2023

Tree-Based Approaches for Predicting Financial Performance

Ahmed Shafeek Abouhassan
University of Windsor

TREE-BASED APPROACHES FOR PREDICTING FINANCIAL PERFORMANCE

By

Ahmed Shafeek (Abouhassan)

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada

2022

TREE-BASED APPROACHES FOR PREDICTING FINANCIAL PERFORMANCE

By

Ahmed Shafeek (Abouhassan)

APPROVED BY:

_____

H. ElMaraghy
Department of Mechanical, Automotive & Materials Engineering

_____

Z. Kobti
School of Computer Science

_____

A. Ngom, Advisor
School of Computer Science

August 17, 2022

Author’s Declaration of Originality

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

The lending industry commonly relied on assessing borrowers' repayment performance to make lending decisions. This is to safeguard their assets and maintain their profitability. With the rise of Artificial Intelligence, lenders resorted to Machine Learning (ML) algorithms to solve this problem.

In this study, the novelty introduced is applying ML's Tree-based methods to a large dataset and accurately predicting financial repayment performance without using any repayment history, which was utilized in all literature reviewed. Instead, the attributes used were demographics and psychographics of applicants, only. The study's proprietary US-based dataset comprises an anonymous population whose owner does not wish to be disclosed and it contains the information of about half a million beneficiaries with a very balanced bimodal binary target distribution.

An Area Under the Curve of Receiver Characteristic Operator (ROC-AUC) of 85% was achieved with a binary classification target using CatBoost API. The study also experimented with a given tri-class target. Furthermore, this research used ML to gain insight into which attributes contribute the most to the repayment prediction. The study also tested whether similar results can be achieved with fewer attributes for the sake of the practicality of application by the data owner. The best model was applied to one of the biggest publicly available financial datasets for verification. The original research of said dataset had an accuracy score of 82%, this study achieved 79% using 5-fold Cross-Validation (CV). This result was achieved with Tree-Based models with a complexity of O(log n) compared to $O(2^n)$ in the original research, which is a significant efficiency enhancement.

Dedication

To my amazing wife and children for their help, support, and encouragement.

Acknowledgment

I, the author of this thesis, would like to express my deepest gratitude to my supervisor and ML mentor, Dr. Alioune Ngom for his ample help, support, and precious guidance. Special thanks also go to Dr. Ziad Kobti for putting me on the right track and enabling me to fulfill my life's dream of obtaining a master's degree in Artificial Intelligence. Furthermore, I am honored to have a world icon in industrial engineering and a true trailblazer, Dr. Hoda ElMaraghy accept being my external reader on the thesis committee.

I would like to express my love and gratitude to my bigger family: my mother, late father, sister, and brother, and to my wife and children for their endless patience and support throughout the 3 years it took me to finish my second bachelor's degree with great distinction and enabled me to embark on this master thesis journey with a good footing and high readiness.

I would also like to give my special thanks and great appreciation to my master colleague and Python expert Surajsinh Parkashchandra Parmar for his sleepless nights with me, coding and debugging the models we built together during that research. I wish him the best of luck in finishing his master's thesis soon.

Finally, I acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC).

Finalement, Je remercie le Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG) de son soutien.

Natural Sciences and Engineering Research Council of Canada

Conseil de recherches en sciences naturelles et en génie du Canada

Canada

Table of Contents

Chapter 1

Introduction

The economy in the US is a capitalistic debt-based, sometimes called debt-driven, economy. The oversimplified idea is that financial institutions lend their financial assets to borrowers who use these borrowed monies to purchase goods and services. These monies, used for purchases, go up the supply chain of the providers, change hands, and keep circulating through the economy of the country. However, borrowers usually promised to pay back their loans to the financial institutions with some interest. The financial institutions in turn circulate the paid back capital into future loans to other borrowers and they reap their profits from the interest paid. If hypothetically

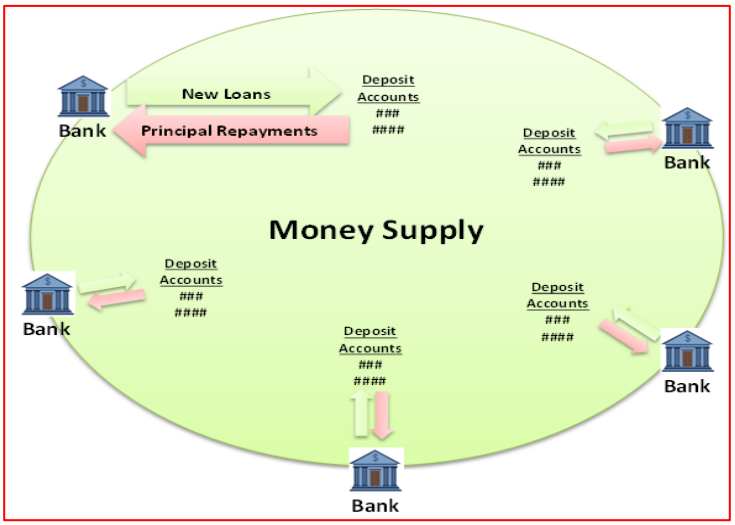


Figure 1: Money Supply Diagram
Note. Adapted from Workable Economics

speaking, all borrowers were to pay or fail to pay back for that matter, all lent assets to financial

institutions, and no one were to borrow anymore, the financial institutions would not make any profits to sustain their business. As a result, services and goods purchases will slow down to a grinding halt, the circulating assets will dry up, and the economy of the country would collapse.

Accordingly, having a streamlined loan and loan repayment cycle is an essential pillar in the structure of the US economy and for the profitability of financial institutions. For this to work, lending institutions need to be selective of their borrowers to avoid those at risk of default to keep that revolving money/profit cycle going. In the digital age, this is achieved by applying state-of-the-art ML algorithms to the enormous amounts of credit history available about almost everyone, including potential borrowers, to create models that can predict the risk of default[The World Bank]. This practice results in assigning scores for creditworthiness, which is higher for lesser-risk individuals and lower for high-risk ones. All this is maintained in an individualized and controlled financial record to which all lending institutions have regulated access. This enables lending institutions to pick and choose who to lend their monies to according to their risk appetite and tolerance. This approach is also endorsed, supported, and regulated by the US laws such as the *Fair Credit Reporting Act.* The Act (Title VI of the Consumer Credit Protection Act) protects

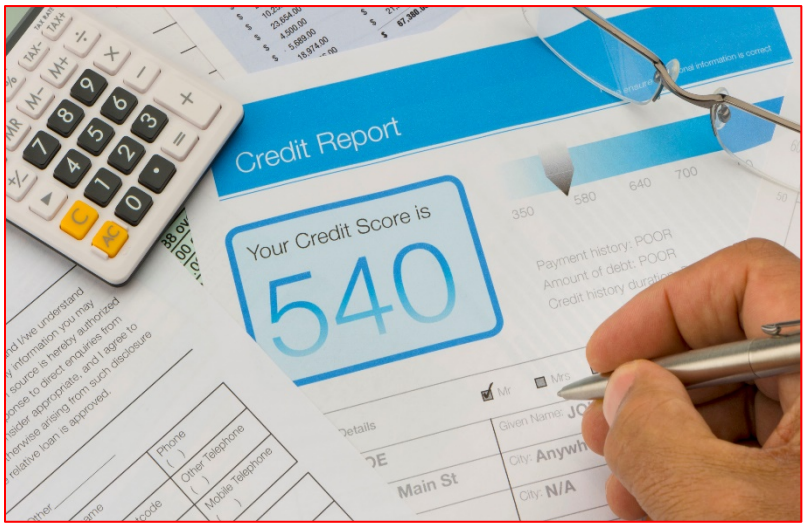


Figure 2: Credit Scores Indicate Individual's Risk of Default
Note. Adapted from: The Balance Careers

information collected by consumer reporting agencies such as credit bureaus, medical information companies, and tenant screening services. Information in a consumer report cannot be provided to anyone who does not have a purpose specified in the Act. People living in the US must have the healthy credit worthiness to be integrated into and enrich the US economy.

1.1 Motivation

The proprietary owner of the dataset used in this research is neither a licensed financial institution nor seeking profit from their loan program. Instead, they target a sector of the US population that is generally struggling with financial literacy, is underbanked, and has low creditworthiness. The program aims to help this sector be more successful in its integration into the US economy. This is achieved through issuing this population unsecured and interest-free loans that are also penalty-free and term-flexible, for the sole humanitarian purpose of helping this population stand on their feet, be economically more successful, and, as a result, more integrated into the US debt-based economy. Indirectly this endeavor helps this population build their credit scores enabling them to take loans from the financial sector to purchase their first expensive vehicle or first home property. However, due to the underprivileged financial status of this population in the US, the lender was interested in understanding this population more and those of them who succeed through this humanitarian approach and those who do not. This is because, the more that is known about the population, the more support and tailored solutions can be devised and provided to them to lift them out of the perils of their financial disadvantage. The loans that they pay off also help fund new loans for other people struggling financially in the future.

The present study will use the available demographic and psychographic data collected by the dataset owner to achieve two main goals:

1. predict the future repayment performance of the future borrowers of the program so that the dataset owner can better plan the humanitarian fund for future loans, and
2. understand what the factors, contributing to the desirable performance of the borrowers, are and what the ones leading some borrowers to underperform, are.

1.2 Contribution

This dataset of this research has roughly 680,000 records, which is several multiples of the biggest publicly available credit scoring datasets. Based on this, the data facilitate three processes:

1. using a large proprietary dataset with purely non-financial data to train a predictive model to classify financial performance with acceptable metrics using ML, for the first time,
2. applying tree-based methods and deep learning in a multitude of algorithms to find the best performing model with this dataset type, and
3. applying the best models found in this research on one of the few publicly available financial performance datasets and attaining a similar metric performance with more efficient implementation compared to the previously used methods.

1.3 Problem Statement

This research aims to predict future borrowers' ability and consistency in repaying their loans with acceptable ROC-AUC. The dataset owner would benefit from this predictive model in forecasting and the maintenance of their humanitarian loan funds. Furthermore, the purpose of the study is to learn, understand, and decipher the contributing factors to the population's desirable

and undesirable repayment performance. This enables the dataset owner to provide tailored solutions and support to future borrowers. Finally, this research takes advantage of the unprecedented opportunity to apply its models on a much larger and cleaner dataset than any other that is publicly available. Utilizing this opportunity, the research will use non-financial data available from past borrowers to predict future borrowers' repayment performance.

The research explores two hypotheses. The first hypothesis asserts that applying the state-of-the-art Tree-Based Methods to a larger dataset would significantly reduce the training time complexity from $O(2^n)$ with the best Neural Networks applied on similar problems to $O(\log n)$ with balanced trees this study will use. The second hypothesis asserts that, while attaining this efficiency, the main performance measure, ROC-AUC, would still be at or above the 85% mark.

The thesis is divided into several sections. Chapter 2 offers a literature review, while Chapter 3 covers the ML algorithms used in the present research. Chapter 4 takes a closer look at the methodology used and the dataset, while Chapter 5 discusses the experiments, which will lead to the results and discussion in Chapter 6. Finally, Chapter 7 discusses the conclusions and future work. References and a bibliography of resources for readers who would like to know more about credit scoring and how different ML algorithms are utilized in it.

Chapter 2

Literature Review

This chapter outlines the studies that influenced and guided the present research. Though they are only a small sample, the papers that were consulted throughout this research, sum up the approach this study used when it was decided to focus our experimentation on tree-based methods or an ensemble that utilizes them. They validate the present study's research approach.

2.1 Neural Nets for Tabular Data

In an experiment that compared Neural nets to other ML methods, Shwartz-Ziv and Armon [1] lamented that some studies recently claimed that deep neural networks could outperform traditional tree-based models, such as in TabNet [2], NODE [3], DNF-Net [4], and 1D-CNN [5] when working with datasets with tabular data. Each study used a specific dataset because, unlike vision problems' ImageNet, there is no standard benchmark for tabular datasets. All research claimed that their proposed Deep Neural Networks outperformed tree-based methods on their specific datasets.

Shwartz-Ziv and Armon [1] also compared the multiple deep learning methods proposed, against multiple tree-based methods using CatBoost [6] and XGBoost [7] on a multitude of tabular

real-life datasets. They ensured that they tested every approach on every dataset and compared the same variables to secure objective comparison results. They found that XGBoost not only outperformed deep learning by more than an order of magnitude but also it required significantly less tuning. Moreover, Shwartz-Ziv and Armon [1] discovered that structuring XGBoost and deep learning together in an ensemble outperformed XGBoost when it was used solely.

2.2 Credit Scoring Using CSVM

To use the Support Vector Machine (SVM) for credit scoring, Harris [8] compared different SVM techniques on German and Barbados datasets for credit scoring from UCI. The biggest of these datasets was in the range of 10,000 instances. He proposed a novel Clustered Support Vector Machine (CSVM) approach and compared its variations. The results suggest that, traditionally, as credit scoring datasets become larger, the use of traditional non-linear SVMs would continue to yield accurate results but would become expensive computationally. Using financial attributes from these datasets, the best scores were achieved by Harris when he used the Linear CSVMs, which he compared to traditional non-linear SVMs. While Harris's methods yielded similar or better results than those found in his literature review, it was far more efficient computationally and had a complexity much smaller than the traditional $O(2^n)$ best achieved by the Kernel-Based SVMs. These were the state of the art when this research was authored in 2014. The final contribution outlined in [8] was the introduction of the CVSM as a computationally cheaper alternative to traditional credit scoring algorithms that were available at the time. Harris used the Area under the curve (AUC) as the chosen performance measure and could achieve more favorable results compared to the traditional methods of the time they experimented with.

2.3 Behaviour-Based Prediction

This is an interesting application of deep learning in credit scoring. Wang et al. [9] used a private peer-to-peer (P2P) lender company's data, which had information about users' online behaviour such as logins, repetitive clicks, and manner while the label was whether the user defaulted or not. This kind of sequential data suits well for Deep Learning. Especially, long short-term memory models (LSTMs) and attention mechanisms. Wang et al. used Event2Vec library to encode the events to have an event vector. With AUC as the performance measure and comparator, the best results were achieved with attention mechanism-based LSTM, and though XGBoost was also used on vectors, it failed to achieve a similar performance similar to AM-LSTM [9].

2.4 Comparative Assessment of Ensembles

Tree-based models, where multiple trees are built, are also called ensemble models. Li and Chen [10] compared different ensemble models using Lending Club's loan dataset for Q4 of 2018, which had around 36000 samples and 150 variables having 22.03% loan defaults and remaining fully paid. They experimented with traditional learners and ensemble methods. The traditional learners included Neural nets, Logistic Regression, Decision Tree, Naïve Bayes, and SVM; the ensemble methods included AdaBoost, Random Forest, XGBoost, LightGBM [11], and Stacking of learners. The ensemble methods were utilized as an ensemble of trees underneath. The best performance was achieved by Random Forest (RF) on almost all metrics of accuracy, especially AUC. Furthermore, RF was the fastest model to train. Thus, Lin and Chen suggest using a bigger dataset in the future and considering the methods to balance the dataset as a computational cost [10].

