A project report on

CREDIT PROFIT-RISK ANALYSIS

Submitted in partial fulfillment for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

by

MANOJ ARULMURUGAN (19BCE1301)

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

April, 2023

CREDIT PROFIT-RISK ANALYSIS

Submitted in partial fulfillment for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

by

MANOJ ARULMURUGAN (19BCE1301)

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

April, 2023

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

DECLARATION

I here by declare that the thesis entitled "CREDIT PROFIT-RISK ANALYSIS" submitted by me, for the award of the degree of Bachelor of Technology in Computer Science and Engineering, Vellore Institute of Technology, Chennai, is a record of bonafide work carried out by me under the supervision of Guide Name

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Chennai

Date: Signature of the Candidate

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

School of Computer Science and Engineering

CERTIFICATE

This is to certify that the report entitled **"CREDIT PROFIT-RISK ANALYSIS"** is prepared and submitted by **Manoj Arulmurugan (19BCE1301)** to Vellore Institute of Technology, Chennai, in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** programme is a bonafide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma and the same is certified.

Signature of the Guide:

Name: Dr./Prof.

Date:

Signature of the Examiner 1	Signature of the Examiner 2
Name:	Name:
Date:	Date:

Approved by the Head of Department
B. Tech. CSE

Name: Dr. Nithyanandam P

Date: 24 – 04 – 2023

(Seal of SCOPE)

ABSTRACT

After the worldwide mortgage crisis of 2008, it is obvious that corporate credit scoring is playing a significant role in credit risk management. This has highlighted the significant percentage of the banking sector that is focused on consumer lending, where credit rating is crucial. This has again renewed an interest in me and the world when COVID-19 struck the world a year ago and people started having irregular income, therefore forcing a large number of people to become credit or loan defaulters. From the point of view of financial lending institutions, this task of accurately rating the credit worthiness of potential customers while also keeping in mind the aspect of making a profit is indeed a challenging one. This research aims to propose and develop a credible profit-risk model in three distinct steps. First, a novel ensembling machine learning model that can predict the probability of default for borrowers of a financial institution Second, estimate the potential loss of a bad loan and the potential revenue of a good loan for each customer using data from the dataset. Finally, use the outcomes of the previous two steps to determine up to which percentile of potential borrowers the financial institution can lend.

ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to Guide name, , SCOPE, Vellore Institute of Technology, Chennai, for her constant guidance, continual encouragement, understanding; more than all, she taught me patience in my endeavor. My association with her is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of

It is with gratitude that I would like to extend thanks to our honorable Chancellor, Dr. G. Viswanathan, Vice Presidents, Mr. Sankar Viswanathan, Dr. Sekar Viswanathan and Mr. G V Selvam, Assistant Vice-President, Ms. Kadhambari S. Viswanathan, Vice-Chancellor, Dr. Rambabu Kodali, Pro-Vice Chancellor, Dr. V. S. Kanchana Bhaaskaran and Additional Registrar, Dr. P.K.Manoharan for providing an exceptional working environment and inspiring all of us during the tenure of the course.

Special mention to Dean, Dr. Ganesan R, Associate Dean Academics, Dr. Parvathi R and Associate Dean Research, Dr. Geetha S, SCOPE, Vellore Institute of Technology, Chennai, for spending their valuable time and efforts in sharing their knowledge and for helping us in every aspect.

In jubilant mood I express ingeniously my whole-hearted thanks to Dr. Nithyanandam P, Head of the Department, Project Coordinators, Dr. Abdul Quadir Md, Dr. Priyadarshini R and Dr. Padmavathy T V, B. Tech. Computer Science and Engineering, SCOPE, Vellore Institute of Technology, Chennai, for their valuable support and encouragement to take up and complete the thesis.

My sincere thanks to all the faculties and staff at Vellore Institute of Technology, Chennai, who helped me acquire the requisite knowledge. I would like to thank my parents for their support. It is indeed a pleasure to thank my friends who encouraged me to take up and complete this task.

Place: Chennai

Date: Name of the student

CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS

AE - Auto Encoders

SMOTE -Synthetic Minority Oversampling Technique

XGBoost - Extreme Gradient Boosting

CNN – Convolution Neural Networks

Chapter 1

Introduction

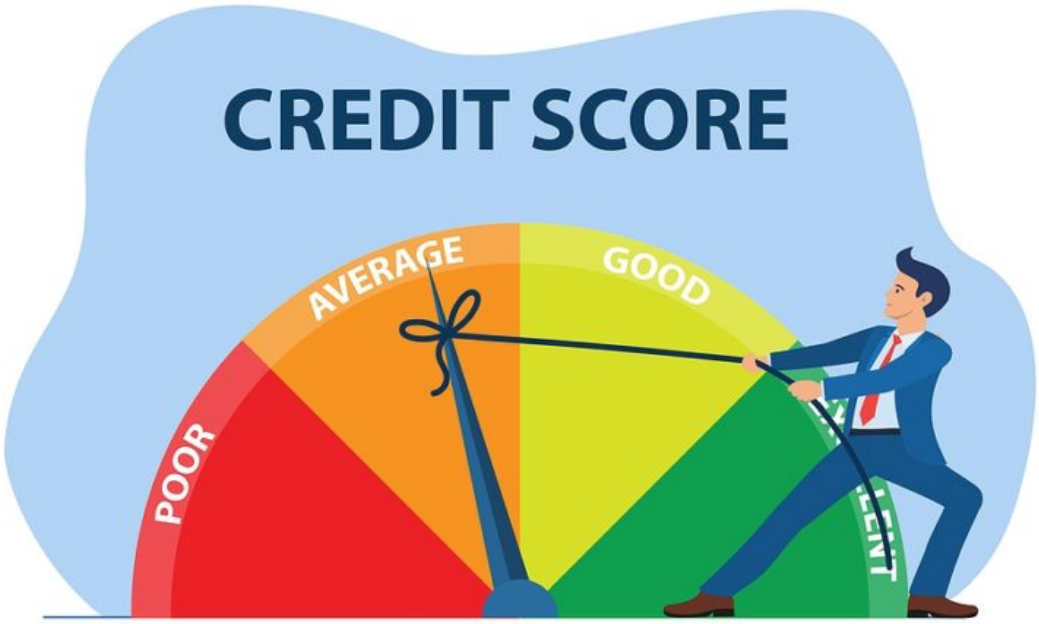


Fig 1: Credit Score

1.1 RESEARCH MOTIVATION

The objective of this research is to create a suitable machine learning model for banks or financial institutions to determine a customer's credit rating. This credit rating, 'Good' or 'Bad', refers to the process of evaluating an individual's creditworthiness that reflects whether an application of the customer should be approved or declined.

Third-party information on consumer creditworthiness is frequently unavailable to the majority of financial institutions who are just entering the market. Using real customer data gathered from banks and financial organisations and showing whether the loans granted to them were successful or not, this research tries to evaluate credit risk using a combination of machine and deep learning classifiers.

Lack of a credit history is a major barrier in lending markets when determining a borrower's credit worthiness and, consequently, reasonable interest rates. In this context, there is a worldwide rivalry between banks for market dominance and rivalry to establish a competitive edge. High levels of non-performing loans have been seen to be detrimental to banks and financial institutions.

In order to make loan choices for businesses and retail consumers, borrowers have historically depended on credit ratings. Credit scoring models may be built using historical data on transaction and payment history from financial institutions to estimate/calculate these credit scores, and by employing credit scoring and a profit-risk analysis, financial institutions can be maximised.

Thus, one of the most common application areas for both data mining and operational research methodologies is credit scoring. Future savings can be considerable if the credit decision's scoring accuracy is increased by even a small percentage.

Inaccurate customer classification can have significant financial and reputational repercussions. Dataset building, modelling, and documentation are the three key processes that make up the production of a conventional credit scorecard. While the main goal of this project is to concentrate more on the dataset development and modelling portion, significance is also placed on results analysis and the creation of a suitable risk and profit model.

1.2 RESEARCH GAP

While there is an abundance of published research and implementations for credit scoring, there are very few implementations that address the low-default portfolio problem by addressing huge output class imbalances. And many of the implementations have only considered small datasets, with fewer than 10,000 instances.

Another common thing in most of the published works is the usage of the publicly available credit datasets from the UC Irvine machine learning repository, due to the difficulty in getting actual real-world financial data of real customers and borrowers.

Thus, a need for more implementations that use a different dataset and address the principle issues of a low default portfolio and a small dataset is required.

While there is a lot of published work predicting the credit class or rating of borrowers, there is a need for more implementations that address the profitability aspect of the financial institutions. The banks usually decide on who to lend to based on these credit ratings, but a more organised method of deciding who to lend to is required to actually make a profit.

1.3 RESEARCH CHALLENGES

The low-default portfolio problem is a prevalent issue in credit assessment caused by the usual absence of defaulters in the real world. Due to this, the majority of datasets used for credit scoring are unbalanced, and they may need to be corrected.

Customers' credit scores that are only based on past payments and transactions may not always be accurate. As a result, behavioural scoring, another type of credit scoring, may be useful in this situation. After credit has been approved, behavioural scoring is used to determine how likely it is for an existing client to default within a specific time frame. Although this aspect won't be properly explored in this thesis,

Most publicly available credit scoring databases aren’t big enough to get the real-world picture. To address this, larger datasets from other private sources can be obtained, or artificial synthetic data can be used to supplement the current, small dataset.

There is a lack of well-established methods to calculate the estimated losses and revenue of each borrower present in credit rating datasets, and generally standard default loss and profit values are assumed for credit risk-profit analysis. So to calculate these estimated profit and loss values, a proper methodology has to be devised.

1.4 PROBLEM STATEMENT

The primary objective is to develop a suitable machine and deep learning ensemble model for predicting credit ratings of potential borrowers from a financial institution or bank using historical credit data of previous customers.

The outcome of the model will be used to calculate the probability of default for each customer, estimate the expected loss of a ’bad’ loan and the expected revenue from a ‘good’ loan, and present a well-examined credit profit-risk analysis for the lending institution/bank using suitable estimation formulae.

This study aims to examine the precision and specificity of several machine learning algorithms and show how well these algorithms can categorise consumers into those who can repay the loan, or "good" customers, and those who won't, or "bad" customers.

After classifying the customers, the probability of default is found for each borrower and sorted in ascending order and split into percentile groups. Next, each customer's potential losses or profits is estimated using suitable formulae, and all the customer’s net loss or profit is calculated for each percentile. Finally, these results will be provided to the bank, and they can make the decision to lend to customers up to whichever percentile group they see fit.

1.5 RESEARCH OBJECTIVES

The core objectives of this research starts with first to find a suitable credit rating balanced dataset with adequate number of attributes and sufficient number of records. Then, if the above mentioned dataset has less number of records, the number of records should be increased by creating artificial or synthetic data.

Next, for the dataset to be usable for machine learning process, the data should be pre-processed using suitable data pre-processing techniques in such a way that the dataset is not imbalanced, thus avoiding the the low-default portfolio problem.

Then, feature engineering should be done using the relevant algorithms, to reduce the number of attributes and select only the most consequenctial ones. This prevents the machine learning model to not overfit to less usefull attributes.

Next is to run the processed dataset using different machine learning and deep learning algorithms as the base learners to a an ensemble stacking model and compare their accuracies to choose the best one.

Use the outcome of the model to calculate estimated losses and profit revenue for each borrower and ultimately make a credit risk-profit model for the financial institute to make a profitable decision.

Chapter 2

Literature Survey

In Thesis [1], K. Kennedy provides a thorough and long, detailed explanation and implementation of credit scoring and its related machine learning techniques and challenges. It starts off by analysing how various supervised machine classification approaches perform on a variety of unbalanced credit datasets. This paper provided a good introductory gateway into the world of credit scoring and analysis using machine learning techniques and algorithms and what problems currently persist in the study up to the year of its publication.

Thesis [1] also provides a very detailed exposure to one of the most prevalent issues present in credit scoring, which is the 'Low Default Portfolio Issue, where the number of 'good' loans usually always exceeds the number of 'bad' loans in any credit scoring dataset, causing a class imbalance. Thus, this study goes on to contrast the effectiveness of many semi-supervised classification techniques with that of logistic regression to deal with this issue. It draws the conclusion that both techniques warrant attention when working with low-default portfolios based on a thorough evaluation of their effectiveness. Some of the relevant algorithms used in thesis [1] include various 2-class machine learning classifiers, 1-class machine learning classifiers, Oversampling techniques etc.

