.0, 9.0])	0.301419095	0.291443182	0.314583	0.299651611	0.25600019	0.277410944	0.282107504	0.307734945
emp_length_cat(9.0 <)	0.311269168	0.224192506	0.27111632	0.143652054	0.303764778	0.393133039	0.405472574	0.366670769
loan_amnt_cat	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)	loan_amnt_cat(19750.0 <)
purpose	purpose(house)	purpose(major_purchase)	purpose(medical)	purpose(moving)	purpose(other)	purpose(small_business)	purpose(vacation)	purpose(wedding)
emp_length_cat((-0.001, 4.0])	0.387632505	0.431565734	0.365331772	0.491976057	0.339733875	0.386259807	0.414165428	0.53166405
emp_length_cat((4.0, 9.0])	0.304779458	0.294082806	0.330211534	0.227571518	0.300592101	0.294827837	0.257256067	0.285733961
emp_length_cat(9.0 <)	0.307588036	0.27435146	0.304456693	0.280452426	0.359674024	0.318912356	0.328578504	0.182601989

DTI Variable								
charged_off	charged_off(0)	charged_off(0)	charged_off(0)	charged_off(0)	charged_off(1)	charged_off(1)	charged_off(1)	charged_off(1)
loan_amnt_cat	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat(19750.0 <)	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat(19750.0 <)
dti_cat((-0.001, 11.37])	0.236714125	0.302316765	0.270535226	0.237904247	0.163216497	0.232407134	0.188738305	0.169380301
dti_cat((11.37, 16.92])	0.254817955	0.244971429	0.255218911	0.274076938	0.210078315	0.221016416	0.219062303	0.230781356
dti_cat((16.92, 22.91])	0.258008295	0.230159165	0.242789763	0.261785148	0.264198622	0.247176454	0.255691766	0.278756856
dti_cat(22.91 <)	0.250459625	0.222552642	0.231456099	0.226233666	0.362506566	0.299399996	0.336507626	0.321081487

Purpose Variable								
dti_cat	dti_cat((-0.001, 11.37])	dti_cat((-0.001, 11.37])	dti_cat((-0.001, 11.37])	dti_cat((-0.001, 11.37])	dti_cat((11.37, 16.92])	dti_cat((11.37, 16.92])	dti_cat((11.37, 16.92])	dti_cat((11.37, 16.92])
loan_amnt_cat	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat(19750.0 <)	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat(19750.0 <)
purpose(car)	0.010532109	0.039613224	0.016247189	0.004744477	0.00505154	0.025414768	0.009397303	0.002152414
purpose(credit_card)	0.224301712	0.188556558	0.23084376	0.200092123	0.240186278	0.201868122	0.24590214	0.236437122
purpose(debt_consolidation)	0.566970383	0.440175722	0.538370664	0.544825378	0.638168658	0.491298794	0.605137112	0.645094383

purpose(home_improvement)	0.077526981	0.079298944	0.077838251	0.117403578	0.048227385	0.06866574	0.048509477	0.056882486
purpose(house)	0.008899983	0.006289895	0.006921202	0.013261289	0.004069474	0.004776949	0.004173301	0.005273786
purpose(major_purchase)	0.023861135	0.054385672	0.026627705	0.029180901	0.011785709	0.037620564	0.016860164	0.010637697
purpose(medical)	0.008783403	0.020599181	0.010282512	0.006145978	0.005297057	0.021640995	0.006649614	0.003091856
purpose(moving)	0.004236767	0.018502761	0.006789386	0.002552386	0.002245636	0.016452058	0.003969768	0.001455409
purpose(other)	0.041542496	0.10723496	0.05299092	0.037913329	0.027989804	0.096379383	0.040266409	0.020638208
purpose(small_business)	0.024988079	0.020477296	0.02076189	0.039242957	0.012241669	0.012707768	0.011500473	0.01581978
purpose(vacation)	0.002488061	0.015065607	0.004746237	0.001079013	0.001158349	0.015154824	0.002918183	0.000485662
purpose(wedding)	0.005868892	0.00980018	0.007580283	0.003558592	0.003578441	0.008020035	0.004716054	0.002031196
dti_cat	dti_cat((16.92, 22.91])	dti_cat((16.92, 22.91])	dti_cat((16.92, 22.91])	dti_cat((16.92, 22.91])	dti_cat(22.91 <)	dti_cat(22.91 <)	dti_cat(22.91 <)	dti_cat(22.91 <)
loan_amnt_cat	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat(19750.0 <)	loan_amnt_cat((12000.0, 19750.0])	loan_amnt_cat((499.999, 8000.0])	loan_amnt_cat((8000.0, 12000.0])	loan_amnt_cat(19750.0 <)
purpose(car)	0.003258702	0.021814209	0.005961007	0.001604112	0.003344743	0.015492017	0.005744451	0.001206795
purpose(credit_card)	0.232736721	0.194700031	0.244262031	0.236627302	0.225053021	0.190095463	0.238764492	0.236757343
purpose(debt_consolidation)	0.67091574	0.509051795	0.623504141	0.678358363	0.692571047	0.550837079	0.645747153	0.700459801
purpose(home_improvement)	0.034241393	0.06287115	0.042720757	0.03721002	0.030001472	0.056548327	0.034766375	0.029497631

purpose(house)	0.003990053	0.003794434	0.003618305	0.003146934	0.002745361	0.00307469	0.002433451	0.002119402
purpose(major_purchase)	0.00827843	0.031910178	0.012644598	0.00735188	0.007887428	0.025505009	0.010677164	0.005378715
purpose(medical)	0.003757351	0.022364898	0.007235712	0.002783917	0.003597114	0.021548507	0.007298593	0.002021623
purpose(moving)	0.001663027	0.015358907	0.004100626	0.001301598	0.001641236	0.013574636	0.003345665	0.000880863
purpose(other)	0.026828153	0.10209246	0.039723476	0.017365096	0.024764766	0.094835086	0.038111161	0.012972914
purpose(small_business)	0.010405997	0.012207741	0.00947506	0.012192105	0.005994642	0.009252919	0.007704022	0.007334303
purpose(vacation)	0.000998163	0.016399098	0.002205793	0.000333553	0.000884121	0.014761586	0.003278094	0.000424559
purpose(wedding)	0.00292627	0.007435101	0.004548495	0.001725118	0.00151505	0.004474683	0.00212938	0.00094605