2.5 Ensemble of GBDTs in P2P lending

Li et al. [12] explored a dataset of a P2P lending company in China that had 1138 attributes and 15,000 samples with highly imbalanced labels in both training and test splits. To tackle the curse of dimensionality, they converted categorical columns to numerical and performed various dimensionality reduction methods, such as principal component analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE). Li et al. argue that [12] PCA would not be suitable here because of missing values; therefore, they experimented with feature selection. The primary idea of their research was to see whether ensembles perform better than underlying individual models. Thus, they experimented with gradient boosting decision trees (GBDT) and XGBoost as individual models. They trained a linear model on top of these models and took the weighted average as the final prediction. Their primary metrics were ROC-AUC scores and F1 scores. As they had hypothesized, the ensemble of GBDT and XGBoost outperformed their underlying models and beat Neural Networks (NN), Linear Regression (LR), SVM, and K-Nearest Neighbor (KNN) by a significant margin.

2.6 ML on Imbalanced Credit Data

Addo et al. [13] utilized ElasticNet, RF, Gradient boosting, and deep learning to create a binary classifier model. This model was used to predict the probability of loan default for borrowing consumers of a real-life financial institution. They posit that this predictive model would help the financial institution update its existing predictive model by utilizing the new ML revolution. The ML revolution was the result of two phenomena: (1) the exponential growth of computing powers and (2) a large amount of digitalized data that institution has about their

borrowers and their repayment behaviours. The dataset given to Addo et al. [13] was significantly imbalanced as 98.5% of the consumers were good regular payers while consumers at default were just slightly over 1.5%. Due to that imbalance, and to ensure that their results were not biased, Addo et al. [13] resorted to synthetic data generation using the Synthetic Minority Oversampling Technique or SMOTE Algorithm. SMOTE is one of the most used oversampling methods to solve the imbalance problem, and it aims to balance class distribution by randomly increasing minority class examples by replicating them. SMOTE manufactures new minority instances between existing minority instances [14].

The primary metrics, used by Addo et al. [13] were primarily AUC and RMSE, although other metrics—such as Gini, recall, precision, F-Score, and Akaike information criterion (AIC)—were frequently used through their research. For AUC, they resorted to building the Receiver Operation Characteristics (ROC) for each consumer. The ROC is often associated with the statistical point of view of error computation. As for the Root Mean Squared Error (RMSE), it is usually preferred to go to metric when dealing with labels with numerical values. The findings outlined by Addo et al. [13] suggest that tree-based models outperformed logistic regression models, significantly, and outperformed all Deep Learning Models with superiority in both RSME and AUC. They also conclude that it is best to do a feature selection for the top 10 relevant features and use the tree-based model, which could reliably predict with the same accuracy using only those top 10 features. Their logic is that minimizing the number of features enables the loan officer to come to a quick conclusion on whether they should accept the consumer's loan application based on taking minimal information from consumers rather than asking them to fill in 181 pieces of information which takes a long time [13].

2.7 Data Mining on Credit Risk Dataset

After promoting a novel Sorted Smoothing Method (SSM), Yeh and Lien [15] trained various data mining models on the binary targets; however, for evaluation, they used the SSM method to sort the samples based on predicted probability by the model and then raked a sliding window average of binary targets to get the real *probability* for a sample. They then used R^2 to compare the models. The dataset was one of the largest publicly available sets and can be found on the University of California in Irvine's dataset repository and is called the Taiwanese Credit Card Default dataset. It has 30,000 records with 18 financial and demographic attributes about borrowers. The financial information includes their recent credit payments and credit limit. As for the demographics, they include information such as gender, marital status, age, and education level. In Yeh and Lien's experiments, they examined several models—NN, K-nearest neighbor, Logistic Regression, Discriminant Analysis, Naïve Bayes, and Neural Networks—and NN proved to be the best. However, the research was carried out in 2007, when tree-based methods had become widespread.

2.8 Feature Selection and Trees

Trivedi [16] engaged the loan approval problem of financial institutions and proposed research for the best ML algorithm to accurately predict consumers' credit scores. The model should receive the consumer demographic and financial history information and predicts their creditworthiness. Trivedi used the publicly available German Credit Dataset and feature selection techniques such as Information-Gain, Gain-Ratio, and Chi-Square. The study compared multiple ML models—such as Bayesian, Support Vector Machine, Decision Tree, and Random Forest—

and several performance metrics were performed, such as accuracy, F-Score, Recall, and Precision. Trivedi concluded that, after a myriad of meticulous experiments, Chi-Square was the most optimum feature selection method. Alternately, for the ML model, the results suggest that Random Forest is the most optimal algorithm as it could generalize well, achieved 93% accuracy, and enhanced both recall and precision. Trivedi notes, however, that the software and hardware setup for Random Forest required significant time and that the Decision Tree was very comparable to Random Forest and posed a close second-best model. Trivedi did not recommend any performance indicator over the rest and lamented the common problem of credit performance research, which is the scarcity of practical and meaningful datasets.

2.9 Comparison of Tree-Based Models

Ampountolas et al. [17] also used Random Forest for credit scoring and applied it as the deciding factor for financial institutions whether institutions approve a consumer loan application. They demonstrate that off-the-shelf multiclass classifiers, such as Random Forest, can perform this credit score prediction task effectively by using readily available demographical data about consumers. Ampountolas et al. also demonstrate that their approach presents an inexpensive and reliable means for micro-lending financial institutions. They focused their humanitarian approach on the developing world in which potential borrowers generally lack credit history information and sometimes have never even had a bank account before. This is analogous to a lot of borrowers in the current research. Ampountolas et al. experimented with Decision Trees, Extra Tree, Random Forest, XGBoost, AdaBoost, Neural Network, and Multilayer Perceptrons, and they conclude that the best performance was achieved with AdaBoost, XGBoost, and Random Forest Classifiers. In their finding, they stress that the best approach with their tabular data was an ensemble utilizing

Tree-Based algorithms. Finally, they also found that the most important features across all algorithms were age, log of amount (they manufactured that feature to reduce the variability of the loan amounts), and annual interest rate.

2.10 Model Interpretation

Lundberg and Lee [18] looked at model interpretation: Based on the complexity of the data, the model becomes complex too. Often leading to sub-features that are not understandable by humans. Deep Learning models are called “Black-box” models because of their complex architecture and low interpretability. However, tree-based models can be interpreted by looking at their decision nodes; thus, a unified method for model interpretation becomes important. Shapley values have been used in game theory to find importance, so Lundberg and Lee proposed *Shapley Additive exPlanations* (SHAP) as a unified framework for an interpreting model. They tried to approximate the contribution of each feature by removing the feature from inputs and observing how that affects the result.

2.11 The Demographics of Delinquency

Emmons and Ricketts’ [19] conducted a traditional statistical analysis study, and though it is not an ML paper, its insights can provide insights into Survey of Consumer Finances data from 1995 to 2003 regarding inspecting the correlation between loan delinquency and demographic attributes of the population. They posit that demographic characteristics—such as age, education, race, or ethnicity—correlate with the financial and economics of the population and that this in turn affects their loan repayment behaviour and performance. Based on data analysis, Emmons

and Ricketts argue that there is low empirical support for the Demographics Don’t Matter framework of understanding economics and that age, education level, and race affect repayment.

2.12 Ordinal Classification

Having a multi-class problem where labels have an ordinal relationship, is a special case of the multi-class classification domain. Thus, Frank and Hall [20] proposed a framework for ordinal classification with the primary idea that we can train n-1 trees to solve an n-class problem. For example, n-1 probabilities for a single record can be classified as follows:

- prob(1) = prob(M_1)
- prob(2) = prob(M_2) - prob(M_1)
- prob(3) = prob(M_3) - prob(M_2)

Here, prob(1) means the probability of a sample being from class 1, and prob(M_1) means the probability score from binary model 1. The final class of the whole model will be the maximum probability of any previously calculated probability. The core idea behind this approach is that the classes here are ordinal. Therefore, subsequent class probability can be calculated by subtracting the probability of the previous class.

2.13 Summary of literature review

The reviewed literature outlines various ML methods and how effective they are on datasets analogous to the one in the present study. It was clear that deep learning, which took the field by storm achieving advanced performance on so many problems, falls short on tabular data. The transfer learning technique is also not applicable in the present study as the dataset is unique and unavailable publicly. Thus, selecting metrics to measure the model’s performance is a major step.

For example, the reviewed literature has shown that using the accuracy metric on an imbalanced dataset does not tell the actual story of the model's performance. The most appropriate metric should be chosen depending on the problem, data, and goals. Based on the review at hand, the use of ROC-AUC was determined to be the ideal metric for the present study. Furthermore, it was clear that a model is only as good as its data. Data pre-processing, cleaning, and feature selection can significantly improve the model's performance. Random Forest (RF) seems to be a robust structure amongst the techniques used in research that examined similar problems. Thus, RF and newer tree-based methods will be experimented with in the present study for credit scoring and repayment performance prediction. Lastly, the literature reviewed inspired the best approaches that impact the results of the model used in the present study. It also underscored the importance of understanding interpretability and determining the feature importance of the model used in the present study. As outlined in Table 1, there is a research gap in the reviewed literature that apply an ML algorithm on non-financial data in a large dataset to predict loan repayment.

Paper	Demographics Psychographics Only	Machine Learning	Tree Based Methods	Financial Repayment Performance	Public Dataset
2.1		X	X		
2.2		X		X	X
2.3		X		X	X
2.4		X	X	X	X
2.5		X	X	X	
2.6		X	X	X	
2.7		X		X	X
2.8		X	X	X	X
2.9		X		X	
2.10		X	X		
2.11	X			X	
2.12		X	X		
Current Study	X	X	X	X	

Table 1: Research Gap Analysis

Chapter 3

Utilized Machine Learning

Machine learning is a subclass of broader artificial intelligence (AI). Simply put, it is the subset of AI models that learns from experience (***E***) how to better perform a task (***T***) evidenced by performance metric (***P***) while enhancing over time and more ***E***. ML models and algorithms aim to imitate the way the human brain neurons learn. After humans carefully extract the important features of a given dataset, the data is passed to the ML neurons, and they then try to build a relationship between the inputs and the targets/labels of a set. There are two modes of ML approaches: supervised and unsupervised learning. Figure 3 illustrates the simplest ML neuron, the Perceptron, and how it compares to the human neuron.

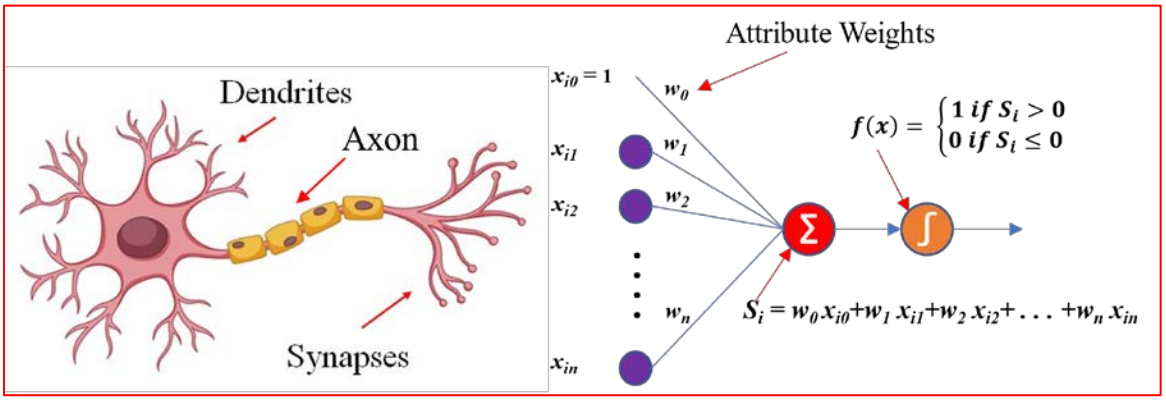


Figure 3: Comparison Between a Human Neuron and The Perceptron

The human neuron works in gigantic networks inside our brains. Dendrites from one neuron are interconnected with layers of other neurons' Synapses and that neuron's Synapses are,

in turn, interconnected with other neurons' Dendrites and so on. Electric signals in the nervous system move from one neuron to another such that the output of each neuron is the input of another. Dendrites modulate the received signals amplifying or attenuating them before passing them to the subsequent Synapses. The neuron's Axon (the orange cylinders in Figure 3) creates a certain resistance for the neuron's modulated inputs that causes the neuron not to fire up output to the subsequent neurons, except if the strength of collective modulated signals can overcome the resistance threshold. Therefore, it is said that intelligence in humans is not achieved by intelligent neurons but instead is in the modulated connections that give varying importance to different input signals to facilitate obtaining accurate results. Intelligence in human brains is in the map connecting one neuron to another and in modulations.

As seen in Figure 3, the perceptron is the simplest ML neuron unit, and it is closely modeled after the human neuron. For example, the electric signals that Dendrites receive are represented by numerical values that are the outputs of previous perceptrons coming as inputs to our perceptron (x_{i1} - x_{id}). The signal modulation is represented by the varying weights that the perceptron assigns to each input (w_0 - w_d). Like the human neuron, the perceptron will sum all the weighted numerical inputs that it receives and will only fire up a numerical output (1) to subsequent perceptrons if the collective weighted input ($\sum$) becomes above a certain threshold of function ($f(x)$). Otherwise, the perceptron will default to the default signal (0). Finally, like the human neuron, the perceptron is a simple addition unit with no intelligence inside it. However, its intelligence is found in the right weights that it can assign to the various inputs that it receives and how it is connected to other layers of perceptrons.

Unsupervised learning occurs data is fed into the training of the model without the target, sometimes called label, which is needed to eventually make predictions. The model is allowed to

group the data based on their input features, which is called **clustering**. Alternatively, the model is then used to analyze the input features and either eliminate redundant features or extract the most relevant features, statistically speaking, and this is called **Dimensionality Reduction**.

In broad terms, a supervised learning problem involves splitting a given dataset into a training set and a testing set. The model works on learning, as much as attainable, from the training set and is consequently tested by the test set. The learning could also be called training or fitting, stops once the model approaches the optimum result achieved without overfitting the training set, which is memorizing the data points rather than learning from them. Supervised learning is the best approach used for **Regression** and **Classification**.

In the current research, several ML approaches were used for experimentation and achieved the best **classification** results attainable. To understand the results, it is important to take a deeper look at the approaches that were used in this research with relevant illustrations.

3.1 Classification

Classification uses an ML model to assign a data point to one of two or more classes. For example, to prevent bots from spamming websites, access requires users to select cats or dogs from a collection of photos. Objectives such as False, Pass, and Fail can be multiclass when there are more than two classes, such as customer rating on a scale from a set Likert scale of discrete values. This could be in forms such as {1, 2, 3, 4, 5}, {A, B, C, D, E}, or {Red, Blue, Yellow, Green, Black}. Figure 4 illustrates a simple two-dimensional binary classification model where the separating curve is simply a straight line that separates between the red circles and the green plusses.

