UTILIZING MACHINE LEARNING TECHNIQUES TO PREDICT CREDIT CARD PAYMENT DEFAULTS

Madison Guerra
madisonguerra@dusty.tamiu.edu

UTILIZING MACHINE LEARNING TECHNIQUES TO PREDICT CREDIT CARD PAYMENT DEFAULTS

A Thesis

by

MADISON RENEE GUERRA

Submitted to Texas A&M International University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

May 2023

Major Subject: Mathematics

UTILIZING MACHINE LEARNING TECHNIQUES TO PREDICT CREDIT CARD PAYMENT DEFAULTS

A Thesis

by

MADISON RENEE GUERRA

Submitted to Texas A&M International University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Chair of Committee,	Saqib Hussain
Committee Members,	Runchang Lin
	Muhammad Mohebujjaman
	Tariq H. Tashtoush
Head of Department,	Rohitha Goonatilake

May 2023

Major Subject: Mathematics

ABSTRACT

Utilizing Machine Learning Techniques to Predict Credit Card Payment Defaults (May 2023)

Madison Renee Guerra, B.A., Texas A&M International University;

Chair of Committee: Saqib Hussain, Ph.D.

The question of accurately predicting credit card defaulters has been explored in numerous studies in the past. In these studies, the researchers utilized various machine learning theories and techniques to make the determination the extent of defaults. Unfortunately, some constraints were encountered, and the limitations that existed from the previous works have been discussed. This project attempted to address these issues with special attention given to more recently available data. Specifically, in this project, we looked at data provided by one Kaggle user, which utilized the data from the American Express credit card competition, which ranges from late March 2018 to late October 2019, approximately 18 months. The extent of credit card defaulters was looked into using the data and used a machine learning technique, called Extremely Randomized Trees. Furthermore, a balancing technique, called Synthetic Minority Oversampling Technique, also known as SMOTE, was used to ensure the classes that were explored and balanced. Finally, the findings from the current research were compared with that of previous findings. The outcome of this project was understanding and analyzing previous research utilizing the updated available data to predict credit card payment defaults more accurately.

ACKNOWLEDGEMENTS

In this section, I would like to take the opportunity to acknowledge and express my sincere appreciation to the individuals and the organization who contributed their feedback, time, and understanding during the completion of this thesis.

Firstly, I would like to give my warmest thanks to my thesis advisor, Dr. Saqib Hussain for agreeing to work with me and made this work possible. His consistent guidance carried me through all the stages of working on this study. I would also like to thank my committee members and the head of the department for agreeing to be present at my defense and providing essential comments and suggestions.

Next, I would like to display my sincerest appreciation to all the users in the Kaggle Community who generously gave their time to answer my questions and assisted me throughout the creation of the programming needed for this work. These users provided me with the opportunity to overcome many obstacles. I would also like to give special gratitude and recognition to Kaggle users, Munum and Kuldip Gusani who provided such an essential hand to my understanding and the creation of the program.

Thirdly, I would also like to offer special thanks to my friends and family as a whole for their continuous support and understanding when undertaking the research and writing the project. Each one of you contributed to my journey as a graduate student in special ways, and I will forever be grateful to you all. Your kind wishes words of encouragement, and prayers for me were what sustained me this far.

Finally, I would like to express special thanks to God. Your guidance is what provided me with the motivation and determination to keep going. You are the one who will allow me to finish this degree program, and I will keep on trusting You for my future endeavors.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

In today's world, the credit card has become a vastly utilized mean of a financial instrument by many individuals. The credit card can help these individuals grow their credit scores, establishing credits, and allowing the consumer access to borrowing to buy a house, or a vehicle. However, with poor financial decisions, an individual's account can be flagged as a delinquent account by the credit card issuer, which then causes the user to default. The reason why the credit card issuer may declare an individual's account as delinquent is that the individual may have misused their credit card due to nonpayment and delays. For example, the customer may have spent more than they can afford to pay back in a timely manner, which may cause the user to not be able to pay the minimum amount required by the designated deadline. At times, an individual may not pay any amount, and after a specific number of days, the credit card issuer must take action, as they are dealing with a financial loss. Eventually, the credit card issuer will close the account completely, which causes the issuer to lose the amount of money they lent to the individual.