Thesis [1] further determines how different implementations of a real-world behavioural score dataset affect classifier performance in credit scoring. But this part is not further elaborated in this thesis. Thesis [1] also shows how fictional data may be used to get around the challenges of collecting and utilising real-world data. Additionally, the drawbacks of using generated data to assess the performance of the classifier are underlined. As this thesis is very outdated, some more future papers have to be addressed to check how future works have improved on the problems stated in thesis [1].

In their paper [2], the authors construct a collection of machine learning models with the goal of creating an accurate credit rating prediction system with very good output results. It explores the latest ML/AI principles, beginning with natural language processing applied to text data of major economic issues, utilising embedding, AE, and gradient boosting machines. This paper also uses a very unique approach of using genetic algorithms, which borrow ideas related to genetics and Darwinian evolution, to give credit ratings. In [2], models may be made more understandable by utilising excellent, explaining approaches like SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME), which describe predictions locally in the space of features.

Some of the relevant algorithms used in this paper [2] include Light-GBM, Logistic Lasso Regression, Pearson's Criterion, Chi-Square Criterion, Random Forest Classifier, and Genetic Algorithms. One of the most important perks of this study is the usage of a very large dataset, which prevents the issue of overfitting and helps the machine learning models predict the test cases better. A major con of this paper was that it did not sufficiently address the issue of the low default portfolio or the output class imbalance.

In paper [3], the authors use data from a microlending institution to build multiple two-class classifier machine learning models and compare their accuracies. This offers low-cost and trustworthy ways for microlending organisations in developing nations to evaluate borrower creditworthiness in the absence of centralised historical credit databases. Some of the relevant algorithms used in the paper are AdaBoost, XGBoost, Decision Tree, Extra Trees, Random Forest, K-NN, and Neural Networks.

This paper [3] addresses the most important issue in credit scoring, which is a low default portfolio. This study addresses the class imbalance problem, where mostly 'poor' loans dominate the output class, by generating synthetic data for 'good' and 'average' loans. This is achieved by the Synthetic Minority Oversampling Technique algorithm, or SMOTE.

Some major cons of this paper [3] are the issue of failing to explore the option of using one-class machine learning classifiers, which could yield good results themselves, as cited in the thesis [1]. The other issue was the low output accuracy of the machine learning models used (the highest being 81.2%), which could have been significantly higher. This is a potential improvement to this paper's work.

Paper [4] makes use of one of the most commonly used dataset for credit scoring, the German Credit dataset by UC Irvine machine learning repository. This dataset is used to build models using common two-class machine learning classifiers by avoiding overfitting by utilizing k-fold cross validation with 10 as k value. This approach is often used to evaluate and choose a model for a specific predictive modelling challenge.

This paper [4], places a lot of emphasis on model perfomence measure of their machine learning models, particularly the usage of Area Under ROC Curve or AUROC. Some of the relevant and unique algorithms used in this paper are Lasso regression, Support Vector Machine, Logit regression and AUROC. A major con of this study is the usage of the German Credit Dataset, whose small size only includes 1000 instances. The study has failed to address this issue by not making any approaches to increase the size by adding synthetic/artificial data.

Paper [5] gives a thorough analysis of more than 200 studies, books, theses, research papers, and articles in credit scoring. The survey provides an extensive look at credit scoring and all its aspects, from its origins, definitions, and need to its methodologies, usage, etc., involving sources from multiple papers. The study also looks at various credit scoring techniques and methodologies, including statistical, machine learning, deep learning, and judgmental approaches. One of the main arguments the authors provide is why credit scoring is a better method to judge borrowers than the traditional judgmental method.

The study [5] also showcases the best machine learning model accuracy achieved by various other research works on some of the most commonly used and publicly available credit datasets, such as the German and Australian credit scoring datasets by UC Irvine machine learning repository. This gives anyone new to the credit scoring scene a very good idea of where there is a research gap and which datasets have potential for improvement.

Paper [6] analyses the current personal credit loan evaluations in Egypt using data from commercial public sector banks in Egypt and provides a more efficient method. It examines the current decision-making process used in the Egyptian public sector and then considers whether applying credit scoring models can substantially improve the process. As this is a relatively old paper (2009), this is one of the first papers to analyse why the currently used approach, based on personal judgement, is not efficient and practical enough. This paper provides evidence on how statistical scoring techniques are shown to provide more efficient classification results than the currently used judgmental techniques.

The algorithms used in this paper [6] are pretty advanced for a study dated before the 2010s. Some of the more relevant ones are multiple discriminant analysis, logistic regression, probabilistic neural networks, and multi-layer feed-forward networks.

Though paper [6]'s research theoristically proves why credit scoring techniques using statistical and machine learning techniques are far superior to the judgmental system, the model's in-world working is in question because a developing nation such as Egypt's lending policies usually involve other factors, such as field visits, to ensure a more accurate way to provide credit scores. Thus, in this case, the judgmental system to provide credit rating shouldn't be completely abandoned, and a model involving both methodologies' methods should be adopted for Egypt.

Thus, papers [5] and [6] provide a convincing explanation as to why credit scoring using machine learning and statistical methodologies/algorithms is far better than the older judgmental methods involving physical checks such as field visits.

In this paper [7], the authors use a large dataset consisting of 661 publicly traded firms from eight regions to develop a deep learning approach to credit scoring. The main deep learning algorithms used here are multilayer perceptrons, deep belief networks using restricted Boltzmann machines, and DBN, where more emphasis is placed on deep belief networks.

To evaluate the accuracy of the deep learning models, these algorithms are compared with other popular machine learning classification algorithms generally used for credit scoring, such as logistic regression and support vector machines (SVM). One of the pros of this paper is the dataset's huge size, which kind of partially solves the low default portfolio issue by avoiding overfitting of the model in favour of the class having the fewer instances. The highest accuracy was recorded by DBN, which recorded more than the above-mentioned machine learning classification algorithms.

Paper [8] uses various deep learning models to evaluate more than 1700 IT companies' credit ratings in India provided by CRISIL. Some of the relevant deep learning methods utilised here were multilayer perceptrons, convolutional neural networks, and long short-term memory, with multilayer perceptrons providing the highest accuracy.

A unique aspect of paper [8]'s dataset is the usage of up to eight classes for classifying the credit rating of companies instead of the more prevalent two-class binary bad and good loan system. Though the source of the dataset has been mentioned, not much about the details regarding the attributes or number of records is known.

Thus, we have seen two consequtive papers ([7] and [8]) showcasing the proficiency of deep learning algorithms in credit scoring over more popular machine learning classification algorithms, thus validating the use of deep learning algorithms in my research.

Paper [9] makes use of two classes of machine learning and deep learning models to predict a binary credit rating for customers of a bank. These models are finally used to predict the probability of default, which is the probability by which a customer or borrower fails to pay back his loan.

An important pro of this paper [9] is the usage of a huge dataset with about 117 thousand instances, which is a good way to tackle the low-profile default issue and overfitting. To simplify and improve the model's prediction, features are reduced to ten from more than a hundred and the process is repeated.This paper showcases how to handle big data and proves how having a huge dataset can significantly increase the model's accuracy.

Some of the relevant algorithms used in the paper [9] include Elastic Net, Random Forest, Gradient Boosting Machine, Neural Networks, and SMOTE.

Paper [9] ultimately concludes that models based on multilayer artificial neural networks are less stable than models based on trees, such as the decision tree algorithm and the random forest algorithm, having secured better accuracy than other deep learning algorithms, thus creating a direct contradiction to the conclusions of papers [7] and [8].

Paper [10] makes use of publicly available UC Irvine machine learning repository datasets from Japan, Australia, and Poland to propose a two-layer stacking ensemble machine learning approach. The primary layer of the stacking model includes five of the most widely used techniques for credit scoring, including decision trees, support vector machines, random forests, XGBoost, etc. These algorithm outputs are stacked together in a two-layer ensemble model.

Paper [10] also includes class and attribute noise reduction methods, which pre-process the datasets in a better way, without noise, for training the models accurately. Further, a novel backflow learning strategy is suggested so that the base classifiers can relearn the incorrectly classified data point, enhancing the power and variety of the basic classifiers.

The final output of paper [10]’s research, utilising the ensemble stacking model, exceeds the individual accuracies of the base primary models.

Paper [11] makes use of a a multi-layer ensemble model with improved outlier adaption is presented. To detect potential outliers and then boost them back into the training dataset to create an outlier-adapted training set that improves the outlier adaptability of primary base classifiers, a local outlier factor algorithm is enhanced with the bagging strategy. This reduces the negative effects of outliers present in the noise-filled credit datasets.

Paper [11] further implements dimension reduction of the dataset by implementing feature reduction using suitable techniques. To assess the effectiveness of the suggested methodology, 10 publically available credit datasets from UC Irvine machine learning repository, are examined using 6 evaluation markers.

This paper is different from [10] in implementing multiple layers of stacking ensemble algorithm as opposed to just two layers and yielded convincing accuracies. The base leaarners include XGBoost, AdaBoost, Random Forest, Decision Tree etc. Thus, we can see both [10] and [11] using the same base learning algorithms.

Papers [12] give an evaluation of the performance of the ensemble machine learning approaches, including Random Forest, AdaBoost, Gradient Boosted Decision Tree, Extreme Gradient Boosting, and Stacking. This research compares how these above-mentioned ensemble techniques work as the secondary layer in a stacking ensemble setup with a sufficient number of base classification algorithms. The objectives of this paper differ from papers [10] and [11].

The study [12] used a credit dataset from a U.S. Lending Club that represents real-world credit data instead of publicly available and commonly used credit datasets from the UC Irvine machine learning repository.

Among the base learners, logistic regression gave the best performance, and among the ensemble methods, random forest gave the highest overall accuracy score. The output accuracies are evaluated using multiple evaluation methods, including the area under the ROC curve and the Brier score [12].

Thus, from papers [10], [11], and [12], a rough idea of an ideal implementation of an ensemble model can be constructed, knowing what the best base learning (primary layer) and stacking (secondary) algorithms are. Furthermore, from papers [7] and [8], usage of deep learning algorithms as supervised learners can also be considered.

Paper [13] tackles the absence of positive samples, which is one of the most pressing problems in the field of credit rating and is also known as the low profile fault problem. In order to create enough default transactions in the origin data, this research has developed the concept of conditional tabular generative adversarial networks (CTGAN).

The CTGAN method is used to increase the size of datasets with very few records. This algorithm adds synthetically generated records into the dataset, taking into consideration the nature and type of data for each attribute. Increasing the dataset helps train the base learning models better by offering more data to find patterns upon. This method also simultaneously addresses the issue of class imbalance.

The datasets used in the research [13] include those from a Chinese financial institution that is anonymous and that has more than 15,000 client records and a publicly available dataset from the UC Irvine machine learning repository that has 30k client records. The model is trained and tested on both of these algorithms after synthetically increasing the dataset size using CTGAN.

[13]'s research further makes use of CNN-ATCN, another deep learning algorithm, for feature engineering and reduction. Once again, for stacking the algorithms, use the common base learners such as logistic regression, extreme gradient boosting, random forest, ADABoost, etc.

Paper [14]'s primary objective is to tackle the low profile default or the output class imbalance issue in most credit scoring datasets. There are substantially fewer bad output records than good output records. So to tackle this issue, the number of bad output class instances are increased. Instead of following the most popular oversampling method, by creating these new records sythtically using Synthetic Minority Oversampling Technique or SMOTE, this implementation uses the technique of bagging which is implemented via bootstrapping. While this method can maintain the originality of the data by recycling th existing values randomly, unlike SMOTE, it runs the risk of overfitting the model.

After tackling the class imbalance problem, the rest of the implementation is a fairly normal ensemble stacking implementation using some common base learners including Logistic Regression, K-Nearest Neighbor, Decitions Tree etc. This work [14] also makes use of the of the famous publically available credit datasets of German, Chile and Default from the UC Irvine machine learning repository.