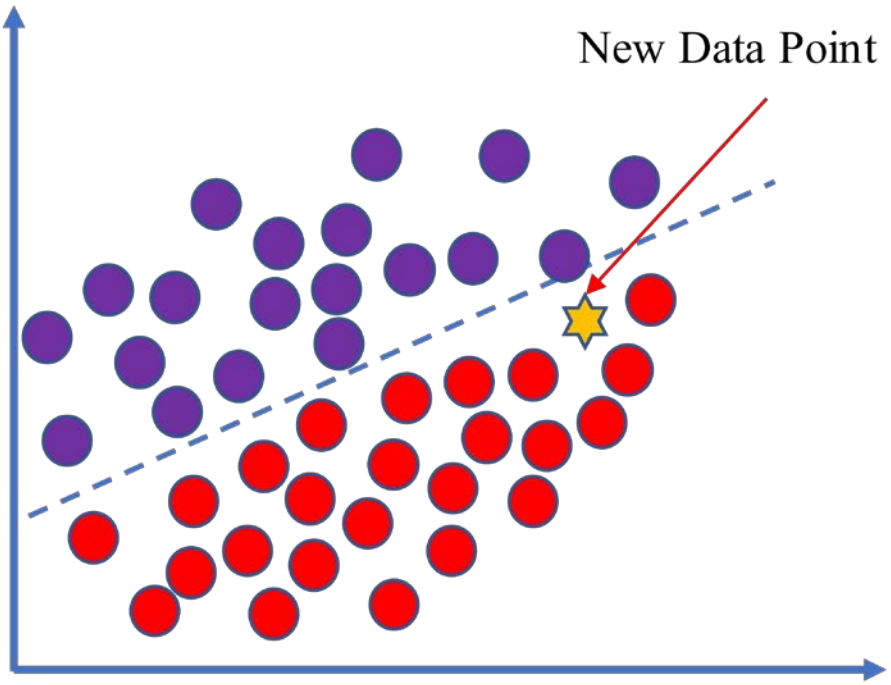


Figure 4: Binary Classification Illustrated

Just like other ML models, the number of features and data points plays a huge role in the accuracy of the classification model's prediction. When features, also called attributes, are extensive and the number of data points is low, accuracy will be affected as it will be unlikely that the model would learn and generalize adequately with the few samples and the confusing large number of features. In this case, usually, Dimensionality Reduction techniques would help enhance the accuracy. However, if the number of data points is vast but the number of features is low, the model will tend to overfit quickly and may not be able to provide a good prediction. Finding a

Definition (Classification Problem) For $n, c \in \mathbb{N}$, a set of c labels I, and a sequence $x^{(i)} \in \mathbb{R}^n, y^{(i)} \in I$, $1 \leq i \leq M$, one calls the problem of finding a function $f : \mathbb{R}^n \to I$ such that $f(x^{(i)}) = y^{(i)}$ for all $1 \leq i \leq N$ an n-dimensional *classification problem* with c classes.

- The set $(x^{(i)}, y^{(i)})_{1 \leq i \leq M}$, is called *training data* or *prelabeled data*.
- In case, $c = 2$ one referes to it as *binary classiciation problem*.
- Furthermore, the problem is called a *linear classification problem* if the given data points $x^{(i)}$ can be separated according to their respective labels $y^{(i)}$ by means of hyperplanes. If this is not the case, one refers to the problem as *non-linear*.

Figure 5: Mathematical Definition of Classification

dataset with a large number of attributes and a large number of data points, though, will become excessively resource-intensive, which could overwhelm the computing and processing capacity that is available or need an impractically long duration to generalize and predict accurately. The latter is called the curse of dimensionality. Therefore, it is vital to have a thoughtful and methodical pre-processing approach with the dataset before any training to establish a model that maximizes success from the start.

The following sections take a more focused look at the usage of some prominent ML models, as classifiers. Many other models can be used as classifiers; however, the present study concentrates on several specific models.

3.1.1 Deep Learning

Deep Learning is a subclass of ML models and algorithms that also aims to imitate the way the human brain neurons learn. In Deep Learning, at least three layers of artificial neurons are interconnected in different ways to break down the existing features in the dataset into other sub-features without human extraction, as in ML. This approach is used because it learns to predict more accurately. Hence, the biggest difference between other ML and Deep Learning is that the former requires human help to extract the features and only learns through them while deep learning not only does not require the human extraction of features but also creates its own as mentioned earlier. Furthermore, while most other ML algorithms and models are generally linear or polynomial, Deep Learning is a stack of hierarchical layers of Artificial Neurons with much higher complexity and abstraction. However, it uses the same concept of feed-forward and feed-backward to find the absolute or at least one deep local, minima on the hyperspace boundary that it creates to separate and segregate the classes it is learning to predict.

It is often said that the Deep Learning algorithms resemble a black box where input goes from one side and prediction comes out the other. It remains unclear how the model determined how to make accurate predictions, primarily because of the human inability of imagining higher dimension spaces. Therefore, Deep Learning is not suitable when feature importance is a desired output along with the accurate prediction. However, Deep Learning is extremely useful with large datasets with higher dimensions as it could benefit from the high dimensionality to produce more and more sub-hidden features to learn from inside its so-called Black Box.

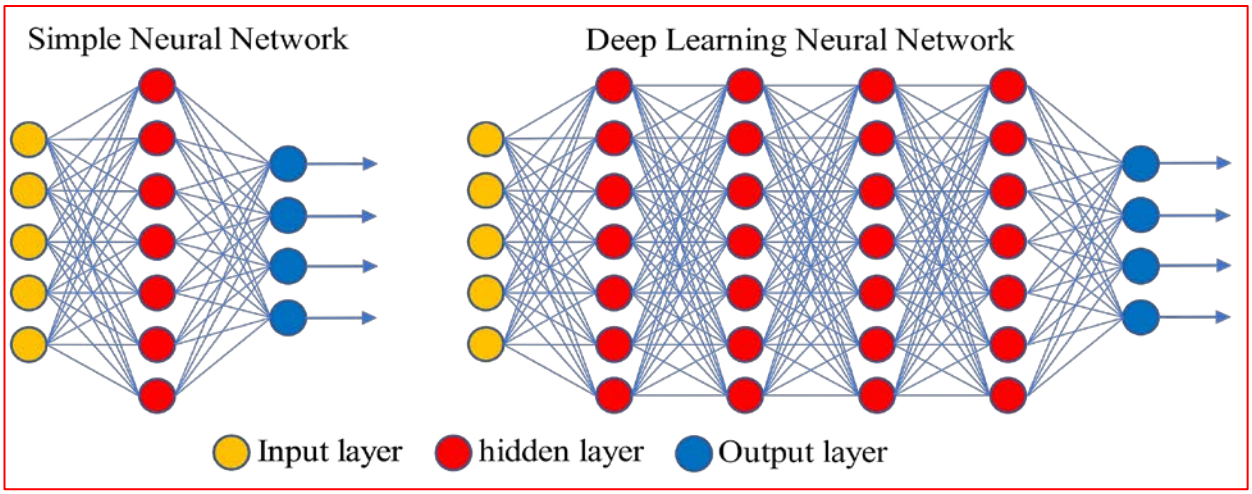


Figure 6: Illustration of Deep Neural Network vs. Simple (Machine Learning) Neural Network

Deep Learning provides exquisite results when it deals with vision, NLP, or numeric problems in general. However, it is usually not the best recommendation for discrete tabular data as it requires an extensive enumeration to be able to process them. One of the most common approaches for enumeration is one-hot encoding where a tabular feature is replaced with as many features as the values inside the original feature and assigning one for only one of the values and zero for all the other new features. For example, if we have a feature that has three values—Red, Yellow, and Green—these features would be broken and replaced by three new features if each of

them expresses one color only. Then, 1 is assigned to the record's actual color, such as Red, and the other two colors would be assigned zeros (see Figure 7).

Color		Red	Yellow	Green
Red		1	0	0
Yellow		0	1	0
Green		0	0	1
Red		1	0	0

Figure 7: Illustration of One-Hot Encoding

One of the biggest drawbacks of Deep Learning is that it is data and resources hungry. This is the reason why, unlike other ML approaches, it only became popular in the past decade when computing powers exploded, and Big Data came to life. This problem is further exacerbated with tabular data as the number of features could increase exponentially for features having lots of unique values, such as country of citizenship. Despite being not recommended for the tabular data in the present research, the research computer used for this study was powerful enough to experiment with Deep Learning, which was done in case it could provide better results.

3.1.2 Tree-Based Methods

Trees are a prominent hierarchical data structure in computer science. They are composed abstractly of nodes and edges connecting them. Any node that produces other nodes below it is called a parent or father node while the nodes below are called children or siblings relative to each other. All nodes must have a parent except the top node which is called the root. Also, when a child does not produce any more nodes, it is called a leaf and represents where its branch of the tree ends. Trees have a lot of uses in computer science as their hierarchical abstraction is conducive to

and can be built with several programming languages, especially Object-Oriented Programming languages such as Java and Python. Python is the language of choice for all models built in the present study.

Trees, in the supervised ML context, are called Decision Trees because they progressively split the feature space of the dataset to optimize the information gain in ways that mimic the human decision-making process. ML trees are composed of decision nodes and leaves. Decision nodes are what splits data around features and leaves are the outcomes, at any subtree, of the decision node. People use decision trees intuitively every day. For example, one might ask oneself whether or not to take an umbrella to work. If the weather is cloudy (Decision Node), then Yes (Leaf 1) and if not, then No (Leaf 2).

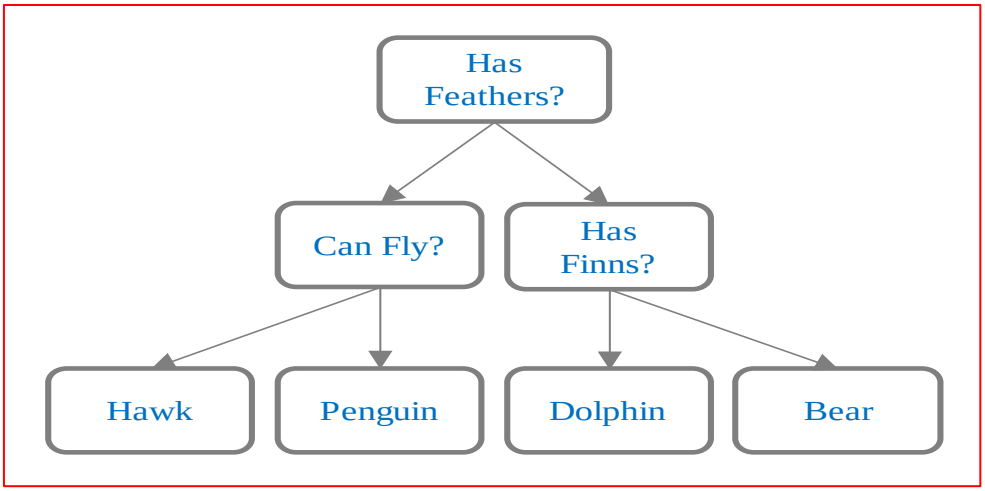


Figure 8: Basic Decision Tree Example

Decision trees in ML have become widespread in recent years due to their unmatched efficiencies with tabular data for two reasons. First, data in real life is mostly tabular in Excel workbooks and similar tables. Second, tree-based methods have achieved leaps in excellence and efficiency that they became the most recommended solution, compared with state-of-the-art Deep Learning, due to their superiority in efficiency and accuracy with tabular data. The most contributive factor to this superiority is the ability of trees to work with tabular categorical features

as they are, while Deep Learning must create many more features to decipher one categorical feature through algorithms such as one-hot encoding. Thus, to manage additional matrix multiplication calculations, Deep Learning becomes excessively resource-hungry. It is also important to note another major difference between the two ML approaches: Tree-based methods are deterministic while Deep Learning is largely probabilistic.

Gradient Boosting is one of the most successful approaches that enabled tree-based methods to overtake Deep Learning with tabular categorical data. In this approach, the model is an ensemble of weak learners, which in tree-based methods are usually trees with a single split. However, unlike other ensemble methods, boosting the weak learners that are combined sequentially in such a way that each subsequent tree enhances the error of the previous one. This enhancement is done using the derivative of the loss function with respect to the previous tree's output. This derivative is what is known in mathematics as Gradient, hence the name, Gradient Boosting. Many loss functions can be used with boosted trees. However, the most common one is the Cross-Entropy, which is demonstrated in Equation 3.1 below, where y is the actual label/target value (0 or 1 in binary classification), $\hat{y}$ is the prediction value, and k is one class out of the K classes ($K = 2$ in binary classification).

$$L(\hat{y}, y) = -\sum_{k}^{K} y^{(k)} \log \hat{y}^{(k)}$$

Equation 3.1: Cross Entropy Loss Function in Binary Classification

The following three subsections quickly review the tree-based gradient boosting libraries, which were used to implement in the present study's experiments.