The likelihood of an individual defaulting on their credit card payments may depend on specific features, such as an individual's age, extent of their education, gender, and marital status. However, an individual with features that are considered "high risk" is not simply considered a "blueprint" for all borrowers. Thus, it is imperative to understand that the likelihood of an individual resulting in a credit card payment default is not strictly dependent on those features. Although credit card payment defaults are more common in individuals with specific features, it is essential to note that any customer can default on their credit card payments if they become irresponsible with their payments (i.e., not paying the minimum amount that they owe each month), regardless of these features. Thus, various machine learning techniques, such as the Random Forest (RF) and Extremely Randomized Trees (ERT) methods, can be used to most accurately predict the rate of an individual defaulting. However, in machine learning algorithms, the main factor considered is the habit of an individual's monthly payment over a series of a certain number of days.

Previously, researchers completed work comparing multiple machine learning techniques to predict credit card payment defaults accurately. Their research showed that

This thesis follows the model of *Latex*.

many machine learning techniques each have their own accuracy and precision scores. Although numerous methods have similar accuracy rates, there are a couple of machine learning algorithms that stand out as being the most accurate technique to predict credit card payment defaults when accessing previous data.

When accessing data for machine learning, it is important to acknowledge the risk factors that come along with it, which are Loss Event Frequency (LEF) and Loss Magnitude (LM), according to The Factor Analysis of Information Risk (FAIR) Institute (n.d.) [7]. FAIR provides a quantitative risk model that displays what the risk is, how it works, and how to quantify it, as in Figure 1.1.