Paper [15], in order to achieve better performance and great resilience, this work attempts to create a unique ensemble model for credit scoring that can be tailored to various imbalance ratio datasets. Initially, the proposed method improves on the BalanceCascade technique to produce changeable balanced subsets based on the imbalance ratios of training data in accordance with the peculiarities of the credit scoring data.

The suggested approach uses random forest and extreme gradient boosting as the foundational classifiers for a three-stage ensemble model. The majority of assessment metrics for various datasets show that the suggested model performs, on average, better than other comparable methods.

Chapter 3

Proposed Work

3.1 DATASET

The dataset chosen was SAS Enterprise Miner's Credit Risk Dataset, which was accessed from a case study conducted by professor Min-Yuh Day of Tamkang University. This two-class dataset only has 3000 instances; it can be increased by using synthetic artificial data, and the final size would be much larger. The dataset has 30 attributes, which is always good for training machine learning models.

The most attractive part of the dataset is that it has all the necessary attributes to calculate estimated losses and revenue of bad and good loans, respectively, thus avoiding having to assume the loss and revenue of each bad and good loan.

VarID	Name	Model Role	Measurement Level	Description
1	BanruptcyInd	Input	Binary	Bankruptcy Indicator
2	CollectCnt	Input	Interval	Number Collections
3	DerogCnt	Input	Interval	Number Public Derogatories
4	ID	Input	Nominal	Applicant ID
5	InqCnt06	Input	Interval	Number Inquiries 6 Months
6	InqFinanceCnt24	Input	Interval	Number Finance Inquires 24 Months
7	InqTimeLast	Input	Interval	Time Since Last Inquiry
8	TARGET	Target	Binary	1=Bad Debt, 0=Paid-off
9	TL50UtilCnt	Input	Interval	Number Trade Lines 50 pct Utilized
10	TL75UtilCnt	Input	Interval	Number Trade Lines 75 pct Utilized
11	TLBadCnt24	Input	Interval	Number Trade Lines Bad Debt 24 Months
12	TLBadDerogCnt	Input	Interval	Number Bad Dept plus Public Derogatories
13	TLBalHCPct	Input	Interval	Percent Trade Line Balance to High Credit
14	TLCnt	Input	Interval	Total Open Trade Lines
15	TLCnt03	Input	Interval	Number Trade Lines Opened 3 Months
16	TLCnt12	Input	Interval	Number Trade Lines Opened 12 Months
17	TLCnt24	Input	Interval	Number Trade Lines Opened 24 Months
18	TLDel3060Cnt24	Input	Interval	Number Trades 30 or 60 Days 24 Months
19	TLDel60Cnt	Input	Interval	Number Trades Currently 60 Days or Worse
20	TLDel60Cnt24	Input	Interval	Number Trades 60 Days or Worse 24 Months
21	TLDel60CntAll	Input	Interval	Number Trade Lines 60 Days or Worse Ever
22	TLDel90Cnt24	Input	Interval	Number Trade Lines 90+ 24 Months
23	TLMaxSum	Input	Interval	Total High Credit All Trade Lines
24	TLOpen24Pct	Input	Interval	Percent Trade Lines Open 24 Months
25	TLOpenPct	Input	Interval	Percent Trade Lines Open
26	TLSatCnt	Input	Interval	Number Trade Lines Currently Satisfactory
27	TLSatPct	Input	Interval	Percent Satisfactory to Total Trade Lines
28	TLSum	Input	Interval	Total Balance All Trade Lines
29	TLTimeFirst	Input	Interval	Time Since First Trade Line
30	TLTimeLast	Input	Interval	Time Since Last Trade Line

Fig 2: Dataset Used

The dataset's attributes and its descriptions have been mentioned above. 'TARGET' is the output class, which only has two possible outcomes, 0 or 1. 0 is classified as a good loan, indicating that the corresponding customer will successfully pay back the borrowed amount, while 0 is classified as a 'bad' loan, indicating the corresponding person will fail to pay back the borrowed amount.

3.2 PROPOSED ARCHITECTURE

First, the credit risk dataset is chosen, and synthetic data is produced using suitable methods to increase the size of the dataset using synthetic data. Next, the dataset is pre-processed, and the dataset is balanced using up-sampling or under-sampling methods using either SMOTE or bootstrapping. Next, feature engineering is done, and only the most impactful attributes are chosen using mutual information gain algorithm.

Next, a multitude of machine learning techniques, including deep learning, are chosen as the base learners of an ensemble model and are stacked together with a stacking algorithm. Then the training dataset is run through these models, and their accuracy is evaluated. The model with the highest accuracy will be chosen as the final model to get the outcomes from and will be used to calculate the probability of default values.

Finally, a risk-profit model is created using the outcome using decile or percentile methods. Where the estimated loss in revenue is calculated if the borrower stops borrowing at this instant and the estimated profit in revenue is calculated for each of the good loans.

Then, the loans are sorted in ascending order of probability of default and split into hundread equal parts (if the percentile method is adopted) or ten equal parts (if the decile method is adopted). Then, if the predicted outcome is good, the profit value is taken, and if the predicted outcome is bad, the loss value is taken and added together to generate a net revenue value for each percentile or decile. Finally, the financial institute is advised to give out loans up to the percentile or decile at which the revenue remains positive. The architectural diagram below demonstrates this clearly.

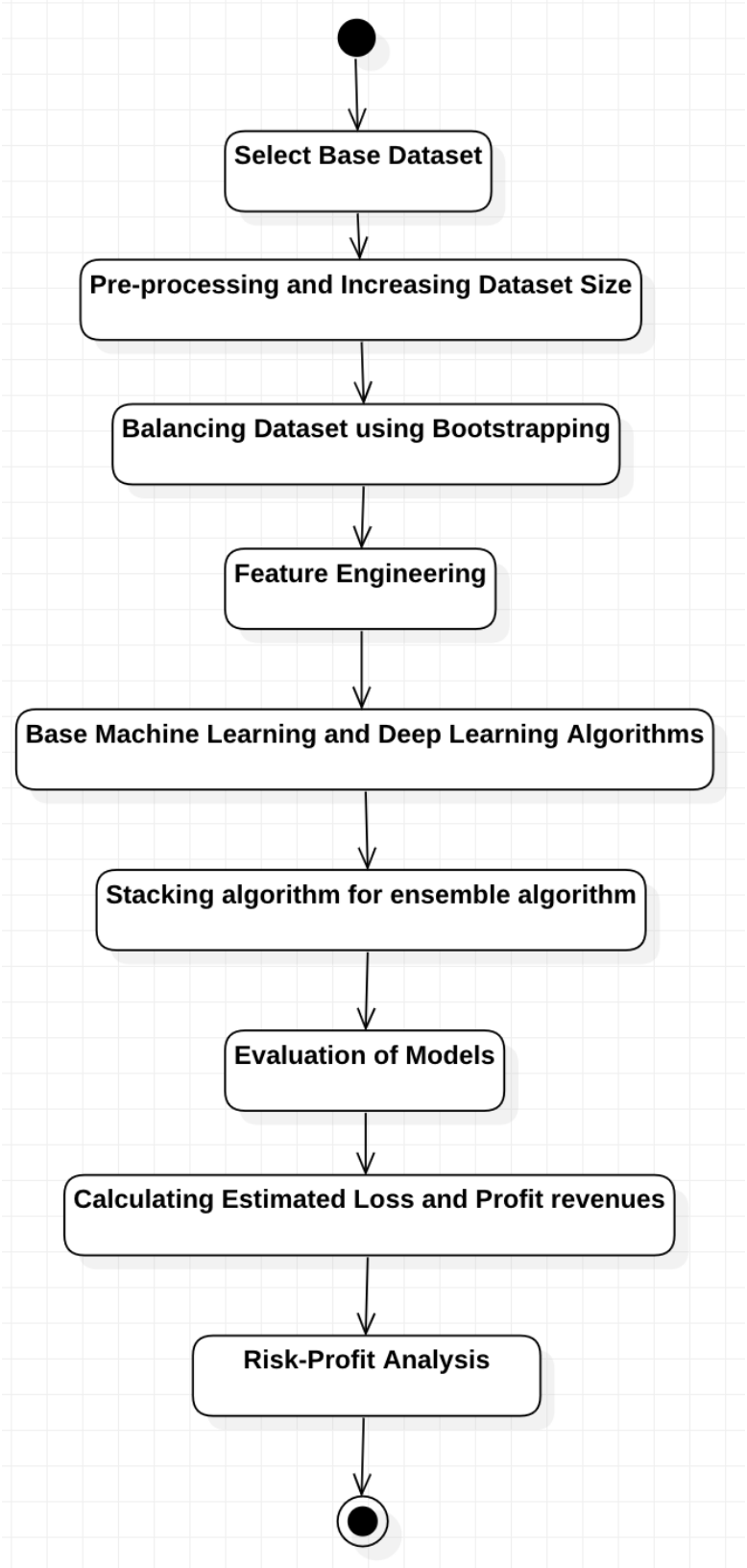


Fig 3: Architectural Diagram

3.3 PRE-PROCESSING DATASET

First, to make sure the dataset doesn't have any NULL values, I start by analysing the value distribution of attributes with missing values. The distribution of almost all the attributes has outliers, which can definitely affect the mean value; hence, the median value is chosen for missing data imputations.

Next, the dataset size is increased by using CTGAN, which is one of the deep learning synthetic data generators for single table data that can draw knowledge from the original data and produce highly accurate artificial data.

Up next, to counter the low-default portfolio problem, the class imbalance has to be mitigated. So two of the most prominent oversampling methods are used, and we choose the one that yields the best accuracy. The first method is Synthetic Minority Oversampling Technique, or SMOTE, which synthetically creates new data based on the trends and design of the old one. The next method is bootstrapping, which randomly chooses attribute values to create new row values. If any row is exactly repeated, it is promptly deleted to avoid overfitting.

3.4 FEATURE ENGINEERING

The next step is to reduce the number of features by reducing lesser correlated features using correlation coefficient analysis. In this model, feats that are highly related to each other contribute the same value to training the model. So to avoid overfitting, one of them is removed. But the pre-requisite for using this method is that the feature values should be normally distributed, which should be checked priorly.

The second feature-reducing method is via the mutual information gain value, which is found for each attribute, and only the top ten attributes are chosen for further processing. The information gain value denotes how much an attribute affects the outcome class of the instance. Therefore, removing the less significant features or attributes is important in training the model better.

Mutual information, which measures the level of knowledge gained about one random variable through the other random variable, is a measure between two random variables, X and Y. It is given by

$$I(X;Y) = \int_X \int_Y p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy,$$

Fig 4: Mutual Information Formula

3.5 MODELLING

As the outcome classes will be predicted using an ensemble method, the chosen machine learning algorithms for the base learners are logistic regression, support vector machines (using 4 different linear and non-linear type kernels), decision trees, random forests, and extreme gradient boosting (XGBoost).

As discussed in the literature survey, ensemble techniques and deep learning techniques yielded good accuracies in suitable conditions; therefore, the use of a deep learning model such as Multi-Layer Perceptron (MLP) as a base learner for the stacking algorithm will be beneficial, as dataset patterns not found by conventional algorithms could be recognised by a deep learning algorithm like MLP. Then, these base learners will be stacked with XGBoost again as the secondary layer stacking model, where the outputs of the base learners will be fed as inputs to XGBoost again to yield better accuracy.

Finally, basic performance measures such as the accuracy score, precision, recall, f1-score, support, macro average, and weighted average will be used to evaluate. These metrics will be divided for each output class as well as for the overall model. The models mentioned above are described in detail below:

3.5.1 Logistic Regression

In this classification algorithm, the weight vector w and the deviation term b are the two components of the linear discriminant analysis. The class label y is forecasted using the formula, given a sample x:

$$y = \text{sign}\left(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x} + b\right).$$

Using a logistic function to convert linear probabilities into logit and assuming that the variable y belongs to the set 0 to 1, logistic regression is a better approach for classification than regression since it assumes that the targeted variable y is a member of the set 0 to 1. This is one of the most popular classification algorithms used in credit scoring.