3.1.2.1 CatBoost

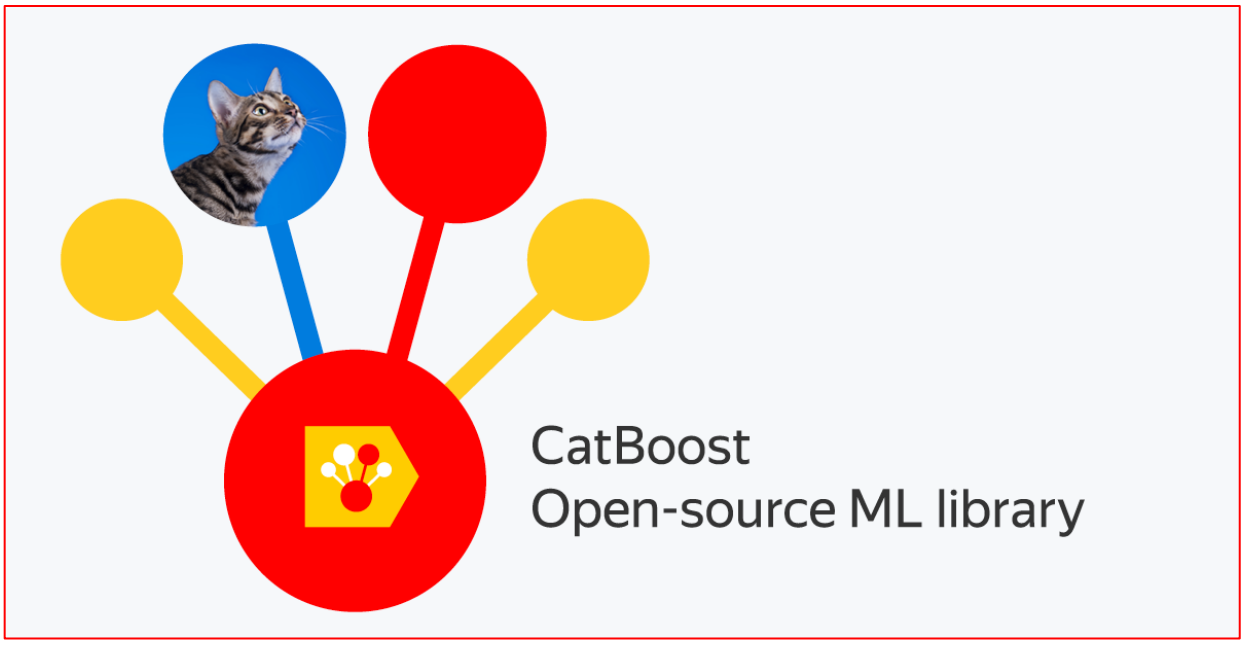


Figure 11: CatBoost Library
Note. Adapted from: Catboot.ai

CatBoost is a tree-based open-source ML decisioning algorithm that was developed in 2017. It came to light by ML researchers and engineers working for a Russian company called Yandex. It was created to help generate Recommendation Systems, personal assistants, weather prediction, and self-driving cars. As it is clear from the name, the algorithm depends on Boosting versus Bagging to optimize its information gain and provide better results while the first half of the name is short for Categorical, which is the data type it performs most effectively with. Bagging is decreasing the variance in prediction by training on additional data created from different combinations of data from the same training set. Boosting, however, is an iterative approach that modulates the weight of an observation based on the previous tree (see figure 12). Some of the advantageous characteristics of CatBoost are its compatibility with numerous data types and its ability to resolve a broad spectrum of problems in a multitude of businesses. It also provides higher

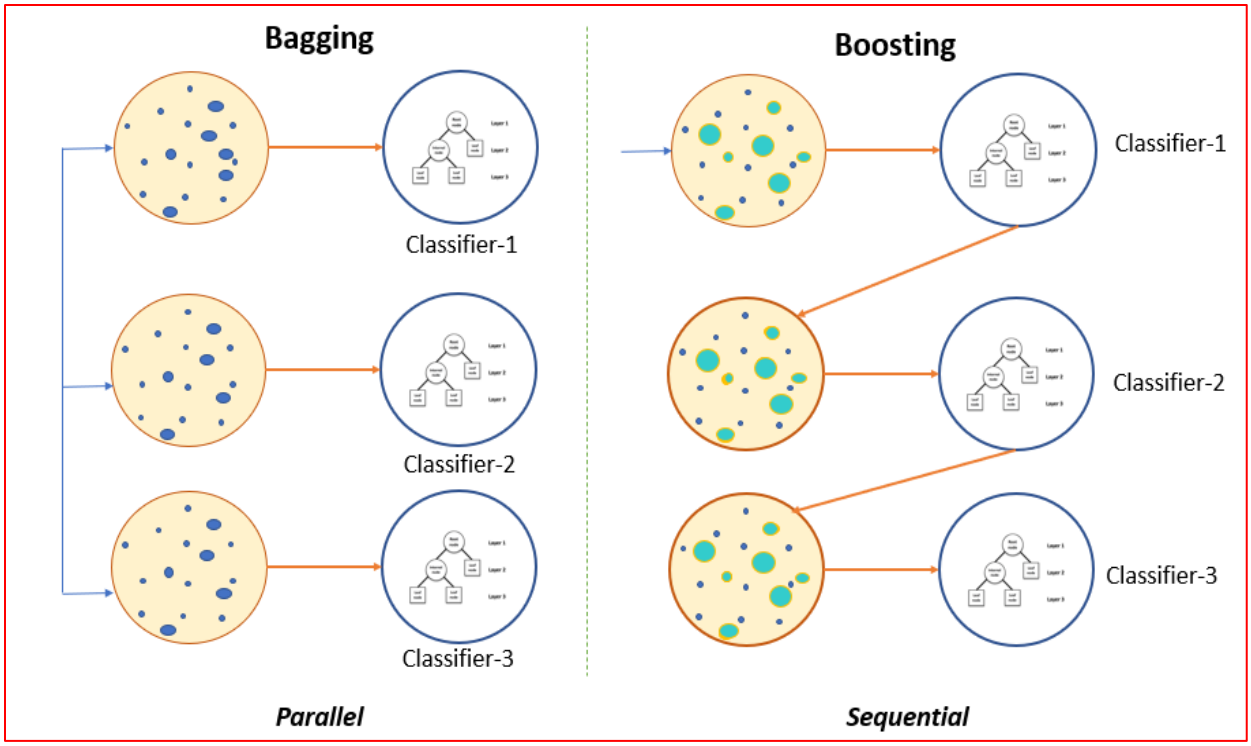


Figure 12: Bagging Versus Boosting

Note. Adapted from: https://www.educba.com/bagging-and-boosting/

accuracy with very efficient resource requirements compared to other ML approaches. Furthermore, despite CatBoost's superior handling of categorical data types, it is also capable of processing other data types, such as numerical data and text data. Although CatBoost can be used to perform both classification and regression, the present study focuses only on classification. Finally, CatBoost enjoys a collection of parameters that can be used to fine-tune the feature space in the pre-processing stage.

CatBoost has many advantages compared to other tree-based and ensemble methods. For example, CatBoost uses ordered boosting where individual weak learner trees train on a random subset of the data and then calculate the residuals to, complete the boosting process, on a different and unseen subset. This practice provides the needed randomization to the model structure and results in less overfitting. Furthermore, the numerous parameters in CatBoost decline the need for hyperparameter tuning. Even with default settings, CatBoost can find the best learning rate and

provide amazing results. Also, with CatBoost, there is no need for the "neumerification" of the categorical data as other ML models do because CatBoost handles this issue efficiently without any explicit support from the developer.

CatBoost models are relatively easy to implement and can be implemented within a plethora of languages and libraries, such as Python using the sci-kit-learn library, R language, and many command-line interfaces. CatBoost also has a superior GPU-supporting version, which is a recurrent problem for ML researchers and engineers suffering to find GPU-supporting libraries to use for their research, especially the ones building on older version models. This makes CatBoost one of the fastest tools in the ML domain that combines, speed, accuracy, and efficiency in one package and is ideal for small datasets. Finally, CatBoost enjoys a large supporting network of users who can provide needed tips and tricks that enabled others to secure accurate results with minimum effort and minimum parameter tuning.

3.1.2.2 LightGBM

Figure 13: LightGBM Library

Note. Adapted from: LightGBM's documentation

Light Gradient Boosting Machine (LightGBM) is a tree-based open-source ML decisioning algorithm that was developed in 2016 by Microsoft and LightGBM researchers. It is considered to be extremely powerful when it comes to speed and efficiency. The fundamental separating characteristic from other tree-based methods is that LightGBM expands its trees vertically rather than horizontally to reduce the loss significantly yet efficiently. The "Light" part

of the name references the small computing power on the CPU, not the GPU. The algorithm needs to give quick accurately predicted results. Unlike CatBoost, LightGBM needs a large dataset, 10,000+ records, for it to work and deliver results effectively. Otherwise, it can easily overfit. However, this was not a concern in the present research, which had access to a large dataset.

Like CatBoost, LightGBM has superior capability when it comes to working with categorical data. While most other ML algorithms use something similar to one-hot encoding to create loads of new columns for just a single categorical column, LightGBM can save the day by just marking the column it is reading as "Categorical" and simply dealing with it as such and without intervention. To help achieve this speed, LightGBM uses both Gradient-based One-Sided Sampling **GOSS** and Exclusive Feature Binding **EFB** along with Histogram-based splitting. The capable handling of categorical data and the speed and low computing cost LightGBM presents are conducive to real-life work environments as it is fast-paced and requires efficiency and minimizing resources to achieve targets. Furthermore, real-world datasets are primarily composed of tabular data sheets with significant categorical data.

3.1.2.3 XGBoost

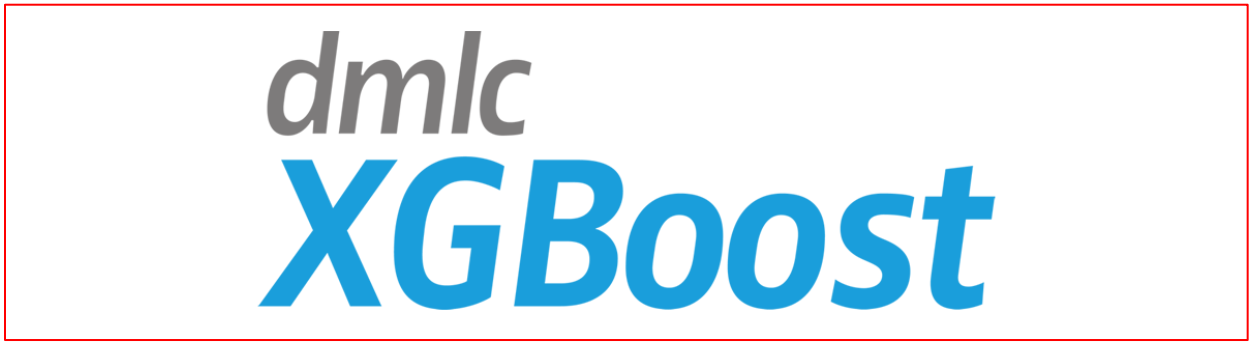


Figure 14: XGBoost Library
Note. Adapted from: XGBoost Documentation

XGBoost is another tree-based open-source decisioning algorithm that was developed in 2016 as a research project at the University of Washington. Since then, it has been widely used in the ML communities and has contributed to many Kaggle competition wins over the years.

XGBoost has a broad spectrum of applications in the fields of regression, classification, ranking, and tailored prediction problems. Moreover, it can run effortlessly on all major operating systems—such as Windows, OSX, and Linux—and it supports a host of programming languages, including C++, Python, Java, Scala, and Julia. Furthermore, it can be integrated into most major cloud systems, most notably AWS and Azure. Its name summarizes its approach, which is eXtreme Gradient Boosting.

There are some important features of XGBoost. For example, it supports three types of boosting: Gradient Boosting, Stochastic Gradient Boosting, and Regularized Gradient Boosting. It also supports each of these with L1 and L2 regularizations as L1 penalizes sums of absolute values of weights while L2 penalizes sums of squares of weight. Furthermore, XGBoost supports a multitude of processing efficiencies. This includes four key approaches: Parallelization, where it engages and uses all cores of the processor; Distributed Computing, where it can build and train a model using a group of networked machines; Out-of-Core computing, where it uses the physical memory to compute if the model was bigger than the CPU memory; and Cache Optimization, where it efficiently utilizes the available computing resources maximizing the hardware performance. Finally, XGBoost can effectively handle sparsely populated features, which is a common problem in real-life datasets, including some features in the present study's dataset. It even supports parallelization at the level of tree construction, and it supports continuous training, where models can be further boosted with the addition of any new data.

3.2 Performance Metrics

Performance metrics are integral parts of any ML algorithm. Knowing how good or bad a model does is what differentiates between a good ML model, and blind guesswork. Furthermore,

they guide researchers on where to go to further optimize and achieve better performance with the model. The loss function inside the model measures the performance of each epoch/iteration/etc. However, loss functions have a completely different purpose, which is to help the model “optimize” its predictions in run time, and they are most likely differentiable to be used in backpropagation or the backward feedback that enables the model to optimize its weights of the features to work better. However, performance measures are usually applied at the end of the run time to judge and give a final verdict on how the model performed the task overall in that particular run with particular parameters, including the chosen loss function. There are a lot of different performance measures: Some are better suited for regression, and some can do better in classification problems such as the problem addressed in the present research.

In classification, the Confusion Matrix is a fundamental metric that is used as the basis for the more advanced classification metrics. The Confusion Matrix is a simple table that organizes four typical values that can be easily observed from the classification operation: True Positives **TP**, False Positives **FP**, True Negatives **TN**, and False Negatives **FN**. Typically, it looks similar to figure 15 below. To understand this, it is important to discuss the classification performance measures used in the present research using the aforementioned four components. This can provide an understanding of the interconnections between them and what sets them apart.

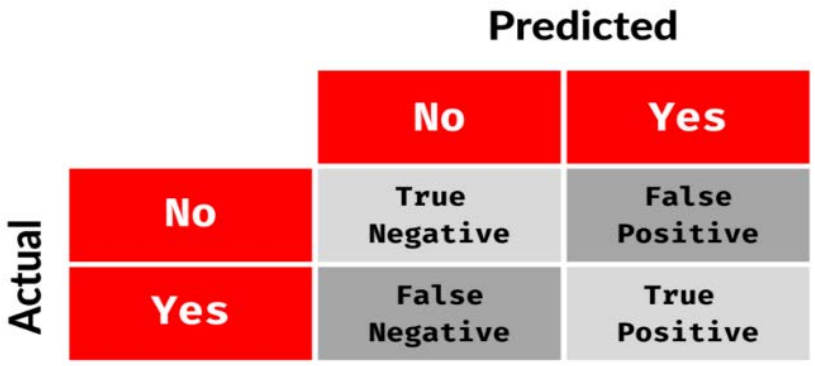


Figure 15: Confusion Matrix

3.2.1 Accuracy

Accuracy is the most commonly used metric in classification problems in ML. Although most common, it is not necessarily the accepted clearest indicator of models' performances. The formula for accuracy is the following:

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

Equation 3.2: Accuracy

Due to this formula, accuracy does not perform well when there is an imbalance of classes as it would reward bias. For example, one needs a model that can predict humans who have a malignant tumor or more. Malignancy is not predominant in humans, so one can assume, for the sake of this example, that the ratio of people in this class is 10% of the population. In this example, FN is way more important than FP because mischaracterizing someone with a tumor as healthy may have dire consequences. If the model simply predicted everyone in the sample to be healthy, that would have an accuracy of 90%. However, the reality is that the model failed to catch 100% of the tumor patients despite the high accuracy of the model got. Due to this bias issue with imbalanced data and other issues, more performance measures were needed and used.

3.2.2 Precision

Another common performance measure in ML is Precision. Unlike accuracy, Precision rewards models with a low rate of FP regardless of TN and FN. In other words, the model is a "precise" measure of how correct the model is when it predicts positive results only. The formula for this measure is much simpler than accuracy and it can be seen in the following:

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

Equation 3.3: Precision

If this model is applied to the previous tumor example, and if the model can predict 15 positive cases with tumors out of the 100 cases, then precision would be 10/10+5 or ~67% while accuracy would be 95%, which is misleading if finding tumor patients was of more importance, especially since their class weight is small in this case example used in the present study.

3.2.3 Recall

The recall is similar to Precision when it comes to how common it is used in ML Problems, but unlike Precision, Recall rewards a low rate of FN regardless of FP or TN. In other words, Recall is a measure of how correct the model is when it predicts positive results in comparison to all actual positive results. The formula for this measure is

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

Equation 3.4: Recall

If this model is applied to the previous tumor example, and if the model can predict 15 positive cases with tumors out of the 100 cases, then precision would be 10/10+0 or 100% while accuracy would be 95%, and Precision was 67%. This extremely high performance is an indicator that the model, hypothetically speaking, would never predict a case as negative while it is positive. This also indicates that, while the model can classify some healthy people as having tumors, it will never classify someone as healthy while they have tumors. In our case, that would be an ideal indicator that the model does not miss patients with this serious issue, which is desirable.