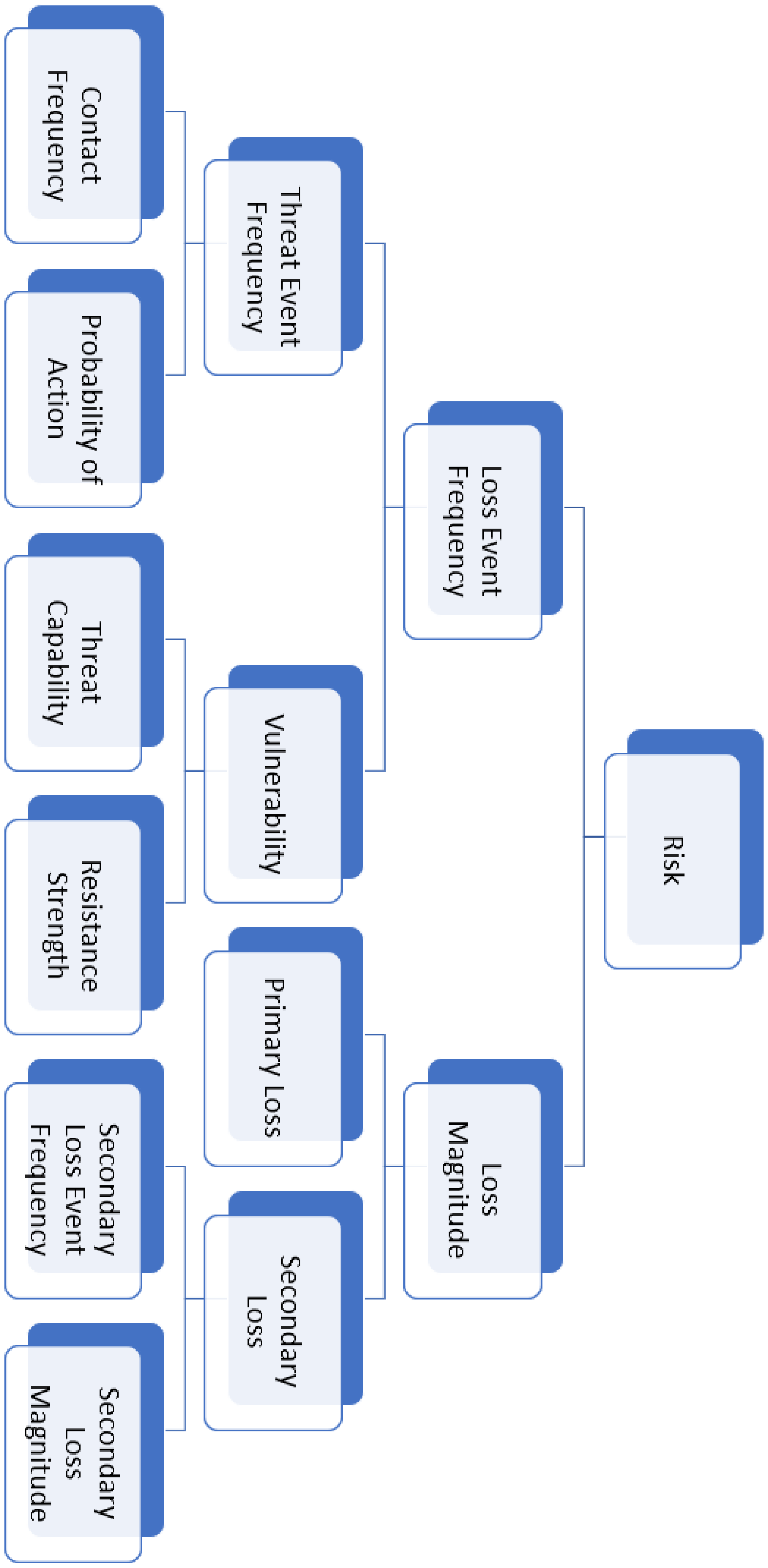


Figure 1.1: Illustration of The FAIR Model.

According to the FAIR Institute (n.d.), the LEF risk measures, in a given timeframe, the probable frequency that a threat agent action will result in loss [7]. Within the LEF risk, there are two additional risks: Threat Event Frequency and Vulnerability. The Threat Event Frequency is defined as very similar to LEF; the only difference between these two threats is whether the agent's action was successful. For instance, an agent could attack a web server and be unsuccessful. Thus, this unsuccessful attempt is considered a Threat Event Frequency, as there was no loss. Meanwhile, Vulnerability measures the probability of a threat event can be turning into a loss event.

Within Threat Event Frequency, there are two other risk analysis terms: Contact Frequency and Probability of Action [7]. Contact Frequency is defined as an attacker getting into contact with the asset(s), but there was no further action taken. Probability of Action is defined as taking into account realistic risks and calculating how often they can occur. It allows you to focus on relevant risk scenarios for analysis.

Within Vulnerability, the two other risk analysis terms are Threat Capability and Resistance Strength [7]. Threat Capability is defined as the probable force the threat agent is capable of applying against an asset. Meanwhile, Resistance Strength is defined as, when compared to a baseline unit of force, the strength of control.
[7] On the right side of the model, there is the risk of Loss Magnitude. According to the FAIR Institute (n.d.), the LM risk is the probable magnitude of primary and secondary loss resulting from an event. Within the LM risk, there are two additional risks: Primary Loss and Secondary Risk. The Primary Loss risk is defined as the losses incurred from the loss event itself and the results of the threat agent's successful action. Meanwhile, the Secondary Risk is defined as the losses incurred from secondary stakeholders or outside parties. As a result, the FAIR model displays a foundation for analyzing, understanding, and calculating risk.

In conclusion, this research project discusses an overview of various factors that contribute to the rate of credit card payment defaults and narrows down the specific factors associated with an individual that is most likely to default. This project then continues to discuss the different machine learning techniques used in previous works and based on their research, this project utilizes the machine learning algorithm that is found to be the most accurate, which is the Extremely Randomized Trees method, also known as Extra

Trees. This machine learning technique was utilized in order to predict credit card payment defaults using the more recently available data. Since this project will be utilizing and accessing data, this work also addresses risk factors found in the FAIR model.

2. MATHEMATICAL TOOLS

This section provides some important definitions of various terms used throughout the project.

1. Machine Learning: Machine learning is a branch of artificial intelligence, also known as AI, and computer science, and focuses on the use of data and algorithms. Machine learning has been gradually improving its accuracy when imitating the way that humans learn.

2. True Positive: The True Positive value, denoted as $TRUE_{positive}$, is an outcome where the machine learning algorithm predicts the positive class correctly.

3. True Negative: The True Negative value, denoted as $TRUE_{negative}$, is an outcome where the machine learning algorithm predicts the negative class correctly.

4. False Positive: The False Positive value, denoted as $FALSE_{positive}$, is an outcome where the machine learning algorithm predicts the positive class incorrectly.

5. False Negative: The False Negative value, denoted as $FALSE_{negative}$, is an outcome where the machine learning algorithm predicts the negative class incorrectly.

6. Accuracy: Accuracy measures the correct classification obtained by the classifier.

$$Accuracy = \frac{TRUE_{positive}+TRUE_{negative}}{TRUE_{positive}+TRUE_{negative}+FALSE_{positive}+FALSE_{negative}}$$

7. Default: Default occurs when an individual fails to make a payment on a debt by the due date.

8. Precision: Precision is the division of the number of true positives by the total number of positive predictions.

$$Precision = \frac{TRUE_{positive}}{TRUE_{positive}+FALSE_{positive}}$$

9. k-Nearest Neighbor: The k-Nearest Neighbor, which is also known as KNN or k-NN, is a classifier that uses the closest training examples to make classifications or predictions about the grouping of an individual data point.