3.5.2 Decision Tree

It is a classification algorithm where a tree is constructed from the root to the leaves, which are the final classes. Each node, including the root, asks a simple true or false question to divide the entire dataset into two halves, and then continues to divide it until each instance is classified into its corresponding output classes. The decision-making nodes at the top are more influential in determining the output class than the nodes way below it.

3.5.3 Support Vector Machine

Support Vector Machine is a classification algorithm that works by plotting the data points in a multi-dimensional plane and categorising them by locating the best hyperplane. It is a significant margin classifier used to resolve two-class classification issues. Particularly, Support Vector Machine divides the data into multiple groups (two class groups in this case) and seeks to identify the highest classification margin on the given dataset as the decision border. To estimate the category of fresh samples, one may use the framework that is produced.

3.5.4 Random Forest

The method of building a prediction aggregation using a collection of decision trees that develop in randomly chosen sub-spaces of data is known as random forest. A random forest is a classifier made up of a number of tree-structured classifier algorithms, Bn(y, Am), where m = 1, 2, 3, etc. and Bn are independent random variables with identical distributions that are used to control the order of the subsequent cuts while creating the individual trees. The effectiveness of a random forest depends on the potency of each tree classifier and the degree of their interdependence.

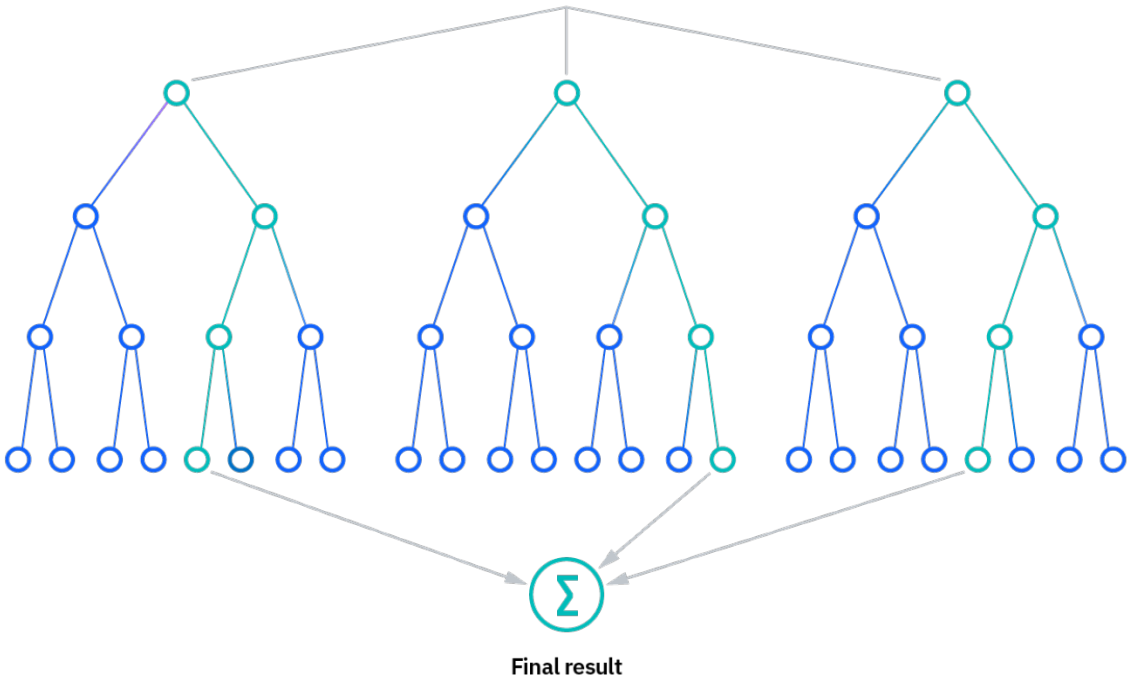


Fig 5: Random Forest Visualised

3.5.5 Naïve-Bayes

This is a probability based classification algorithm works upon the concept of conditional probablity, in particular the Bayes' theorm. This algorithm works on the basic premise that all the attricbutes in the dataset is independent of eacth other.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Fig 6: Naïve Bayes Formula

3.5.6 XGBoost

Extreme gradient boosting is an ensemble algorithm that is used both as a base learner and as the second layer classifier in this research's model. It differs significantly from gradient-boosted decision trees and provides numerous advantages over conventional gradient-boosting techniques. It usually comprises weaker base classifiers (usually decision trees), whose outputs are used as inputs to many 2nd-layer weak base classifiers. These distinct small classifiers are then combined to produce a robust and accurate model. This is one of the most popular ensemble methods used in machine learning.

3.5.7 Multi Layer Perceptrons:

This deep learning and feed-forward neural network algorithm is the multilayer perceptron. It has three prominent layersorward neural network algorithm is the multilayer perceptron. It has three prominent layers. The input is accepted by an input layer for processing. The prediction tasks are within the output layer's purview. There are an endless number of hidden nodes between the output and input layers that make up this algorithm. Here, data travels in a forward direction from the input layer to the output layer, similar to a network operating in feed-forward mode. All of the multilayer perceptron's nodes are trained using the backpropagation learning approach.

MLPs are constructed to approximate any continuous function and are capable of resolving problems that cannot be linearly separated. But this deep learning algorithm will be used in this two-classifier problem to find more trends and patterns in the training dataset that have been missed by other classical machine learning algorithms. The below diagram will help you visualise the setup of this algorithm more clearly.

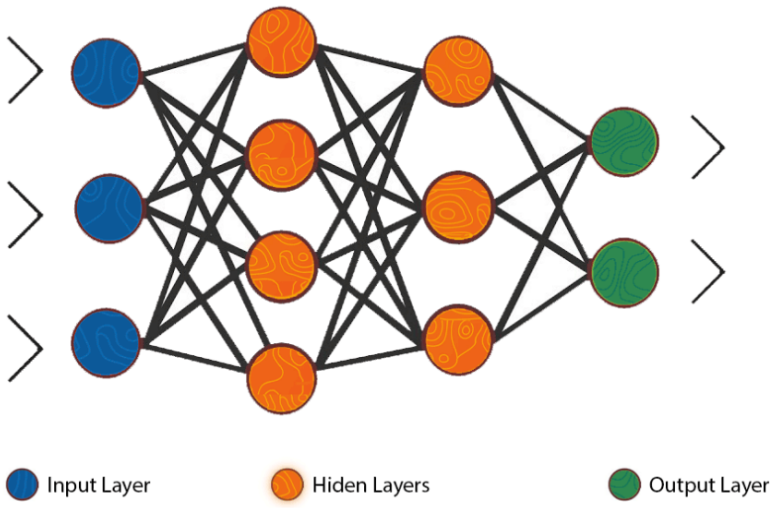


Fig 7: Multi Layer Perceptron Visualised

3.6 RISK-PROFIT ANALYSIS

Using the outcome of the model with the highest accuracy from the previous model, the probability of default (POD) is calculated, which is the probability that a borrower fails to pay back the loan or defaults on paying it back.

3.5.1 FROM THE GIVEN DATASET (FOR GOOD LOANS):

Estimated Revenue = TLMaxSum

3.5.2 FROM THE GIVEN DATASET (FOR BAD LOANS):

Exposure at Default (EAD) = TLSum

Recovery Rate (RR) = (TLMaxSum – TLSum)/(TLMaxSum)

Loss given default (LGD) = EAD * (1– RR)

Estimated Loss (EL) = POD * LGD

Using the estimated losses and revenues, we can use decile and percentile methods to analyse which set of people or up to which percentile or decile of people get granted loans to maximise revenue, as described above in the proposed system module.

Chapter 4

Results

4.1 PRE-PROCESSING OUTPUT:

4.1.1 DATA IMPUTATIONS

Value distribution of attributes with missing values, to figure out what type of imputation to fill out missing values. As we can see many outliers in each of the attribute, we choose median as mean values get skewed/biased by these extreme values.:

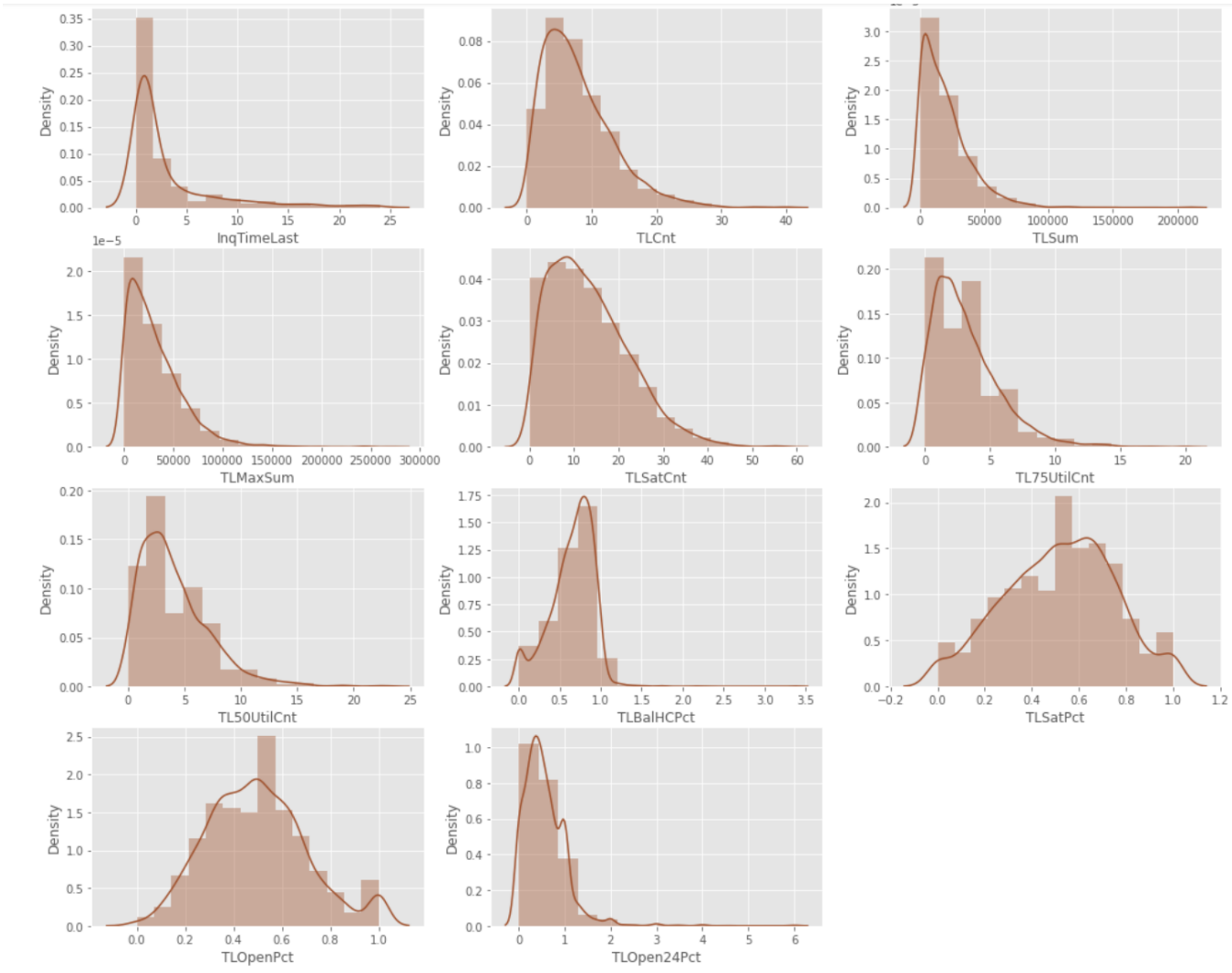


Fig 8: Data Imputation Graphs

4.1.2 CLASS DISTRIBUTION WITHOUT ANY AUGMENTATION:

As we can see, this dataset is imbalanced and is suffering from the issue of low-portfolio default. It needs to be balanced so that, the number of bad loans (1) should almost be equal to good loans (0).