3.2.4 F1 Score

F1 is simply the harmonic mean of both Precision and Recall and utilizes both of these better performance measures. This rewards models that have high Precision and Recall. In other words, the model accurately predicts TP and effectively avoids FN. The formula for that measure can be seen in the following:

$$\frac{2 * Precision * Recall}{Precision + Recall}$$

Equation 3.5: F1 Score

If this performance measure is applied to the same example, where Precision is 67% and Recall is 100%, the result would be 2*0.67*1/1.67, which is 80%. This, so far, can be the most indicative score, given the data class distribution and other measures calculated. One undesirable aspect of the F1 Score is that it gives equal weightage to both Precision and Recall alike. In some cases, one would be more significant, operationally speaking, to the problem owner. Therefore, the following performance measure might be able to help in these cases.

3.2.5 ROC-AUC

Receiver Operating Characteristics (ROC) is simply a graph that plots the relationship curve between the TP Rate and the FP Rate. As they both increase, the curve will show which of the two rates increases faster. Below are the formulas of TPR and FPR:

$$\text{TPR} = \frac{TP}{TP+FN}$$

Equation 3.6: TP Rate

$$\text{FPR} = \frac{FP}{TN+FP}$$

Equation 3.7: FP Rate

Having TPR growing faster than FPR is the desirable case in this performance measure. Therefore, a 90° curve turning at the upper left corner of the graph (see Figure 16) is the ideal case that will never happen when TPR is maximum and FPR is zero. Alternately, the most undesirable case is in the lower right corner when TPR growth is zero and FPR is maximum. In reality, curves in between those two extremes would give a good visual performance measure of how the proposed model is performing. The higher the AUC gets, the better performance the model will have. For all the characteristics this study discussed here, ROC-AUC, is considered to be one of the best performance measures in ML classification problems and the gold standard to get the best

representation/understanding of a model's performance.

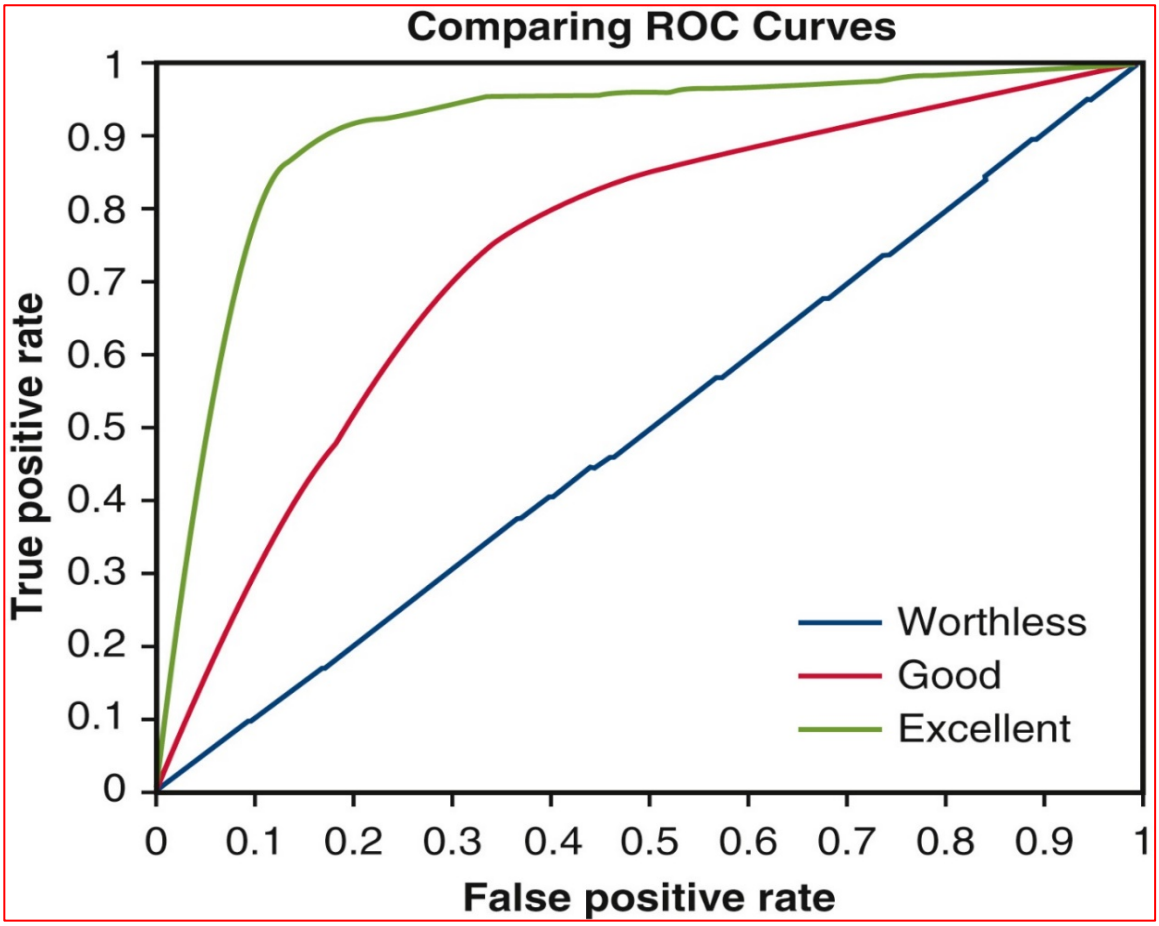


Figure 16: ROC-AUC Illustration

Note. Adapted from: The Journal of Thoracic and Cardiovascular Surgery (jtcvs.org)

Chapter 4

Methodology

The methodology chapter describes the present study's dataset and sheds some light on its unique qualities. It also discusses how to selection of the tree-based approach to solving the problem examined in the present study and explains why this was the optimum approach for this type of problem.

4.1 The Dataset

The present research used a proprietary dataset owned by a US-based entity that runs a humanitarian loan program. The loan aims to extend some help to borrowers who are a less fortunate segment of the population of the USA and who did not authorize the data owner to use their Personally Identifiable Information (PII); therefore, the owner did not share the data to be used in the present study. According to the capacity of the borrower to repay, the loan is interest-free, penalty-free, and term-flexible.

The credit reporting system in the USA, like in Canada, is governed at a high level by federal government authorities. This process is managed at the client level by three enterprises that each have a slightly different model of scoring: TransUnion, Equifax, and Experian. Most of them do the same job in Canada as well. To explain how this process works in simple terms, these

enterprises have created models to score all borrowers' historical financial information and assign a score from 300 up to 850 in the USA or 900 in Canada.

If a person had had bankruptcy, default, any other derogatory information, or if they are a newcomer to the USA, their credit score would be very low, at best. This means that if they go to any financial institution asking for a loan, the institution will decline to give them the loan as soon as they give the lender consent to look up their credit score (the industry jargon is "pull their credit report") and the lender would find that their score is low. Any person in the USA in that position will encounter significant hardship living in one of the most debt-driven economies in the world. The only remedy to that situation is the process of "credit building."

The credit building process is simply applying for small, mostly high interest, and/or secured loans that the person deliberately and systematically pays on time so that they counter the effect of their previous derogatory credit history by demonstrating a new behaviour of honoring commitments over a long period. The process usually entails small increases in their credit score, based on which they can start getting bigger, lower interest, and/or unsecured loans. Over an extended period, years not months, and as they continue their new pattern of strict repayment on time, their credit score would become high enough to get a serious loan amount to buy a vehicle or a home property.

This loan furnished by our data owner also gives borrowers a very affordable opportunity to rebuild their credit scores in the USA as they pay back their loans as agreed with the lender. The lender, as a creditor of the loan, has the same obligations any other financial institution has when it comes to meeting the credit reporting rules and regulations stipulated by federal law. Therefore, this loan program model relies heavily on the opportunity for a "free" credit building process with no penalties or interest, as an incentive along with the goodwill promise of repayment, the program

owner takes from borrowers. This was also demonstrated by the very high recovery rates, which is could be inferred from the dataset.

With an understanding of the framework of credit reporting in the USA, it is possible to discuss the present study's dataset in more detail. The proprietary dataset was compiled from seven table entities from an SQL database that the owner manages. There were clear primary keys linking between those seven tables with different types of relationships such as one-to-one, one-to-many, etc… The data contain over 680,000 records of borrowers and their families and over 51 attributes (features) that include the date of birth, family size, education levels, age, criminal convictions, and languages spoken. The feature space included attributes related to borrowers' repayment history and patterns. However, since future borrowers, whose repayment performance the present research is trying to predict, largely have no relevant/positive credit history by the time they qualify for this loan, all financial attributes were discarded from the learning process, except the total loan amount and the amount of monthly installment per borrower.

This dataset, like most real-life business-related datasets, is a tabular set. Most of the features contained discrete values, numerical or textual. However, there was a small number of continuous features as well. Some features were sparsely populated, and some had a very large number of unique values (over 30,000 unique values in one feature). There was also a surplus of values that were others or unknown in most of the fields. While the large number of records initially seems optimistic, only adults sign and pay for the loan, not the minor children in the family.

Accordingly, a binary feature was used that has the value "1" for loan signatory or their cosigning spouse and 0 for all other family members. After applying this as a filter to get the loan signer alone, the count went down from 680,000 to roughly 412,000 records only. The

preprocessing and some strategic feature selection continued to get the dataset to have the best opportunity of training the model to predict the repayment performance of future borrowers.

Two classification targets (labels) for the data were given by the dataset owner. The two labels were columns 52 and 53 in the combined master table this study created after the 51 original attributes of the feature space. Labels in 52 were a binary classification target, and in 53 was a three-class target. The binary target divides the population of the dataset into "Desirable" and "Undesirable" classes referencing their historical repayment performance (see Figure 17).

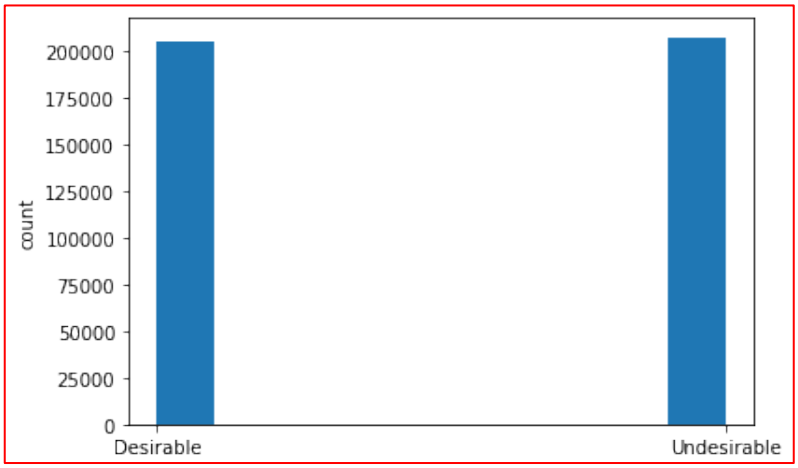


Figure 17: Binary Target's Balanced Distribution

The tripartite classification target divides the population of the dataset into "Below Average," "Average," and "Above Average." Performance measures that were used with the models have unanimously shown that higher measures are always obtained with the binary targets rather than the tripartite target (see Figure 18).

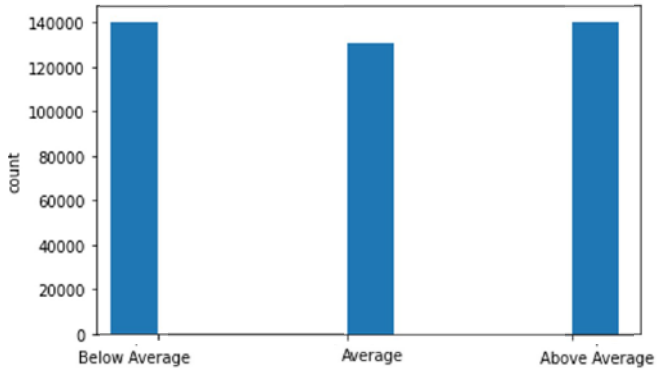


Figure 18: Tripartite Target's Balanced Distribution

Although both cases saw an almost complete class balance in the dataset, which is good to decrease the bias, a closer analysis of raw scores in the data demonstrated that the data is distributed evenly in a Bimodal distribution. The bulk of the data was clustered around the two extreme ends while the section of the data in between the two edge peaks, although similar in number to either of the peaks, was very flat and carried values that would make a model prediction difficult in the middle region. Consequently, predicting the binary classes had much higher performance measures; therefore, it was decided that the experiment would be the focus when building the final model.

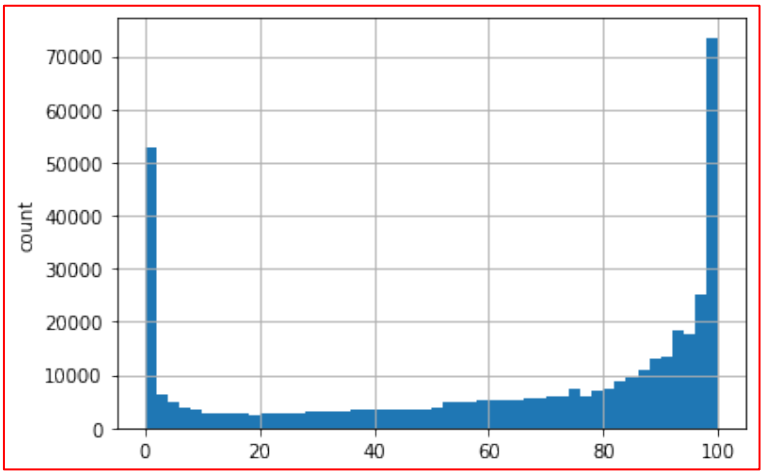


Figure 19: Binary Target's Bimodal Distribution

4.2 Trees Use Case Analysis

Multiple factors validated the use of tree-based methods to build predictive models learning from the present dataset. This section of this chapter goes over the rationale and the advantages and disadvantages of these algorithms with the type of data in the present dataset.

4.2.1 Advantages This Study Utilized

Like most real-life businesses, the proprietary dataset used in the present study was a tabular set of attributes of individual records organized in a massively long table. In this case, most

of the attributes in the feature space were also categorical attributes that have textual or numerical values with two characteristics: (1) they are discreet (non-continuous values), and (2) the values are non-ordinal (without any natural order or rank). This type of data needs a lot of supervised efforts in the pre-processing phase in all other ML Algorithms. In most cases, each categorical attribute with these characteristics needs to be encoded into a set of new made-up attributes that can be easily processed by the algorithm.

However, a decision trees approach uniquely stands out from other ML algorithms in that it requires significantly less effort and no supervised encoding at all in the preprocessing phase of the model training. This is possible because the decision trees approach is categorical-data-friendly naturally as it simply creates subtrees to split the data around values. In other words, a decision trees approach creates sub-trees rather than artificial attributes for categorical values in the feature. This is by far the biggest advantage that made trees a natural choice for us.