10. Recall Score: Out of all the actual correct classifications, the recall score measures the accuracy of correct predictions.

$$Recall\ Score = \frac{TRUE_{positive}}{TRUE_{positive}+FALSE_{negative}}$$

11. F1-Score: The F1-Score is classified by a range in values between 0 and 1, where 1 is the best. It is the harmonic mean of precision and recall values of the classifier.
12. Cross Validation: The Cross Validation technique is used to evaluate and assess the effectiveness of the model.

3. LITERATURE REVIEW

This section discusses the most significant factors associated with used to predict credit card payment defaulters based on prior research. Additionally, this section is devoted to discussing the various works referenced during the creation of the program that predicts the individuals who defaulted on their credit card payments.

3.1 Credit Card Default Factors

A preliminary assessment of the literature reveals that previous research has shown multiple and various features considered when predicting default risks. Dominguez identified that a person's credit limit, sex, education, marital status, age, history of past payments, amount of bill statements in the past 6 months, and amount of previous payments for the past 6 months are significant features when predicting default [3]. The features identified in this study are imperative because they allow researchers to compare the differences between people, such as a higher credit limit and a lower credit limit, individuals' sex, and education, to narrow the factors that are significant predictors of default. Identifying each person's contrasts and features will permit credit card companies to more easily deny persons with a higher default risk prediction based on specific factors. Meanwhile, another study further suggests that when predicting payment defaults, the most significant features are gender, education, age, marital status, and limit balance, which is also known as the amount of credit given [10]. Additionally, other significant predictors of credit card payment defaults are the times-payment-delayed, which is derived from the repayment status, and the average-payment-amount, which is derived from the payment amount [10].

3.1.1 Limit-Balance, Education, and Age

Using Pearson correlation, research found that the "limit- balance and education [features] were negatively correlated" with payment defaults [10]. An individual with a higher limit balance and higher education has a less likely chance of payment defaults and vice versa; an individual with a lower limit balance and lower education has a higher chance of payment defaults.

On the other hand, age was found to be positively correlated with payment default

[10]. As an individual increase with age, the possibility of payment default increases as well. Therefore, older individuals, with little education, and lower credit limits are more likely to default compared to younger people, with higher education and higher credit limits.

3.1.2 Dependence of Gender

Furthermore, since gender is considered a significant factor when predicting payment default, studies have shown the likelihood of missed payments by men and women. Li (2018) observed that women typically have lower credit scores due to having more difficulty in repaying [11]. Unfortunately, Li (2018) mentioned that there were certain limitations in the research, as there were more non-white, single women under the age of 30 with higher education than men. As previously stated, an individual's marital status, age, and education significantly predict payment defaults. Thus, Li (2018) concludes that specifically single women have somewhat lower credit scores than single men with comparable demographic characteristics.

More recently, Dunn and Mirzaie (2022) found that women tend to have greater difficulties repaying their debt and greater revolving credit utilization rates [4]. Furthermore, their research shows that women tend to exhibit about 30% more debt stress scores compared to men.

Therefore, according to Li (2018) and Dunn and Mirzaie (2022), as women are more likely to carry balances each month, not pay by the due date, and have a greater debt stress score, women are more likely to have payment defaults.

3.1.3 Marital Status

Next, as discussed previously, marital status is another substantial feature when predicting credit card payment default. According to Stolba (2020), married couples typically have more than doubled the amount of debt compared to single individuals [15]. However, single individuals are more likely to have delinquency accounts. The reason for this is that married couples usually pay their credit card bills on time, which ultimately prevents their account from being recorded as delinquent. Consequently, unmarried individuals are statistically more likely to have credit card payment defaults.

3.1.4 Amount-Paid-Back and Payment-Status

The factors, amount-paid-back, and payment-status, were found to be negatively correlated with defaults [10]. This means that an individual with a higher chance of paying back what they owe was less likely to default on their credit card payments. Additionally, a person's payment status affected their probability of default, as an individual who has issues with their payment status (i.e., pending transactions and rejected transactions), the more likely they were to default. Jain and Jayabalan (2022) continue to state that this was an expected outcome because, as someone who has a greater ability to pay ultimately reduces their probability of defaulting [10].