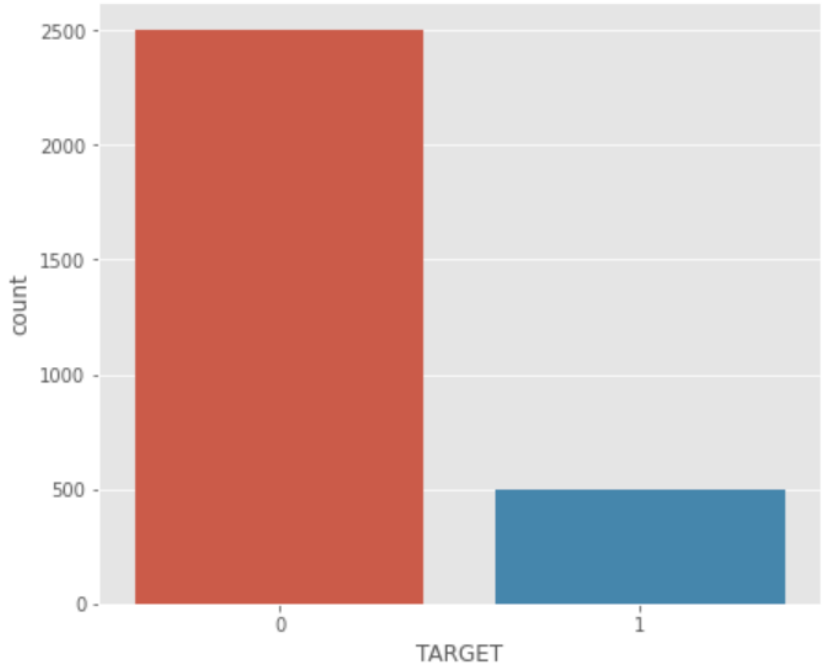


Fig 9: Class Imbalance

4.1.3 CLASS DISTRIBUTION AFTER APPLYING SMOTE:

Here, the number of good loans is reduced and the number of bad loans is increased by producing synthetic data and equalising the number of good and bad loans.

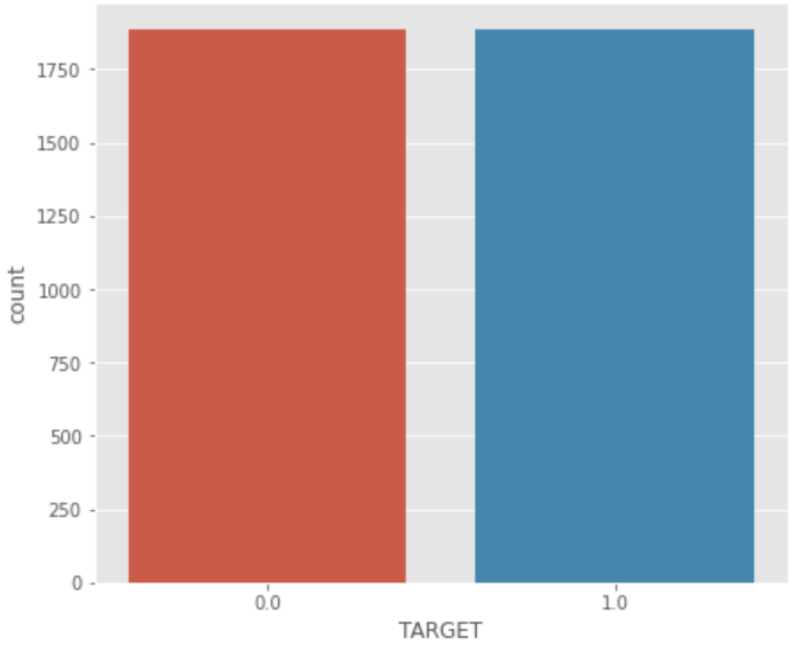


Fig 10: After SMOTE

4.1.4 CLASS DISTRIBUTION AFTER SYNTHETICALLY INCREASING THE DATASET SIZE USING CTGAN:

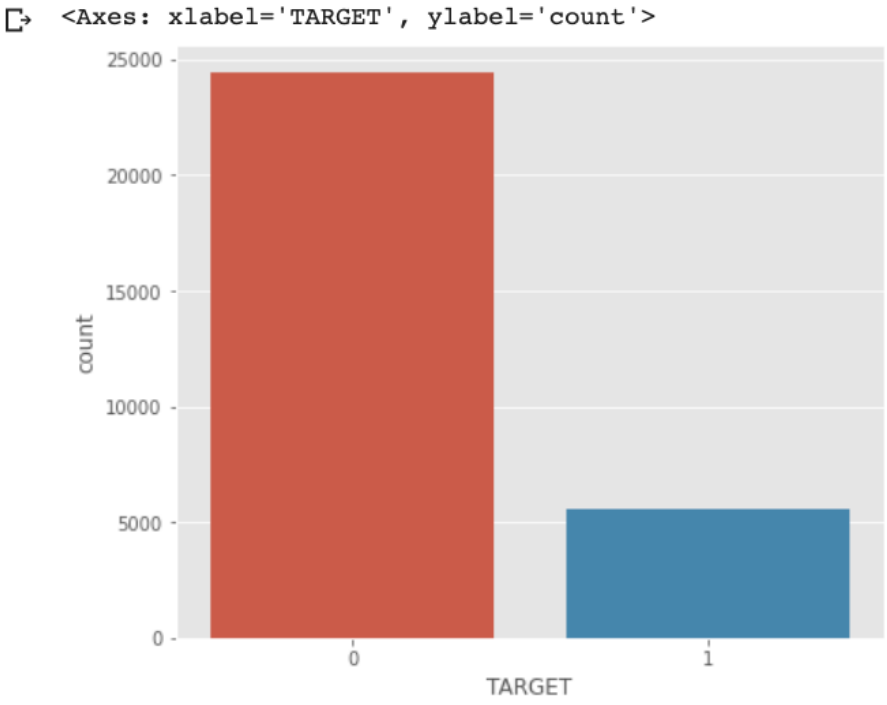


Fig 11: Sythetically Increasing Data

4.1.5 CLASS DISTRIBUTION AFTER APPLYING BOOTSTRAP METHOD TO BALANCE THE DATASET USING BAGGING:

Here, using the bootstrapping method, random values from already existing features are chosen and mix-matched together to form new instances, which are then added to emulate the bad loans output class. If any row gets exactly replicated, it is promptly removed to avoid overfitting.

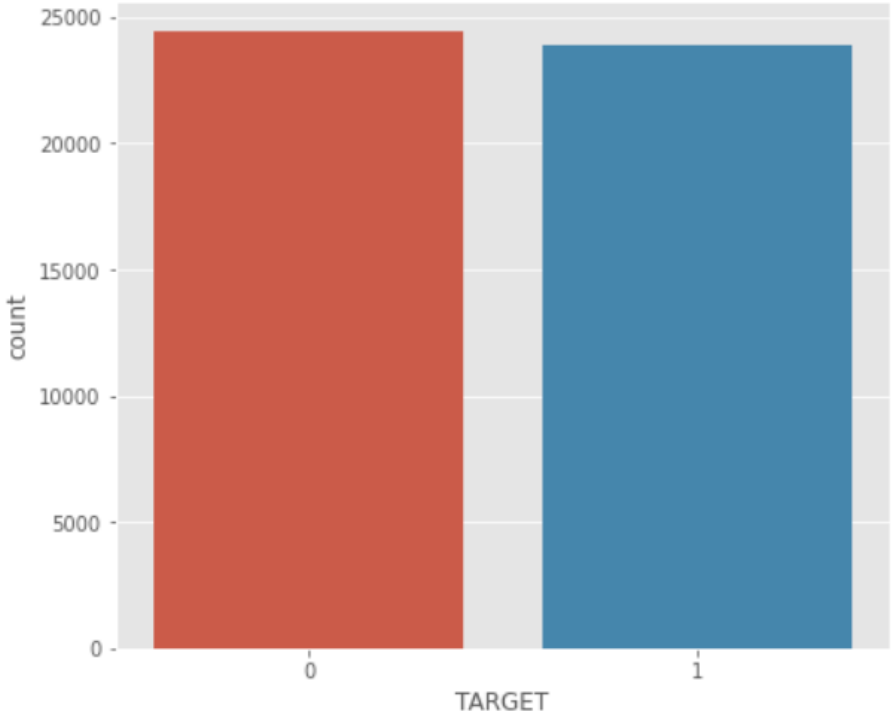


Fig 12: After Bootstrapping

4.1.6 MUTUAL INFORMATION GAIN VALUES FOR FEATURE ENGINEERING:

Here, we can see, the attributes on the left side of the chart having the higher mutual information value are more consequential in predicting the output value then the rest. The top ten attributes alone are chosen. This value is chosen based on the observation that many features end up contributing almost nothing to the outcome class.

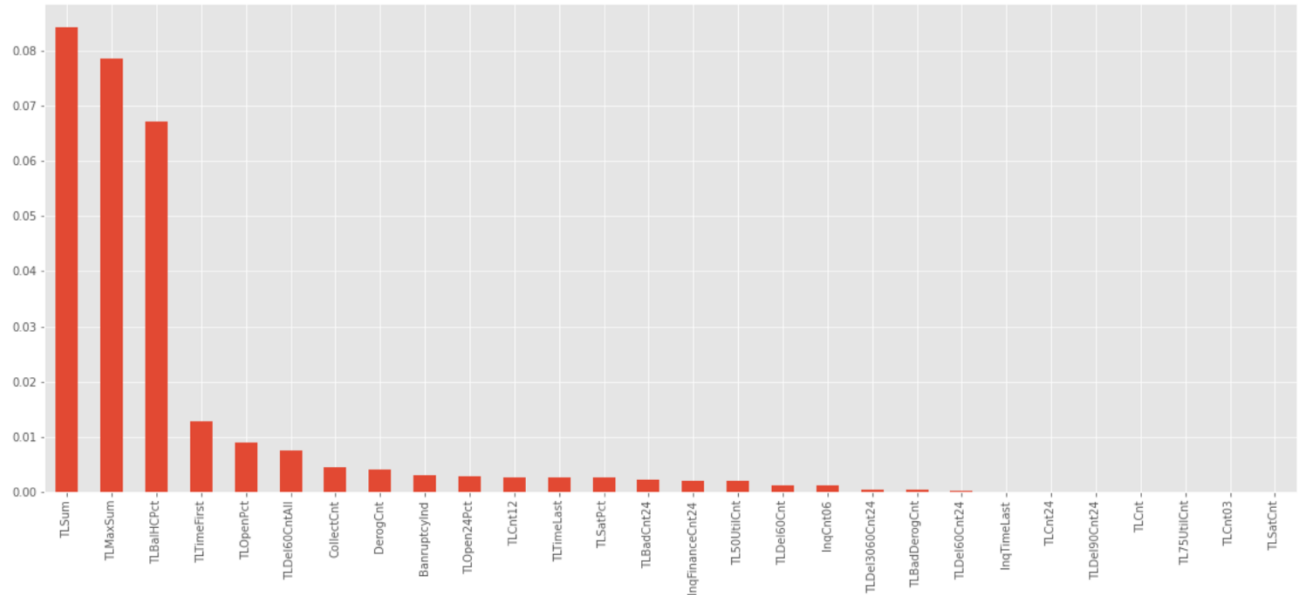


Fig 13: Mutual Information

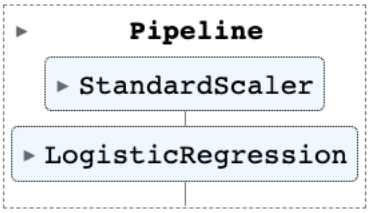
4.1.7 PEARSON CORRELATION CHART:

None of two attributes are too highly (>0.9) or too lowly correlated (<-0.9). So none of the columns area actually removed using this method.