Additionally, the tree-based approach is advantageous as it eliminates the need for two operations: Normalization and Scaling. These operations are confused with each other so frequently that a great sector of the math and statistics community use them interchangeably. However, the best distinctive definition between them is that **Scaling** is modifying the **range** of the data while **Normalizing** is altering the **distribution** shape of the data. An example of scaling would be modifying a data range that was from zero to five to from zero to 100; an example of normalization would be reshaping normally distributed data to a skewed or bimodal distribution. Now the reason why decision trees do not need both normalization and scaling is that they have a low bias (there is no preconceived restriction on the model shape) and high variability (the model performs differently with different data), which make trees agnostic to both operations.

Another common characteristic in real-life data is missing values. In day-to-day business data capture, it is common to have unknown/null values. While this poses a significant challenge to most other ML algorithms, this does not hold for tree-based methods and algorithms. Trees handle those values well because they have several options to use, such as considering “Null” values as an additional category, simply ignoring the null/unknown value, or finding correlations between different features to conclude a missing value based on the most common value attached to a similar correlated value in the fully populated feature.

Finally, in ML, it is commonly recognized that non-tree models calculate their prediction through the “Black Box.” This name comes from the fact that most ML algorithms decipher the feature space and create sub-features that mostly do not make sense to the human brain’s interpretations and those models are impossible to be visually represented, or difficult at best. However, decision trees are simple to represent visually and comprehend by average human intelligence and stakeholders who are on the business side of the ML solution. Furthermore, this simplicity also enables knowing the feature importance and ranking inside the model, which was the secondary requirement from our dataset owner.

4.2.2 Disadvantages This Study Worked Around

Due to the deterministic nature of the decision trees approach, its defining characteristic is having a low bias and high variability, as explained in 4.2.1. This means that tree-based models are very unstable and can change the tree structure significantly as data changes. To work around this issue, the present study resorted to tree libraries that allow boosting and ensemble approaches. The idea is that the tree instability is true if the whole model is consisting of a single big tree because its structure will surely get altered as data changes.

However, using the ensemble approach changes the structure from one tree to the aggregate of weak learner trees, and each of them is small enough not to be affected by data change. Furthermore, boosting is simply using sub trees to finetune the results of the previous tree. Boosting process is very similar to the backpropagation of a neural network. When used together, Ensemble and Boosting significantly enhance the stability of the tree model as a whole and decrease its variability. Thus, CatBoost, LightGBM, and XGBoost because they utilize these approaches.

The other disadvantage decision trees have is that they are not suitable for regression problems due to their deterministic nature as mentioned above in 4.2.1. However, this disadvantage was irrelevant to the problem examined in the present study as the dataset was overwhelmingly categorical, discreet, and non-ordinal values, and the labels were provided as a classification problem.

4.2.3 Result Verification

Verification of achieved results was verified outside the API libraries used for training and testing. Results were frequently and methodically exported into CSV files that would either be sampled and verified manually using MS Excel or electronically using Python code. The core of the verification relied on ensuring the accuracy of the reported results, how they were concluded, and how they compare to the actual label values. After thorough external verification, the author of this research was very confident the algorithms and APIs used were working as intended and providing accurate results and observations.

Chapter 5

Experiment

This chapter goes through the details of the experiments and how and why these experiments were chosen in place of other approaches.

5.1 Building the Models

All of the models that were experimented on were built using Python as a programing language. For the few Neural Networks, the Keras library was used to build the model's layers and specify the size, activation function, regularizer, etc. Keras with Python provided some of the most agile libraries that can be used to build neural networks. While neural nets can be built by a host of other Python-related libraries, nothing was as powerful and efficient as Keras. The layers that had required long pages to design could almost entirely be replaced with a single line of code in the library of Keras.

5.2 Data Pre-Processing

As mentioned briefly in the previous chapter, the data in the present study was compiled using a dataset from seven separate tables extracted from an SQL database. Using the linking primary keys, all the attributes that were needed were linked and concatenated for one individual

in one record line in the resulting dataset. This produced 53 attributes in the feature space and two labels that were provided by the data owner. The attributes provided biographic and psychographic data that included categories such as age, education, employment history, military service, and law violations. As for the labels provided, one of them was binary 0/1 indicating Undesirable Performance versus Desirable Performance, respectively, while the other was a tripartite 0/1/2 referencing below average, average, and above average, respectively. All pre-processing work described in this section was executed by Python commands.

In almost all the seven tables that the dataset was compiled from, the number of records did not match, and some primary keys were available in some tables and not in others. Although this was a little baffling at the beginning, the explanation was found: Some borrowers either (1) did not have information reported in the tables they were missing from or (2) the attributes of that table did not pertain to them. For example, the military service table had only a very small subset of the data for the individuals who were in the military. As a result, when the consolidated dataset was compiled, Null was used to fill in the attributes that were missing/inapplicable to the record borrower, and all their other information was compiled in the rest of the attributes. There was a very small subset of borrowers who were missing almost all data; thus, it was decided that, given how sparsely populated they would have been, they were completely removed from the dataset to avoid adding noise to the data. The total number of records that were discarded was approximately 4,000, which is under 1% of the data size this study ended up using.

Another notable effort included ensuring that all primary keys were in lowercase, which was not true in all tables. This change was vital because Python is very case-sensitive. Other work included handling one-to-many tables. For example, in table education, the same individual could have more than one record due to them obtaining more than one education degree. This issue was

solved for this table and others, such as employment, by selecting only the highest education level to maintain this data in a single line record in the compiled dataset. Furthermore, calculated attributes were created with this type of data to count or add the incidents or their totals. This was done to avoid missing the value of the omitted data. For example, in education, a column was added that calculated the number of degrees earned while in employment, and another column was added to calculate the total number of years of employment. The same was done again with languages and their proficiencies but in that case, English and Spanish languages were made available for all individuals before a column was added to count the number of languages spoken.

Efforts were also put concerning handling some format mismatching or typo dates. All future and unrealistic older dates (in the 1700s in one case) were replaced with nulls indicating that there was no correct information available. In some cases, it was not possible to conclude and calculate missing or erroneous dates based on other connected dates that were available. Furthermore, the format of all dates in the seven tables was unified. This was applied to many table attributes, such as military service and law violations. In both former example cases, it was assumed that the individual who did not appear in these tables was a person who did not join the military or had no law violations. For those who were in these tables, the date format was unified before all but the most recent incident was removed and a column was added to calculate the number of years in service or prison. Finally, the most time-consuming pre-processing involved dealing with textual data attributes that had a large number of unique values that would create noise and would not add learning to the model training. In those attributes, the data were studied meticulously and then grouped into fewer categories. In one of the attributes related to illness/disability history, there were almost 110,000 unique values that would look similar to anybody without any medical experience. Therefore, this attribute was binarized by changing the

values inside that column to 1 (has illness/disability) if it had any textual data and 0 if Null (no illness/disability).

At the end of the pre-processing effort, the master dataset had 47 attributes, 29 of which were categorical and 18 numerical/continuous attributes. It is important to note again that the financial attributes were discarded except for the total loan amount and the amount of the monthly installments. This was done because the new borrowers would not come to the data owner lender without any reliable and/or non-derogatory credit reporting. This allowed all the predictions to be based only on the biographic and psychographic information, which the new borrowers must have to be able to benefit from the data owner's loans.

5.3 Models Used

The present study experimented with various types of Tree-based Algorithms and with both targets that were provided by the data owner: the binary and the tripartite. Finally, once the best model was identified, the same model was applied to the Taiwanese "Default of Credit Card Clients," which is publicly available on the dataset repository of the University of California in Irvine (UCI). The following is a detailed record of our experiments and how they were implemented:

5.3.1 Binary Classification

Binary classification is simply classifying between two targets or one of only two choices. More likely than not, binary classification usually aims at distinguishing between a "normal" state and an "abnormal" or "unwanted" state. This also holds for the present study as the data that was provided had labels indicating whether the historic performance of the record was "Desirable" or "Undesirable".

Due to the data owner's restrictions, a completely de-identified dataset was received. Furthermore, based on the success of the present research and to ensure that the data were independent and identically distributed (IID), the data were repeatedly checked and verified using Python. This was achieved by running statistical analysis on random samples and ensuring that classes remained balanced and that no time series was in the data. The data was then divided in an IID manner into 10% for final testing, and 90% for training and validation. K-Fold Cross Validation was then used to ensure that ALL the data were used at some point in training and were used for validation. The K of choice was 5 as it was realized, through research and experimentation, that it provided the balance between bias and variance in our models while maintaining efficiency as well in computing resources and training time. Once the training was finished and the performance measures of all folds are roughly equal, a model was trained using the same K-Fold configuration on the whole 90% set and tested with the hidden 10%. It was clear that the optimum performance had been achieved when the performance of that final model was still in range with what was seen with the five folds. As mentioned in earlier chapters, three metrics were used: Accuracy, F1, and ROC-AUC. The average scores of the folds would be used as the final score and compared later with the 10% final test set for confidence, which will also be used to evaluate future model updates.

In any classification problem, the class imbalance could be a significant hindrance to achieving meaningful performance. For example, in a binary class classification where one of the classes makes up 90% of the data, if the model predicted all 100% of the cases with the larger state and made no predictions at all of the 10% class, the accuracy of that model would theoretically be 90%. However, if that 10% are cancer patients, it would be problematic to miss them and claim that 90% accuracy had been achieved. Therefore, when classes are imbalanced, some techniques

may be applied, such as lower-class data augmentation or algorithms including Random Under-Sampling and NearMiss. However, the two classes provided by the data owner were very closely balanced. Moving forward, it is important to discuss the tree-based approaches that were experimented with in the binary classification target

5.3.1.1 Scikit-Learn

Initially, the present study experimented with the tree-based models available on the famous Scikit-Learn library for binary classification. This included ExtraTreeClassifier, RandomForestClassifier, DecisionTreeClassifier, and AdaBoostClassifier with DecisionTreeClassifier as base estimator. One of the most interesting models was AdaBoostClassifier as it was a tree-based ensemble approach that creates n base-estimators (weak learners) with every subsequent estimator giving higher weights to samples where the whole ensemble is performing poorly on, thereby enabling the subsequent lineage of the tree to focus and learn more about these samples.

After a long round of experimentation with parameters' tuning and setting the max_depth to 14 for all models with this library, the best performance came from the RandomForestClassifier model while the least performing model was DecisionTreeClassifier. All results specifics are discussed in detail in the next chapter. However, although RandomForestClassifier did very well compared to its other siblings in this library, it was not the best model that could be trained. Please read the following sections for more information.

5.3.1.2 Gradient Boost

The study then moved next to the tree-based models using the three libraries using Gradient Boosting, which were discussed in chapter 4: LightGBM, XGBoost, and CatBoost. Surprisingly, it was found that those three libraries performed well even using their default parameters before

trying any tuning. However, differences arose between the three libraries after doing some parameter tuning to ensure stopping before overfitting. This negative aspect of trees was discussed in chapter 5. Based on the experimentation, the best max_depth for the three libraries was 3.

Once all models were trimmed appropriately and training and cross-validation were done, it became clear that the three models outperformed their default settings by a few percentage-point margins, which in turn outperformed the Scikit-Learn models out of the box. This was the first time, in two years of AI and ML projects, that data from the present study was able to demonstrate that a library could train a model using its default parameters and achieve above 80% results. This illustrates the brilliant minds that worked on and developed them. They made classification problems that much more efficient than other libraries. Just to list another impressive feat; LighGBM only worked on CPU and finished in under 10 minutes and could not even use GPUs.

Among the three Gradient Boosting libraries that the present study experiments with, CatBoost maintained a consistent edge, outperforming the other two by an average of 1-2%. CatBoost was followed closely by LighGBMClassifier, and the final place was for XGBoost. Results and comparisons are detailed in the following chapter along with other experiments.

5.3.1.3 Neural Networks

Early in the research, it was established that Neural Networks do not perform efficiently with tabular data, especially the categorical type. However, for the sake of science and experimentation, it was decided to give Neural Networks a chance, especially since the present study had access to a massive GPU in the computer that was used for the present research. Due to the mostly categorical data with hundreds of unique values in some attributes, the Neural Network ballooned the feature space to 87,000 trainable parameters, which required massive computing resources.

Many parameters and layer architectures were experimented with, and finding the best possible combination, a surprising result arose: While the model seemed to outperform anything previously tried in the experiment before in training, the results in the evaluation were on a completely different performance level. On the highest performing model, there was around 9% lower ROC-AUC and 8% lower Accuracy compared to its evaluation. This was a clear mark that despite concerted efforts, the Neural Networks model was overfitting and memorizing the data rather than learning from it.

The powerful Keras library was used to implement these models and utilized its Embedding Layer. This feature allowed the model to dynamically learn the encoding of the categorical attributes as the training continues, which led to better performance. Using StandardScaler, the numerical values in the dataset were scaled and standardized to be more digestible by the models. Next, the encoded categorical attributes were then concatenated, created by the model, and through the use of the non-categorical values that were standardized and used as input for the following Dense Layers. Finally, the model was optimized using the Binary_Cross_Entropy loss function end-to-end.

5.3.2 Tripartite Classification

The data owner provided us with a tri-class (Above Average, Average, Below Average) target and asked us to experiment with it and see how it would perform comparatively against the traditional binary classification target. The initial analysis showed that the data owner, once again, provided fairly symmetrical and balanced classes with good and bad making about 34% each and the average roughly about 32%. Therefore, the class imbalance was not an issue of concern here. From that point, three different models supporting multi-class classification were examined to match the best metrics for the Binary Classification models. However, the best Tripartite

classification models were no match. Therefore, to preserve time and focus on making what is working better, the experiment did not go much deeper in this direction. The following are the details of the experimentation within the Tripartite classification.

CatBoost was used for the first model: The max_depth was set to 8 and used the objective function of “Multi-Class”. Next, Scikit Learn’s ExtraTreeClassifier was used with a max_depth of 8. In Scikit-Learn, the random forest model was used first as it helps reduce overfitting, which is the most common challenge that was found with tree-based Algorithms. The way the random forest algorithm in Scikit-learn works is that it creates multiple randomized decision trees that would work independently on multiple subsets of the data.

The same CatBoost model was then used as a baseline estimator for this ordinal approach. This was separate from the previous models and was inspired by research found in the article “Simple Trick to Train an Ordinal Regression with any Classifier,” which was based on the ordinal classifier this study reviewed in the literature review section [**2.12**].

For metrics of this experiment, F1, ROC-AUC, and even Accuracy were used. Similar to the binary target approach, K-Fold Cross Validation was used to choose the K to be 5 for a balance between results and efficiency. However, it is worth mentioning that the aim of the innovative ordinal multi-model for the multi-class approach was to give higher performance metrics. However, repeated experiments of the multi-model approach failed to defeat a single CatBoost model, nevertheless by being only 1-2% short. Finally, and due to that the CatBoost tripartite model could not defeat the performance of the CatBoost binary model, it was decided to not pursue the 3-class target any further.