Fig 14: Pearson Correlation

4.2 MODEL CLASSIFICATION REPORTS:

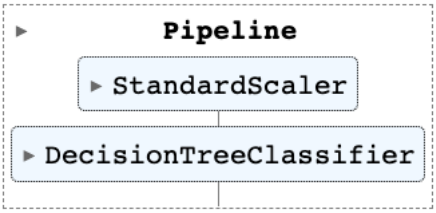
4.2.1 LOGISTIC REGRESSION:




```
Accuracy on Test Data:  50.59652029826015
```

	precision	recall	f1-score	support
0	0.60	0.51	0.55	7216
1	0.41	0.50	0.45	4854
accuracy			0.51	12070
macro avg	0.50	0.50	0.50	12070
weighted avg	0.52	0.51	0.51	12070

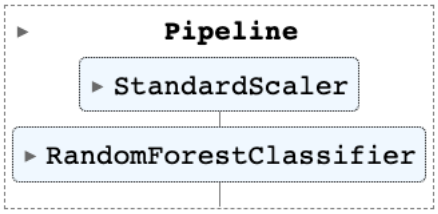
4.2.2 DECISION TREE




```
Accuracy on Test Data:  52.17067108533554
```

	precision	recall	f1-score	support
0	0.39	0.54	0.45	4426
1	0.66	0.51	0.57	7644
accuracy			0.52	12070
macro avg	0.52	0.53	0.51	12070
weighted avg	0.56	0.52	0.53	12070

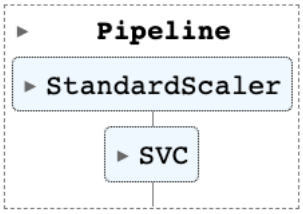
4.2.3 RANDOM FOREST



	precision	recall	f1-score	support
0	0.99	0.96	0.98	6358
1	0.96	0.99	0.98	5712
accuracy			0.98	12070
macro avg	0.98	0.98	0.98	12070
weighted avg	0.98	0.98	0.98	12070

4.2.4 SUPPORT VECTOR MACHINE

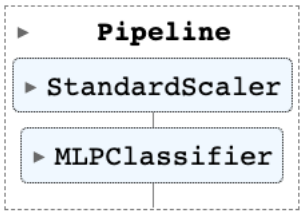
Radial Basis Function Kernel (Best Accuracy):



Accuracy on Test Data: 56.28831814415908

	precision	recall	f1-score	support
0	0.58	0.57	0.57	6267
1	0.54	0.56	0.55	5803
accuracy			0.56	12070
macro avg	0.56	0.56	0.56	12070
weighted avg	0.56	0.56	0.56	12070

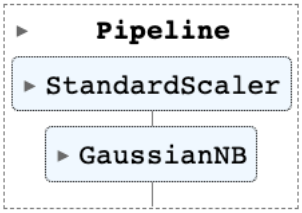
4.2.5 MULTI-LAYER PERCEPTRON



Accuracy on Test Data: 50.75393537696768

	precision	recall	f1-score	support
0	0.60	0.51	0.55	7211
1	0.41	0.50	0.45	4859
accuracy			0.51	12070
macro avg	0.51	0.51	0.50	12070
weighted avg	0.52	0.51	0.51	12070

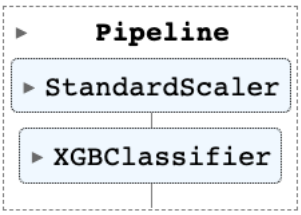
4.2.6 NAÏVE-BAYES



Accuracy on Test Data: 50.33140016570008

	precision	recall	f1-score	support
0	0.34	0.52	0.41	4042
1	0.67	0.50	0.57	8028
accuracy			0.50	12070
macro avg	0.51	0.51	0.49	12070
weighted avg	0.56	0.50	0.52	12070

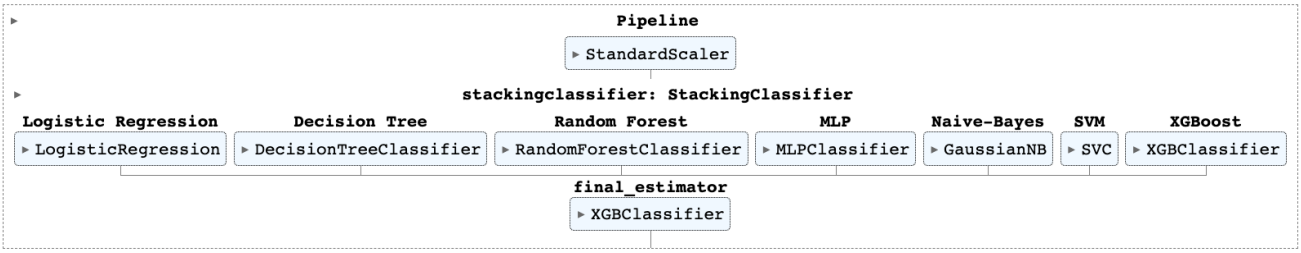
4.2.7 XGBOOST:



Accuracy on Test Data: 75.10356255178128

	precision	recall	f1-score	support
0	0.71	0.78	0.74	5606
1	0.79	0.73	0.76	6464
accuracy			0.75	12070
macro avg	0.75	0.75	0.75	12070
weighted avg	0.75	0.75	0.75	12070

4.2.8 ENSEMBLE - STACKING CLASSIFIER:



Accuracy on Test Data: 97.73819386909693

	precision	recall	f1-score	support
0	1.00	0.96	0.98	6386
1	0.96	1.00	0.98	5684
accuracy			0.98	12070
macro avg	0.98	0.98	0.98	12070
weighted avg	0.98	0.98	0.98	12070

4.3 FINAL ACCURACY GRAPHS:

4.3.1 WITHOUT DATA AUGMENTATION:

```
ACCURACIES:
Random Forest          --> 0.832
Logistic Regression    --> 0.8293333333333334
Decision Tree          --> 0.8133333333333334
SVM                    --> 0.8226666666666667
MLP                    --> 0.8213333333333334
Naive-Bayes            --> 0.7773333333333333
XGBoost                --> 0.8173333333333334
Stacking               --> 0.8093333333333333
```

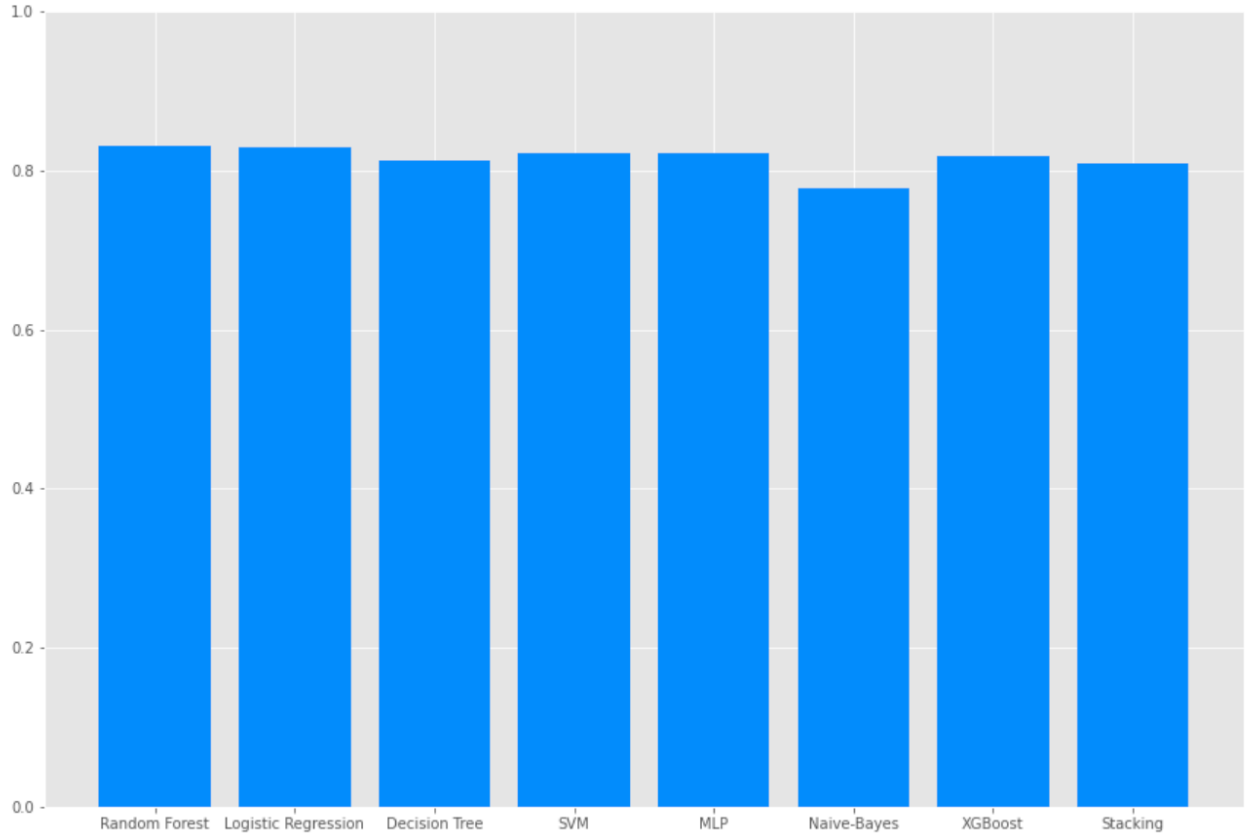


Fig 15: Accuracies 1

4.3.2 AFTER SMOTE UPSAMPLING:

```
ACCURACIES:
Random Forest          --> 0.788
Logistic Regression    --> 0.7053333333333334
Decision Tree          --> 0.7653333333333333
SVM                    --> 0.7133333333333334
MLP                    --> 0.6906666666666667
Naive-Bayes            --> 0.7266666666666667
XGBoost                --> 0.796
Stacking               --> 0.788
```

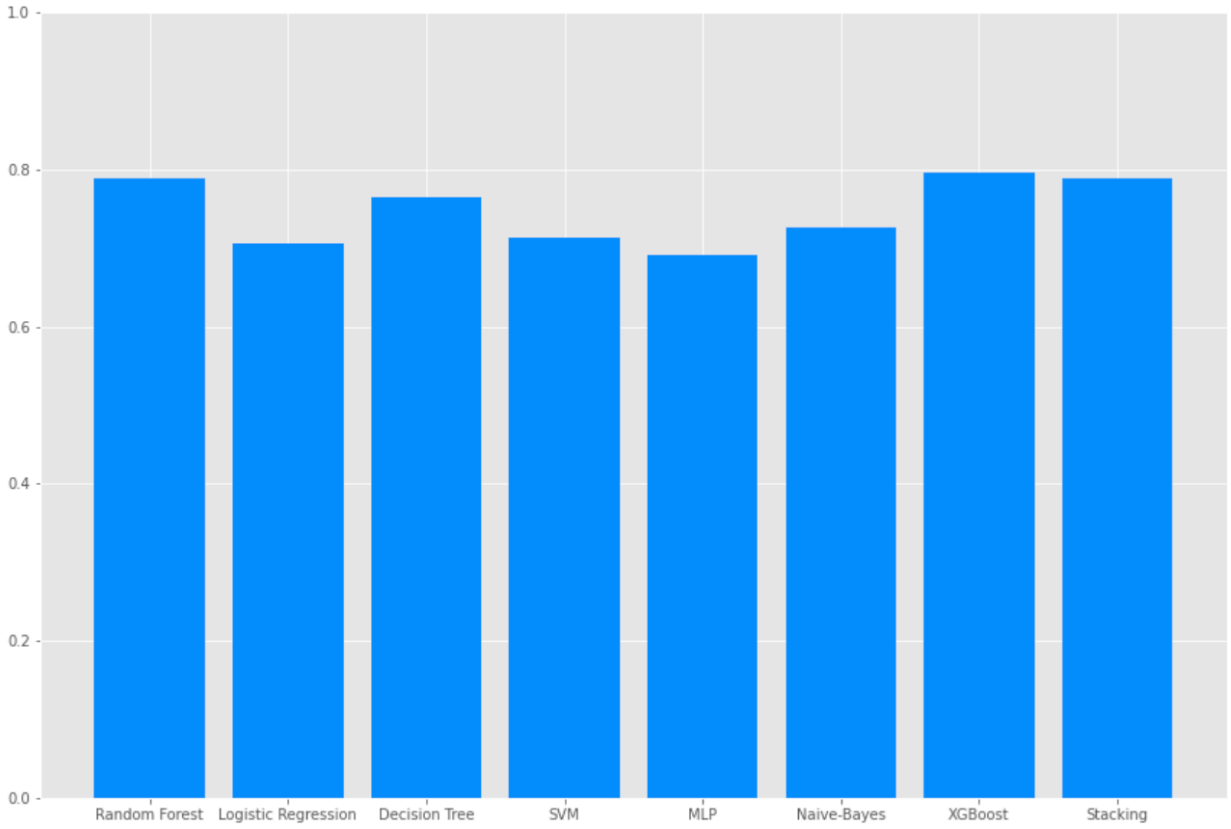


Fig 16: Accuracies 2

4.3.3 AFTER BOOTSTRAPPING UPSAMPLING AND DATASET SIZE INCREASE:

```
ACCURACIES:
Random Forest        --> 0.983676482651136
Logistic Regression  --> 0.511935034041506
Decision Tree        --> 0.5554917562136003
SVM                  --> 0.5808383233532934
MLP                  --> 0.5182511688950865
Naive-Bayes          --> 0.5116069231400213
XGBoost              --> 0.7617914855221065
Stacking             --> 0.9855631203346731
```