5.4 Best Model on A Public Dataset

After running all the experiments on the proprietary dataset, the CatBoost tree was identified as the best model. Furthermore, the best-performing target of the two that were provided by the data owner was the binary target. It accordingly became desirable to experiment with another publicly available dataset with similar characteristics and see if similar or improved results could be achieved. This section discusses the dataset that was chosen and how the experiment unfolded.

5.4.1 Taiwanese Dataset

The formal name of that dataset is “Taiwanese Default of Credit Card Dataset”, but it is commonly known amongst the AI-related credit scoring researchers as the Taiwanese dataset. This dataset was donated by the Taiwanese institutions to the University of California in Irvine (UCI) in January 2016 according to the UCI repository page. The data contains 30,000 records (instances) and 24 attributes, four of which are biographical: Gender, Education, Marital Status, and age. All other attributes are financial history-related, such as the amount of given credit, history of past payments, amount of bill statement, and the amount of previous payment. The target/label is binary and is implied as: "Is this account in default?" and the label answers: "Yes" or "No".

This dataset is one of the larger free datasets that are publicly available for credit scoring and financial/repayment performance research. Furthermore, it is one of the largest datasets when it comes to its feature space with its 24 attributes. Therefore, this dataset has been heavily utilized in credit scoring research for many years now. The data capture a few biographical information about its 30,000 borrowers and much more financial history and performance between April and October of the year 2005.

5.4.2 Pre-Processing

Despite how popular this dataset is, the amount of “cleaning” that was needed to use it was surprising. To begin, feature X1 (Limit-Balance) needed to be normalized by dividing it by 10,000 to bring it from the range of 10,000 to 1,000,000 to the range of 1 to 100. The main objective was to minimize the needed calculations that would not have provided any additional value, especially since the given values under X1 were all multiples of 10,000 due to the natural value of the Taiwanese currency.

The data were converted to tabular/categorical to make it similar to the previous dataset used in the present research. The numerical target 1/0 was converted to a textual Yes/No, respectively. Feature X3 (Education) was supposed to be in a range from 1 to 4. However, numbers out of that range were found, and the records containing these values were removed. the X2 feature (Gender) from 1/0 to textual Male/Female was also encoded, as was feature X4 (marital status) from 1/0 to textual Married/Single. Some analysis was then done on the dataset target distribution and, unlike our dataset, the target in the Taiwanese dataset was skewed with the “in default” class being seen only in 21.963% of all incidents while 78.037% was not in default class.

Some additional features were likewise engineered, such as the ratio of payment to bills across the whole period. When that was done, ratios were found that did not make much sense as it was very high at times. These seemed to be outliers that could impact the performance of the model, so any record that had a value in the new feature that was greater than 2 was removed. This made it possible to focus on the more “inline” data. All these changes that were added/modified enabled the model to learn better from the data while concentrating on centric data rather than the fringe outliers or values that were not sensible.

5.4.3 Models

The best CatBoost model was tested with the same parameters: max_depth = 8 and number of iterations = 1,000. Moreover, LogLoss was used as the objective loss function. Very competitive metrics were found in the training phase. However, when the evaluation stage began, it became clear that there was a difference of roughly 9% between the training and evaluation in F1 and ROC-AUC and accuracy was 2% different. However, due to that, the data classes were skewed in distribution; thus, the accuracy was discarded. The other metrics indicated clearly that there was an overfitting issue in the work.

To overcome the obvious overfitting problem, a bigger model was experimented with, and it had a higher capacity choosing CatBoost's max_depth = 11 and number of iterations 2,000. Similarly, this model gave much better results in training than the previous model and was very competitive compared to the other ML approaches. However, as soon as the evaluation started, the gap between our metrics in training and evaluation became even more significant: 13% instead of 9%. Only accuracy decreased by less than 0.9%. However, accuracy was discarded as it was only going to be high because the data was skewed. The results and discussion will follow in detail in the coming two chapters.

CatBoost was used on the Taiwanese dataset along with the present study's algorithm, which included preprocessing, cleaning, and feature engineering. The aim was to verify whether CatBoost would be able to meet or exceed the original research performance metrics done by the original authors who first used this dataset. They used several custom metrics including error rate. With this algorithm used during training, it was possible to reduce the error by 2%, and on validation, the results were off their best method by 4% with default settings and less tuning effort.

5.5 System Configuration

Name	Parameter/Version
CPU	Intel i9 10850K
RAM	64GB DDR4 5000 MHz
GPU	RTX 3090, Clock 1975 MHz
Memory	24 GB GDDR6X (GPU)
CUDA Cores	10,496 Cores
Tensor Cores	328 3rd Generation Cores
RT Cores	82 2nd Generation Cores
Python	3.8
CatBoost	1.0.4
NumPy	1.21.1
Pandas	1.3.3
Scikit Learn	0.22

Chapter 6

Results and Discussion

This chapter details the results and how they compare to each other.

6.1 Assumptions, Limitations, & Application

This research is anchored on several assumptions; the assumption that the data collected in the dataset is correct, accurate, and collected scientifically. Also, the assumption that this data is representative of the population of the research. Furthermore, the assumption is that there is no missing or erroneous data in the dataset, and the most important assumption is that no previous public research exists, is available, or has found ML Prediction creditworthiness using only demographics and psychographics used in the approach adopted by the present study.

Several limitations hindered this research's full potential. For example, some data attributes had an enormous number of discreet values, 110,000 in one case. A large number of unique values in an attribute serve as noise in model training and generally declines and hampers the quality training, except if the dataset number of records is exponentially higher than the number of unique values. Another limitation was the sparsity of some attributes, which also decreases the information gain from that feature. A final limitation is the data size, although it is a larger size dataset, given the high number of attributes and values inside them, a bigger dataset would perform better.

As for the application of this research, it can only be applied to problems with the following characteristics: larger datasets, tabular data, and when absolute accuracy is not required. This study's approach and algorithm would not be suitable for medical research, for example, due to the high variability and low bias of the Tress-Based methods. More specifically it will perform better than other algorithms on data that are mostly textual discreet values. The advantage of the study's approach in applicability is that it is computationally inexpensive. While it has the potential to produce good results, this depends on the size and the quality of the used dataset.

6.2 Proprietary Dataset

After experimenting with various algorithms and approaches, the tables below provide an easy comparison for the reader as it illustrates the performance metrics that were achieved will. K-Fold (K= 5) Cross-Validation was used with stratification for all our experiments across the board. The stratification ensures, as much as attainable, that all folds' training and validation portions have the same distribution of targets and feature space, which makes all folds Independent and Identically Distributed. With CatBoost, the GPU-enabled libraries were utilized, which significantly shortened training/cross-validation times impressively. Other libraries were used, such as LightGBM, which did not need GPU-supporting API. However, it was using the CPU cores in an impressive manner that easily managed to put it at par with CatBoost GPU performance. The metric of choice in the present study was ROC-AUC, which was selected due to its suitability to our dataset and the problem; however, other metrics were used and logged as well (please see chapter 4 for more details about metrics).

6.2.1 Binary Classification

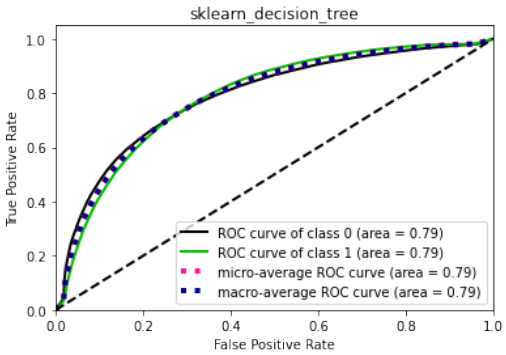
Below is the summary of Cross-Validation scores for training. The best performance is highlighted with a red frame.

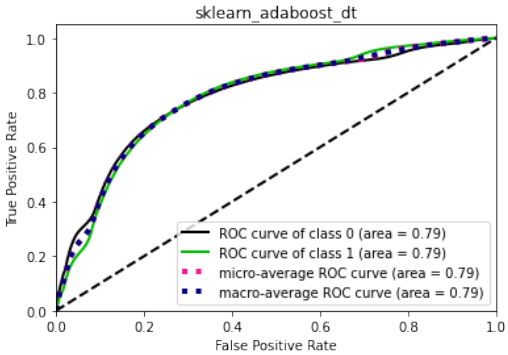
5-Fold Training			
Model	**F1**	**ROC-AUC**	**Accuracy**
Scikit-Learn Decision Trees	**77.38%**	**85.57%**	**77.02%**
Scikit-Learn AdaBoost w/ DT	**86.90%**	**95.36%**	**86.71%**
Scikit-Learn Extra Trees	**75.87%**	**83.40%**	**75.26%**
Scikit-Learn Random Forst	**78.74%**	**86.35%**	**78.01%**
LightGBM	**84.45%**	**92.50%**	**84.19%**
XGBoost	**80.72%**	**88.79%**	**80.37%**
CatBoost	**81.11%**	**89.17%**	**80.71%**
Neural Networks	**81.67%**	**90.01%**	**81.50%**

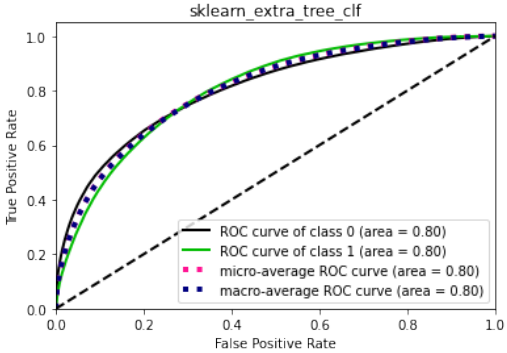
Below is the summary of the Cross-Validation scores for evaluation. The best performance is highlighted with a red frame.

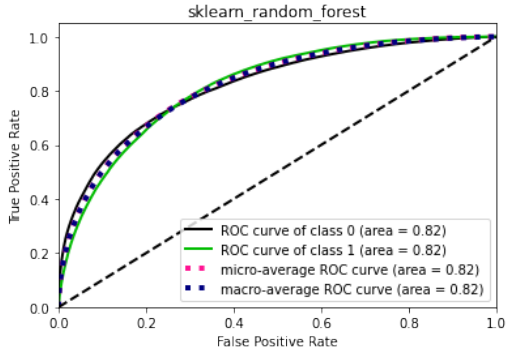
5-Fold Evaluation			
Model	**F1**	**ROC-AUC**	**Accuracy**
Scikit-Learn Decision Trees	**72.69%**	**78.61%**	**72.26%**
Scikit-Learn AdaBoost w/ DT	**73.64%**	**78.55%**	**73.24%**
Scikit-Learn Extra Trees	**73.21%**	**80.39%**	**72.60%**
Scikit-Learn Random Forst	**74.71%**	**81.85%**	**73.90%**
LightGBM	**76.95%**	**84.95%**	**76.61%**
XGBoost	**75.68%**	**83.51%**	**75.28%**
CatBoost	**77.47%**	**85.39%**	**77.03%**
Neural Networks	**73.59%**	**81.23%**	**73.36%**

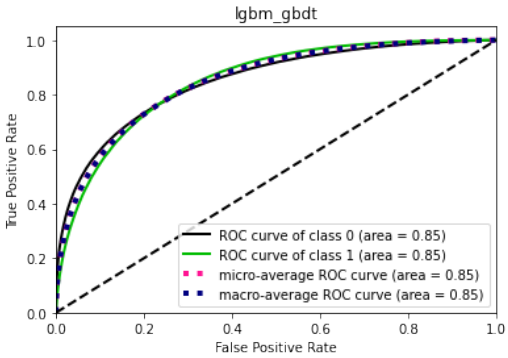
As demonstrated from the above tables, CatBoost was the best model when it came to evaluation scores through the whole Binary Classification experiment. The following page provides the ROC-AUC plots for all the experiments that were run.

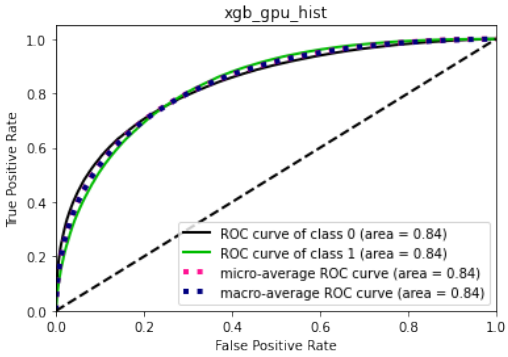

sklearn_decision_tree
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 0.79)
ROC curve of class 1 (area = 0.79)
micro-average ROC curve (area = 0.79)
macro-average ROC curve (area = 0.79)

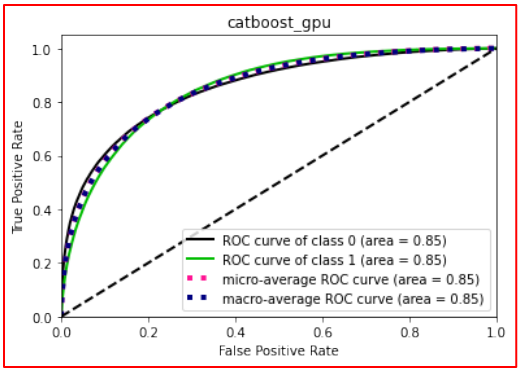

sklearn_adaboost_dt
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 0.79)
ROC curve of class 1 (area = 0.79)
micro-average ROC curve (area = 0.79)
macro-average ROC curve (area = 0.79)

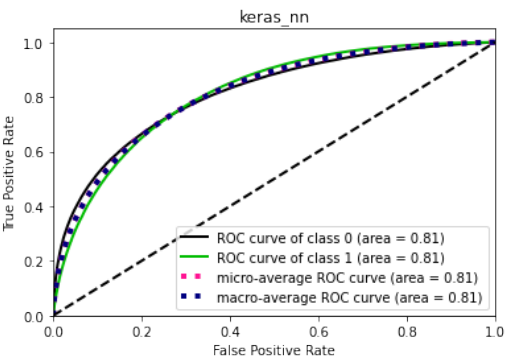

sklearn_extra_tree_clf
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 0.80)
ROC curve of class 1 (area = 0.80)
micro-average ROC curve (area = 0.80)
macro-average ROC curve (area = 0.80)


sklearn_random_forest
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 0.82)
ROC curve of class 1 (area = 0.82)
micro-average ROC curve (area = 0.82)
macro-average ROC curve (area = 0.82)


lgbm_gbdt
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 0.85)
ROC curve of class 1 (area = 0.85)
micro-average ROC curve (area = 0.85)
macro-average ROC curve (area = 0.85)


xgb_gpu_hist
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 0.84)
ROC curve of class 1 (area = 0.84)
micro-average ROC curve (area = 0.84)
macro-average ROC curve (area = 0.84)


catboost_gpu
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 0.85)
ROC curve of class 1 (area = 0.85)
micro-average ROC curve (area = 0.85)
macro-average ROC curve (area = 0.85)


keras_nn
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 0.81)
ROC curve of class 1 (area = 0.81)
micro-average ROC curve (area = 0.81)
macro-average ROC curve (area = 0.81)

6.2.3 Tripartite Classification

After experimenting with the above methods on binary targets, the best binary target model was used and applied to tripartite targets. Therefore, CatboostClassifier was used with `MultiClass` as an objective function, and ExtraTreeClassifier and the probability subtraction method were also used. Below is the summary of Cross-Validation scores for training. The best performance is highlighted with a red frame.

5-Fold Training			
Model	**F1 Macro**	**ROC-AUC Weighted OVR**	**Accuracy**
CatBoost GPU	**66.356%**	**84.620%**	**66.990%**
Scikit-Learn Ex Tree	**49.838%**	**73.296%**	**53.982%**
Probability Subtraction	**67.081%**	**84.922%**	**67.477%**

Below is the summary of the Cross-Validation scores for evaluation. The best performance is highlighted with a red frame.

5-Fold Evaluation			
Model	**F1 Macro**	**ROC-AUC Weighted OVR**	**Accuracy**
CatBoost GPU	**58.697%**	**78.220%**	**59.527%**
Scikit-Learn Ex Tree	**49.425%**	**72.846%**	**53.612%**
Probability Subtraction	**58.909%**	**78.024%**	**59.411%**

Since the primary goal is to perform well on binary targets as requested by the data owner, it was concluded that CatBoost with *max_depth=8* is the optimum model on binary targets. This model was able to beat all other methods on almost all metrics. Using CatBoost and other tree-based models also made it possible to address the second request from the data owner, which is to identify the contributing factors to the desirable and undesirable performances. Using the built-in

feature importance capabilities of these models made it possible to gain a clear knowledge of the most important features, which was shared with the data owner. However, it would not have been possible to publish these results at the request of the owner and therefore, these findings are not illustrated here.

Finally, and for practical purposes, experiments were carried out with regard to the performance of the model with only a subset of the most important features used rather than the full 51 features. This approach was inspired by this approach outlined in the work of P. Addo *et al.* *[2]* when they decided to use the top 10 most important features to enable Loan Officers to decide with new borrowers without having to fill in too many values. Similarly, in the setup for the data used in the present study, future borrowers' data could arrive in compiled data sheets, in which case they can be fed to the model and results would be presented to them en masse. Alternately, in some cases, a specific individual may need to be entered manually. In such cases, the borrowers' data need to be entered manually; therefore, the least number of features entered to provide accurate results, the better and more efficient. The present study's model was tested with the top 15, 10, and 5 features to evaluate the amount of drop in performance. In the cases of 10 and 15 features, the performance only decreased by a few decimal percentage points. However, when this was attempted with 5 features, the performance dropped by almost 4%. Accordingly, it was decided that the "lite" model would be provided to the data owner for efficient manual prediction and would use the top 10 most important features.

6.3 Taiwanese Dataset

It was important to compare whether applying the present study's algorithm would work on a tabular dataset along with the powerful CatBoost library and whether it could achieve better

results on similar datasets. The Taiwanese dataset was chosen due to its close similarity to the research dataset. Original authors of the Taiwanese dataset [21] used custom metrics including

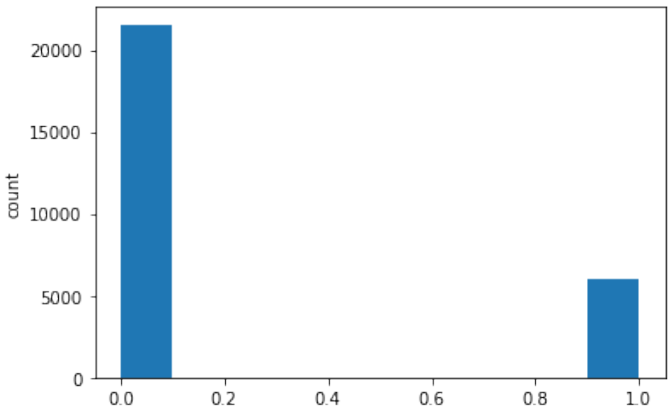


Figure 20: Taiwanese Binary Target's Imbalance

error rate. The dataset is skewed having 78% labeled as non-default and 23% of samples as defaulted, see figure 20.

The present study's algorithm and tools could not outperform the original research of the Taiwanese dataset. However, it was possible to achieve very similar error rates on training and evaluation splits using 5-fold Cross-Validation. Below is the summary of Cross-Validation scores for training and evaluation for our both models with *depth = 11* and *depth = 8*.

5-Fold Training		
Model	**ROC-AUC**	**Accuracy**
Depth = 11	**86.43%**	**83.01%**
Depth = 8	**82.61%**	**81.65%**

5-Fold Evaluation		
Model	**ROC-AUC**	**Accuracy**
Depth = 11	**73.79%**	**79.43%**
Depth = 8	**73.82%**	**79.44%**

The lowest error rate the authors of the original research achieved on their training data was 18%, which is equivalent to an accuracy of 82%, which the present study outperformed in training data using cross-validation with our model settings. However, the lowest error rate authors

accomplished in the evaluation data was 16%, which is equivalent to 84% accuracy, and the present as-is model did not outperform this metric. Although this initial peek was promising and achieved better results with more parameter tuning, data cleaning, and class balancing procedures, this was clearly out of the scope of this study and may be left for future research to pursue.

Chapter 7

Conclusions

This chapter concludes the research and provides insight into what could be further worked on in the future.

7.1 Contribution

In this research, the goal was to create a predictive model for the repayment/financial performance of future borrowers who will be taking a benevolent loan from the lender owning the data. This was achieved through a tree-based binary classifying model implemented with the CatBoost library, which could consistently achieve a ROC-AUC of over 85% evidenced by the 5-Fold Cross Validation approach this study used. The secondary goal was to gain some insight into the contributing factors affecting the repayment performance of the borrowers, and this was achieved through the state-of-the-art built-in feature importance capabilities CatBoost, and some other tree-based libraries have. The findings were discreetly shared with the data owner and were not published in the present study at their request. Finally, the tertiary goal of this research was to explore that repayment/financial performance can be predicted in a controlled environment using ONLY non-financial feature space. The present study was successful in achieving strong performances synonymous with other studies available publicly. This approach and algorithm

would only work and achieve good results for problems with the following characteristics: larger datasets, tabular data, and when absolute accuracy is not required. This study's approach and algorithm would not be suitable for medical research, for example, due to the high variability and low bias of the Tress-Based methods.

7.2 Future Work

Based on the results, it seems that the present study's model's performance was capped at the scores that were achieved. This is because some sparsity was found in the data along with several attributes in the feature space that had more unique values than the model can learn from. Based on this, it was recommended to the data owner that they modify the ways they collect data to be more research guided and driven. Should this be achieved later, the research can be reapproached. Based on the data, it seems that better performance for the predictive model is not out of sight or hand.

Furthermore, if the tri-class target in the dataset is going to be of more prominence to the data owner, there is a great opportunity in the n-1 multi-model approach and especially with better quality data. With more time and focus, and by assembling the multi-model using the multi-algorithm approach, better results can be achieved. This approach barely missed the bar of the single model performance. However, it was abandoned due to the need to focus more on the main goal of the binary classifier. If different algorithms are amalgamated together in a multi-model approach, and if they are designed such that each stage is provided with the best algorithm that can predict its task, the overall classifier will find synergy and perform better.

References

[1] Shwartz-Ziv, R., & Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, *81*(Complete), 84–90. https://doi.org/10.1016/j.inffus.2021.11.011

[2] S. Arik, T. Pfister, Tabnet: Attentive interpretable tabular learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, (8) 2021, pp. 6679–6687, URL https://ojs.aaai.org/index.php/AAAI/article/view/16826.

[3] L. Katzir, G. Elidan, R. El-Yaniv, Net-DNF: Effective deep modeling of tabular data, in: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021, OpenReview.net, 2021, URL https://openreview.net/forum?id=73WTGs96kho

[4] S. Popov, S. Morozov, A. Babenko, Neural oblivious decision ensembles for deep learning on tabular data, in: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, OpenReview.net, 2020, URL https://openreview.net/forum?id=r1eiu2VtwH.

[5] Baosenguo, Baosenguo/kaggle-moa-2nd-place-solution, 2021, URL https://github.com/baosenguo/Kaggle-MoA-2nd-Place-Solution

[6] Prokhorenkova L, Gusev G, Vorobev A, Dorogush AV, Gulin A. CatBoost: unbiased boosting with categorical features. Advances in neural information processing systems. 2018;31.

[7] Chen T, Guestrin C. Xgboost: A scalable tree boosting system. InProceedings of the 22nd ACM sigkdd international conference on knowledge discovery and data mining 2016 Aug 13 (pp. 785-794).

[8] Harris, T. (2015). Credit scoring using the clustered support vector machine. *Expert Systems with Applications*, *42*(2), 741-750.

[9] Wang, C., Han, D., Liu, Q., & Luo, S. (2018). A deep learning approach for credit scoring of peer-to-peer lending using attention mechanism LSTM. *IEEE Access*, *7*, 2161-2168.

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

[11] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*.

[12] Li, W., Ding, S., Wang, H., Chen, Y., & Yang, S. (2020). Heterogeneous ensemble learning with feature engineering for default prediction in peer-to-peer lending in China. World Wide Web, 23(1), 23-45.

[13] Addo, P., Guegan, D., & Hassani, B. (2018). Credit Risk Analysis Using Machine and Deep Learning Models. Risks, 6(2), 38. MDPI AG. Retrieved from http://dx.doi.org/10.3390/risks6020038

[14] Chawla, Nitesh V., Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. Smote: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research 16: 321–57.

[15] Yeh, Ivy & Lien, Che-Hui. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. Expert Systems with Applications. 36. 2473-2480. 10.1016/j.eswa.2007.12.020.

[16] Trivedi, S. K. (2020). A study on credit scoring modeling with different feature selection and machine learning approaches. Technology in Society, 63(Complete). https://doi.org/10.1016/j.techsoc.2020.101413

[17] Ampountolas, Nde, T. N., Date, P., & Constantinescu, C. (2021). A machine learning approach for micro-credit scoring. Risks (Basel), 9(3), 50–. https://doi.org/10.3390/risks9030050

[18] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in neural information processing systems, 30*.

[19] Emmons, W. R., & Ricketts, L. R. (2016). The Demographics of Loan Delinquency: Tipping points or tip of the iceberg?. *Center for Household Financial Stability, Federal Reserve Bank of St. Louis (Oct. 2016)*.

[20] Frank, E., & Hall, M. (2001, September). A simple approach to ordinal classification. In *European conference on machine learning* (pp. 145-156). Springer, Berlin, Heidelberg.

Bibliography

[1] Adaboost - Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of online learning and an application to boosting. Journal of computer and system sciences, 55(1), 119-139.

[2] Svm- Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. Statistics and computing, 14(3), 199-222

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

[4] Perceptron - Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. Psychological Review, 65(6), 386.

[5] Keras- Keras.io

[6] Extratrees- Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. Machine learning, 63(1), 3-42.

[7] Random forests original paper- Ho, T. K. (1995, August). Random decision forests. In Proceedings of 3rd international conference on document analysis and recognition (Vol. 1, pp. 278-282). IEEE.

[8] Blog: https://towardsdatascience.com/simple-trick-to-train-an-ordinal-regression-with-any-classifier-6911183d2a3c

[9] J.N. Crook, D.B. Edelman, L.C. Thomas, Recent developments in consumer credit risk assessment, Eur. J. Oper. Res. 183 (3) (2007) 1447–1465.

[10] D.J. Hand, W.E. Henley, Statistical classification methods in consumer credit scoring: a review, J. Roy. Stat. Soc. 160 (3) (1997) 523–541.

[11] B. Baesens, T. Van Gestel, S. Viaene, M. Stepanova, J. Suykens, J. Vanthienen, Benchmarking state-of-the-art classification algorithms for credit scoring, J. Oper. Res. Soc. 54 (6) (2003) 627–635.

[12] I. Partalas, G. Tsoumakas, I. Vlahavas, An ensemble uncertainty aware measure for directed hill-climbing ensemble pruning, Mach. Learn. 81 (3) (2010) 257–282.

[13] Thomas, Lyn, Jonathan Crook, and David Edelman. 2017. Credit Scoring and Its Applications. Philadelphia: SIAM.

[14] Weinberger, Kilian Q., and Lawrence K. Saul. 2009. Distance metric learning for large margin nearest neighbor classification. Journal of Machine Learning Research 10: 207–44.

[15] Jarrow, Robert, and Philip Protter. 2019. Fair microfinance loan rates. International Review of Finance 19: 909–18. [CrossRef]

[16] Brau, James C., and Gary M.Woller. 2004. Microfinance: A comprehensive review of the existing literature. The Journal of Entrepreneurial Finance 9: 1–28.

[17] Demirgüç-Kunt, Asli, Leora Klapper, Dorothe Singer, Saniya Ansar, and Jake Hess. 2020. The global findex database 2017: Measuring financial inclusion and opportunities to expand access to and use of financial services. The World Bank Economic Review 34 (Suppl. 1): S2–S8.

[18] Deep learning generalization: Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. Communications of the ACM, 64(3), 107-115.

[19] Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

[20] Tsai, C. F., Hsu, Y. F., & Yen, D. C. (2014). A comparative study of classifier ensembles for bankruptcy prediction. Applied Soft Computing, 24, 977-984.

[21] Keras Embeddings: https://www.kaggle.com/code/sudalairajkumar/a-look-at-different-embeddings

[22] Bequé, A., & Lessmann, S. (2017). Extreme learning machines for credit scoring: An empirical evaluation. Expert Systems with Applications, 86, 42-53.

[23] Sadatrasoul, S., Gholamian, M., & Shahanaghi, K. (2015). Combination of Feature Selection and Optimized Fuzzy Apriori Rules: The Case of Credit Scoring. International Arab Journal of Information Technology (IAJIT), 12(2).

[24] Nalić, J., Martinović, G., & Žagar, D. (2020). New hybrid data mining model for credit scoring based on feature selection algorithm and ensemble classifiers. Advanced Engineering Informatics, 45, 101130.

[25] ML interpretation: Molnar, C. (2020). Interpretable machine learning. Lulu. com.

Vita Auctoris

Ahmed Shafeek Abouhassan, better known as Ahmed Shafeek, was born in Giza, Egypt. He graduated in 1996 from the Egyptian Air Defense College, which is one of five Egyptian military academic institutions graduating active-duty officers. It is also one of two military institutions providing engineering degrees. While there, Ahmed earned a Bachelor of Electrical Engineering with a major in Communication and a minor in Computers. He enrolled at the University of Windsor in 2019 to earn his second degree, and in 2021, he earned his Bachelor of Computer Science Honours with great distinction after being on the Faculty of Science Dean's Roll of Honor for all the semesters he spent at the university. In 2021, he joined the School of Computer Science and earned his Master of Science in Artificial Intelligence in August 2022.