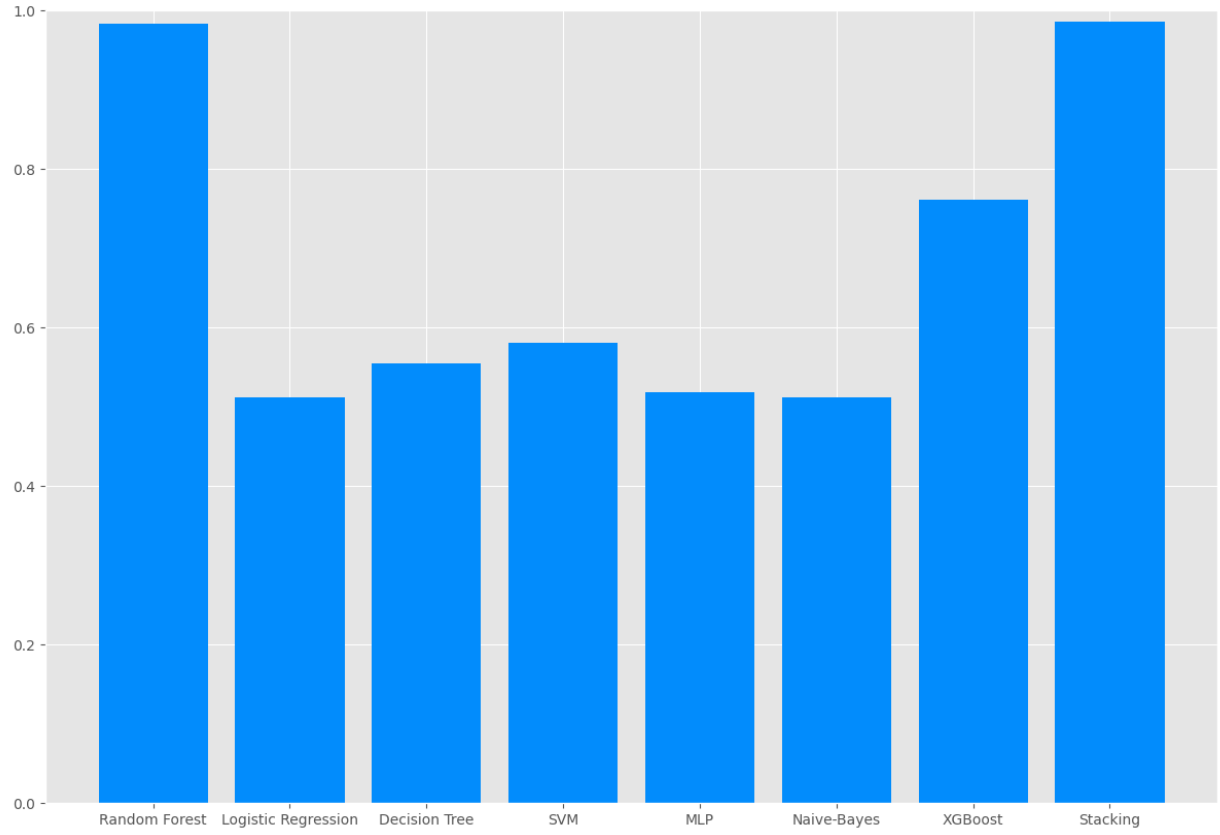


Fig 17: Accuracies 3

4.4 PROFIT-RISK ANALYSIS:

4.4.1 AFTER ESTIMATED LOSS CALCULATION AND SORTING THE DATASET IN ASCENDING ORDER OF PROBABLITY OF DEFAULT VALUES:

	DerogCnt	TLTimeFirst	TLCnt03	TLSum	TLMaxSum	TLDel60Cnt	TLBalHCPct	TLSatPct	TLDel3060Cnt24	TLOpenPct	PoD	RR	LGD	ES	Predicted TARGET
25054	0	155	0	316.0	19676.0	38	0.9391	0.5926	0	0.6400	0.000258	0.983940	5.075015	-0.001308	0
3568	2	125	0	15335.0	172238.0	0	1.1827	0.2500	3	0.6250	0.000592	0.910966	1365.332999	-0.808563	0
14825	0	77	0	3887.0	88702.0	11	0.1965	0.7273	0	0.8333	0.000608	0.956179	170.331774	-0.103632	0
27350	0	251	7	42774.0	3966.0	23	0.7908	0.5882	3	0.0800	0.000622	-9.785174	461325.031770	-287.095387	0
7287	24	38	7	12489.0	14244.0	10	0.5999	0.1071	0	0.1333	0.000637	0.123210	10950.233151	-6.979175	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
18925	6	154	7	18590.0	20576.0	2	0.7330	0.3103	2	0.7059	0.999998	0.096520	16795.689152	-16795.657117	1
18925	6	154	7	18590.0	20576.0	2	0.7330	0.3103	2	0.7059	0.999998	0.096520	16795.689152	-16795.657117	1
18020	0	403	0	7866.0	111167.0	0	1.0086	0.6579	0	0.7241	0.999998	0.929242	556.585641	-556.584646	1
18020	0	403	0	7866.0	111167.0	0	1.0086	0.6579	0	0.7241	0.999998	0.929242	556.585641	-556.584646	1
15468	0	101	0	11029.0	19597.0	0	0.3543	0.1154	3	0.5000	0.999998	0.437210	6207.013369	-6207.003750	1

12191 rows × 15 columns

4.4.2 PERCENTILE RESULTS:

Percentile	Net Revenue					
1	3.029e+06	26	2.322e+06	51	2.393e+06	76	-2.102e+07
2	2.404e+06	27	3.033e+06	52	2.112e+06	77	-2.440e+07
3	2.477e+06	28	2.555e+06	53	-1.085e+08	78	-6.211e+07
4	2.514e+06	29	2.679e+06	54	-1.014e+07	79	-5.400e+06
5	3.057e+06	30	2.261e+06	55	-7.199e+06	80	-1.149e+07
6	2.602e+06	31	2.291e+06	56	-9.399e+07	81	-1.024e+07
7	2.985e+06	32	2.747e+06	57	-9.105e+06	82	-2.802e+07
8	3.186e+06	33	2.769e+06	58	-1.500e+07	83	-4.932e+07
9	2.622e+06	34	2.342e+06	59	-1.494e+07	84	-1.059e+07
10	2.551e+06	35	2.548e+06	60	-1.276e+07	85	-1.174e+07
11	2.518e+06	36	2.370e+06	61	-2.680e+07	86	-5.711e+06
12	2.706e+06	37	2.597e+06	62	-4.520e+07	87	-6.591e+06
13	2.550e+06	38	2.601e+06	63	-2.875e+07	88	-1.820e+07
14	2.500e+06	39	2.376e+06	64	-1.448e+07	89	-6.275e+06
15	2.433e+06	40	2.157e+06	65	-1.157e+07	90	-1.519e+07
16	2.764e+06	41	2.385e+06	66	-1.107e+07	91	-2.639e+08
17	2.995e+06	42	2.380e+06	67	-7.762e+06	92	-2.295e+07
18	2.714e+06	43	2.565e+06	68	-1.727e+07	93	-1.096e+07
19	2.905e+06	44	2.598e+06	69	-1.099e+07	94	-1.509e+07
20	2.588e+06	45	2.299e+06	70	-1.050e+07	95	-1.776e+07
21	2.550e+06	46	2.566e+06	71	-3.452e+06	96	-1.679e+07
22	2.917e+06	47	2.742e+06	72	-1.488e+07	97	-4.716e+06
23	2.697e+06	48	2.447e+06	73	-2.833e+07	98	-7.874e+07
24	2.276e+06	49	2.346e+06	74	-1.453e+07	99	-2.286e+07
25	2.797e+06	50	2.513e+06	75	-1.015e+07	100	-1.939e+05

Net revenue positive observed until the 52nd percentile.

Chapter 5

Conclusion and Future work

Thus, from the above results, we can see that, up until the 52nd percentile, the net revenue is positive, indicating a profit for the financial institute, and after that, losses start coming up. Therefore, the bank or financial institute can formulate a policy that loans will be given to people up to the 52nd percentile when sorted in ascending order of probability of default.

After the dataset size is increased to counteract the small size of the existing dataset, the class imbalance issue is fixed using the bootstrapping method. SMOTE, although popular to rectify class imbalance issues, fails to increase accuracy from the default case. It is also found that the ensemble stacking algorithm provides the best accuracy of 98.55%, which is very exceptional out of all the base learners, and is thus adopted to predict the probability of default values.

In the future, better and bigger datasets will need to be tested out with the current model to further test the model's potency. Further, in this work, only estimated losses or profits on revenue are calculated. With more information, such as the principal amount, interest rate, number of times the loan is compounded, etc., available, a more accurate measure of profit-loss values can be calculated.

Appendices

APPENDIX 1: CODE SAMPLE:

```
import pandas as pd
import numpy as np

df = pd.read_excel("CreditDataset.xlsx")

"""## **Pre-processing:**"""

df.shape

df.head()

#dropping customer ID column
df = df.drop('ID',axis=1)
df.shape

"""**Data Imputations:**"""

#Finding the missing values:
df.isna().sum()

import matplotlib.pyplot as plt
import seaborn as sns

df1 = df

# Replacing missing values with median
df1['InqTimeLast'] = df1['InqTimeLast'].fillna(df1['InqTimeLast'].median())
df1['TLSum'] = df1['TLSum'].fillna(df1['TLSum'].median())
df1['TLMaxSum'] = df1['TLMaxSum'].fillna(df1['TLMaxSum'].median())
df1['TLCnt'] = df1['TLCnt'].fillna(df1['TLCnt'].median())
df1['TLSatCnt'] = df1['TLSatCnt'].fillna(df1['TLSatCnt'].median())
df1['TL75UtilCnt'] = df1['TL75UtilCnt'].fillna(df1['TL75UtilCnt'].median())
df1['TL50UtilCnt'] = df1['TL50UtilCnt'].fillna(df1['TL50UtilCnt'].median())
df1['TLBalHCPct'] = df1['TLBalHCPct'].fillna(df1['TLBalHCPct'].median())
df1['TLSatPct'] = df1['TLSatPct'].fillna(df1['TLSatPct'].median())
df1['TLOpenPct'] = df1['TLOpenPct'].fillna(df1['TLOpenPct'].median())
df1['TLOpen24Pct'] = df1['TLOpen24Pct'].fillna(df1['TLOpen24Pct'].median())

df1.isna().sum()

"""## Increasing Dataset size:"""
```

```
pip install ctgan

from ctgan import CTGAN

df1.columns

discrete_columns = list(df1.columns)

df2 = df1

ctgan = CTGAN(epochs=10)
ctgan.fit(df2, discrete_columns)

syndf = ctgan.sample(30000)

syndf

"""## **Balancing Dataset:**"""

syndf['TARGET'].value_counts()

import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(7,6))
sns.countplot(x='TARGET', data = syndf)

"""### **Data Augmentation:**

#### SMOTE:

Test-Train Split:
"""

# X1 = df1.drop(['TARGET'], axis = 1)
# Y1 = df1['TARGET']

#y1 = df1.iloc[:, 0].values
#x1 = df1.iloc[:, 1:29].values

#X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
random_state=0,stratify=y)
#sc = StandardScaler()
#X_train = sc.fit_transform(X_train)
#X_test = sc.transform(X_test)
```

```
# from sklearn.model_selection import train_test_split
# X1_train, X1_test, Y1_train, Y1_test = train_test_split(X1, Y1, test_size = 0.25, random_state
= 5)

# from imblearn.over_sampling import SMOTE
# sm = SMOTE(random_state=42)
# X1_train, Y1_train = sm.fit_resample(X1_train, Y1_train)

# X1_train.shape

# X_Full = np.hstack((X1_train,Y1_train.values.reshape(-1,1)))
# sdf1 =
pd.DataFrame(X_Full,columns=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,
25,26,27,28,'TARGET'])
# plt.figure(figsize=(7,6))
# sns.countplot('TARGET', data = sdf1)

"""#### Bootstrap Method:"""

# resample uses "bootstrapping" method to regenrate samples by randomly selecting data for
each class
from sklearn.utils import resample
df_0 = syndf[syndf['TARGET'] == 0]
df_1 = syndf[syndf['TARGET'] == 1]
df_1.shape

# Apply Resample
df_1_upsample = resample(df_1, n_samples = 23864, replace = True, random_state = 123)
df_1_upsample.shape

d = df_1_upsample.drop_duplicates()
d.shape

syndf2 = pd.concat([df_0, df_1_upsample])
syndf2['TARGET'].value_counts()

plt.figure(figsize=(7,6))
sns.countplot(x='TARGET', data = syndf2)
"""##### Test-Train Split:"""

X1 = syndf2.drop(['TARGET'], axis = 1)
Y1 = syndf2['TARGET']
from sklearn.model_selection import train_test_split
X1_train, X1_test, Y1_train, Y1_test = train_test_split(X1, Y1, test_size = 0.25, random_state =
5)
```

```
"""## **Feature Selection:**

**Using Information Gain:**
"""
from sklearn.feature_selection import mutual_info_classif
mutual_info = mutual_info_classif(X1_train, Y1_train)

mutual_info = pd.Series(mutual_info)
mutual_info.index = X1_train.columns
mutual_info.sort_values(ascending=False)

mutual_info.sort_values(ascending=False).plot.bar(figsize=(20, 8))

"""Choosing the top 10 features based on information gain:"""

from sklearn.feature_selection import SelectKBest
sel_five_cols = SelectKBest(mutual_info_classif, k=10)
sel_five_cols.fit(X1_train, Y1_train)
top = X1_train.columns[sel_five_cols.get_support()]
top

X1_train = X1_train[top]
X1_test = X1_test[top]

X1_train

"""## **Model Running and Testing:**

### Logistic Regression:
"""

from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
logreg_pipeline = make_pipeline(StandardScaler(), LogisticRegression(random_state=2))
logreg_pipeline.fit(X1_train, Y1_train)

from sklearn.metrics import accuracy_score
prediction_values = logreg_pipeline.predict(X1_test)
logreg_accuracy = accuracy_score(Y1_test, prediction_values)
print("Accuracy on Test Data: ",logreg_accuracy*100)

Y1_Pred = logreg_pipeline.predict(X1_test)
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
report = classification_report(Y1_Pred, Y1_test)
print(report)
```

```
"""### Decision Tree:"""

from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
# maximum depth of decision trees is 9
dtree_pipeline = make_pipeline(StandardScaler(), DecisionTreeClassifier(criterion = "entropy",
random_state = 0, max_depth = 9))
dtree_pipeline.fit(X1_train, Y1_train)

from sklearn.metrics import accuracy_score
prediction_values = dtree_pipeline.predict(X1_test)
dtree_accuracy = accuracy_score(Y1_test, prediction_values)
print("Accuracy on Test Data: ",dtree_accuracy*100)

Y1_Pred = dtree_pipeline.predict(X1_test)
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
report = classification_report(Y1_Pred, Y1_test)
print(report)

"""### Random Forest:"""

from sklearn.ensemble import RandomForestClassifier
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
rf_pipeline = make_pipeline(StandardScaler(), RandomForestClassifier(random_state = 18))
rf_pipeline.fit(X1_train, Y1_train)

from sklearn.metrics import accuracy_score
prediction_values = rf_pipeline.predict(X1_test)
rf_accuracy = accuracy_score(Y1_test, prediction_values)
print("Accuracy on Test Data: ",rf_accuracy*100)

Y1_Pred = rf_pipeline.predict(X1_test)
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
report = classification_report(Y1_Pred, Y1_test)
print(report)

"""### Support Vector Machine (SVM) Classifier:"""

from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
```

```
"""Radial Basis Function (RBF):"""

svm_pipeline3 = make_pipeline(StandardScaler(), SVC(kernel='rbf', probability=True))
svm_pipeline3.fit(X1_train, Y1_train)

from sklearn.metrics import accuracy_score
prediction_values = svm_pipeline3.predict(X1_test)
svm3_accuracy = accuracy_score(Y1_test, prediction_values)
print("Accuracy on Test Data: ",svm3_accuracy*100,"\n")
Y1_Pred = svm_pipeline3.predict(X1_test)
from sklearn.metrics import accuracy_score
report = classification_report(Y1_Pred, Y1_test)
print(report)

"""### Perceptron:"""

from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
mlp_pipeline = make_pipeline(StandardScaler(), MLPClassifier(hidden_layer_sizes=(10, 10,
10), activation='relu', solver='sgd', max_iter=1000))
mlp_pipeline.fit(X1_train, Y1_train)

from sklearn.metrics import accuracy_score
prediction_values = mlp_pipeline.predict(X1_test)
mlp_accuracy = accuracy_score(Y1_test, prediction_values)
print("Accuracy on Test Data: ",mlp_accuracy*100)

Y1_Pred = mlp_pipeline.predict(X1_test)
from sklearn.metrics import accuracy_score
report = classification_report(Y1_Pred, Y1_test)
print(report)

"""### Naive-Bayes:"""

from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.naive_bayes import GaussianNB
nb_pipeline = make_pipeline(StandardScaler(), GaussianNB())
nb_pipeline.fit(X1_train, Y1_train)

from sklearn.metrics import accuracy_score
prediction_values = nb_pipeline.predict(X1_test)
nb_accuracy = accuracy_score(Y1_test, prediction_values)
print("Accuracy on Test Data: ",nb_accuracy*100)
```

```
Y1_Pred = nb_pipeline.predict(X1_test)
from sklearn.metrics import accuracy_score
report = classification_report(Y1_Pred, Y1_test)
print(report)

"""### XGBoost:"""

from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
import xgboost
from xgboost import XGBClassifier
xgb_pipeline = make_pipeline(StandardScaler(), XGBClassifier(random_state = 18))
xgb_pipeline.fit(X1_train, Y1_train)

from sklearn.metrics import accuracy_score
prediction_values = xgb_pipeline.predict(X1_test)
xgb_accuracy = accuracy_score(Y1_test, prediction_values)
print("Accuracy on Test Data: ",xgb_accuracy*100)

Y1_Pred = xgb_pipeline.predict(X1_test)
from sklearn.metrics import accuracy_score
report = classification_report(Y1_Pred, Y1_test)
print(report)

"""### Ensemble - Stacking Classifier:"""

from sklearn.ensemble import StackingClassifier

def get_stacking():
   #base models:
   l0 = list()
   l0.append(('Logistic Regression', LogisticRegression(random_state=2)))
   l0.append(('Decision Tree', DecisionTreeClassifier(criterion = "entropy", random_state = 0,
max_depth = 9)))
   l0.append(('Random Forest',RandomForestClassifier(random_state=18)))
   l0.append(('MLP',MLPClassifier(hidden_layer_sizes=(10, 10, 10), activation='relu',
solver='sgd', max_iter=1000)))
   l0.append(('Naive-Bayes',GaussianNB()))
   l0.append(('SVM', SVC(kernel='rbf', probability=True)))
   l0.append(('XGBoost', XGBClassifier(random_state = 18)))
   #meta learner model:
   l1 = XGBClassifier()
   #the stacking ensemble:
   model = StackingClassifier(estimators=l0, final_estimator=l1)
   return model
```

```
smodel=get_stacking() #construct a stacking model
st_pipeline = make_pipeline(StandardScaler(), smodel)
st_pipeline.fit(X1_train, Y1_train)

prediction_values = st_pipeline.predict(X1_test)
st_accuracy = accuracy_score(Y1_test, prediction_values)
print("Accuracy on Test Data: ",st_accuracy*100)

Y1_Pred = st_pipeline.predict(X1_test)
from sklearn.metrics import accuracy_score
report = classification_report(Y1_Pred, Y1_test)
print(report)

"""## Results:"""

import matplotlib.pyplot as plt
import seaborn as sns
names = ['Random Forest','Logistic Regression','Decision Tree','SVM','MLP','Naive-
Bayes','XGBoost','Stacking']
values =
[rf_accuracy,logreg_accuracy,dtree_accuracy,svm3_accuracy,mlp_accuracy,nb_accuracy,xgb_ac
curacy,st_accuracy]
f = plt.figure(figsize=(50,10),num=10)
plt.subplot(131)
plt.ylim(0,1)
plt.bar(names,values,color='#038cfc')
j=0
print("ACCURACIES:")
for i in names:
  print('{:20s} --> {}'.format(i,values[j]))
  j+=1

"""## Finding Probablity of Default values:"""

predictions = st_pipeline.predict_proba(X1_test)

df_prediction_prob = pd.DataFrame(predictions, columns = ['Prob_Good(0)', 'Prob_Bad(1)'])
#df_prediction_target = pd.DataFrame(st_pipeline.predict(X1_test), columns =
['predicted_TARGET'])
#df_test_dataset = pd.DataFrame(Y1_test,columns= ['Actual Outcome'])
#DF=pd.concat([df_test_dataset, df_prediction_prob, df_prediction_target], axis=1)
#DF.head()
df_prediction_prob
```

REFERENCES

[1] Kennedy, K. (2013). Credit scoring using machine learning. Doctoral thesis. Technological University Dublin.

[2] Provenzano, A. R., Trifirò, D., Datteo, A., Giada, L., Jean, N., Riciputi, A., ... & Nordio, C. (2020). Machine learning approach for credit scoring. arXiv preprint arXiv:2008.01687.

[3] Ampountolas, A., Nyarko Nde, T., Date, P., & Constantinescu, C. (2021). A machine learning approach for micro-credit scoring. Risks, 9(3), 50.

[4] Nyon, E. E., & Matshisela, N. (2018). Credit scoring using machine learning algorithims. Zimbabwe Journal of Science and Technology, 13(1), 26-34.

[6] Abdou, H. A., & Pointon, J. (2009). Credit scoring and decision making in Egyptian public sector banks. International Journal of Managerial Finance.

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

[7] Luo, C., Wu, D., & Wu, D. (2017). A deep learning approach for credit scoring using credit default swaps. Engineering Applications of Artificial Intelligence, 65, 465-470.

[8] Pol, S., & Ambekar, S. S. (2022). Predicting Credit Ratings using Deep Learning Models–An Analysis of the Indian IT Industry. Australasian Accounting, Business and Finance Journal, 16(5), 38-51.

[9] Addo, P. M., Guegan, D., & Hassani, B. (2018). Credit risk analysis using machine and deep learning models. Risks, 6(2), 38.

[10] Wei, S., Yang, D., Zhang, W., & Zhang, S. (2019). A novel noise-adapted two-layer ensemble model for credit scoring based on backflow learning. IEEE Access, 7, 99217-99230.

[11] Zhang, W., Yang, D., Zhang, S., Ablanedo-Rosas, J. H., Wu, X., & Lou, Y. (2021). A novel multi-stage ensemble model with enhanced outlier adaptation for credit scoring. Expert Systems with Applications, 165, 113872.

[12] Li, Y., & Chen, W. (2020). A comparative performance assessment of ensemble learning for credit scoring. Mathematics, 8(10), 1756.

[13] Peng, B., Zhang, A., & Zhang, T. (2021). Credit Scoring Model in Imbalanced Data Based on CNN-ATCN.

[14] Zhang, T., & Chi, G. (2021). A heterogeneous ensemble credit scoring model based on adaptive classifier selection: An application on imbalanced data. International Journal of Finance & Economics, 26(3), 4372-4385.

[15] He, H., Zhang, W., & Zhang, S. (2018). A novel ensemble method for credit scoring: Adaption of different imbalance ratios. Expert Systems with Applications, 98, 105-117.