In conclusion, these factors are considered significant when predicting credit card payment defaults. Credit card users must typically pay their bills on time to avoid late fees and accounts being delinquent. However, due to various circumstances, many individuals cannot pay their debts accordingly, which is more common in older, single women, with less education, on average, and with a smaller credit card limit balance. Additionally, the individual's means of paying back directly affected their possibility of defaulting. Although the previously listed factors are the most common features in credit card customers, it is not the case for every individual with these specific features. Similarly, although other individuals without the characteristics are less likely to default, it does not mean that they will not. Thus, it is imperative to use various machine learning techniques to predict credit card payment defaults and identify which technique is the most accurate.

3.2 Machine Learning Techniques Used to Predict Credit Card Payment Defaults

Earlier research has explored different machine learning techniques to predict an individual's rate resulting in a credit card payment default event. Additionally, previous research has identified the accuracy rates for each technique they used and ultimately showed which technique, specifically for predicting default rates, had the best precision.

3.2.1 Previous Research from Literature

To begin, Dominguez (2021) examined three various machine learning techniques utilizing the data set from the University of Carolina (UC) Irvine's Machine Learning

Repository [3]. The features that Dominguez (2021) considered within the research are 1) Credit Limit, 2) Sex, 3) Education, 4) Marital Status, 5) Age, 6) History of past payments, 7) Amount of bill statements for the past 6 months, and 8) Amount of previous payments for the past 6 months. Dominguez (2021) found that the three largest credit limits are $50,000, $20,000, and $30,000 respectively. Additionally, the data used was distributed evenly between males and females. Finally, the data mostly consisted of couples with individuals in their mid-30s to mid-40s. Meanwhile, the single individuals in the dataset were in their mid-20s to mid-30s.

Next, Dominguez (2021) uses the dataset to compare various machine learning techniques, which are the k-Nearest Neighbors, Logistic Regression, Radio Frequency, XGBoost, and Support Vector Machine. Although each previously listed technique had similar accuracy rates, it was observed that the Support Vector Machine (SVM) had the highest accuracy rate at 0.82. As Ullah et al. (2018) mention, the accuracy rates between these machine learning techniques seem good, it can be misleading and cannot be easily interpreted [17]. Thus, after discovering the accuracy of SVM, Dominguez continued to compare algorithms and used the Random Forest technique as it was comparable to the SVM technique in accuracy [3]. Additionally, the Random Forest technique was less computationally expensive. Essentially, the Random Forest technique groups a generated series of bootstrapped trees and ultimately predict the results by combining the outcome across all the trees [14]. Dominguez (2021) then chose to optimize the recall score of the Random Forest technique. Using the optimization formula, the Random Forest method resulted in a recall score of 0.95.

Unfortunately, Dominguez (2021) noted that the dataset was imbalanced, as there was not an equal number of examples for each class. Thus, their research utilizes Random Undersampling and Random Oversampling techniques. Both techniques essentially distribute the data evenly. According to Alam et al. (2020), the Random Undersampling technique randomly eliminates the majority class instances in the training set until the ratio between both the minority and majority classes is at the desired level [1]. Whereas, the Random Oversampling technique randomly replicates the minority class in the training data. Dominguez (2021) identified that both the Random Undersampling and Random Oversampling had a recall score of 0.79.

As a consequence, Dominguez (2021) checked for overfitting, which means that the model was weak at generalizing, especially unseen data, but was strong at predicting the data that was trained [3]. He analyzed the recall scores of each method and concluded that each of the machine learning techniques was performing similarly due to the validation score and the test score being the same. Thus, overfitting did not occur.

Since each technique performed similarly when generalizing to unseen data, we know that the Random Forest had the highest accuracy rate. Thus, the Random Forest can be used in future research, especially to compare other, highly accurate machine learning methods. Additionally, as the work of Dominguez (2021) used data from the University of California Irvine's Machine Learning Repository from 2005, it is essential to compare various machine learning techniques with more up-to-date data.

Further research suggests that there are other highly accurate machine learning techniques that predict credit card payment defaults with more recent data. For instance, Jain and Jayabalan (2022) first found that linear discriminant analysis (LDA) with stacked sparse autoencoder (SSAE) together had the best results, as the accuracy rate was at 90.00% [10]. Additionally, the LDA-SSAE had an F1-score of 90.00%. However, Jain and Jayabalan (2022) also observed that the LDA accuracy rate was 81.00% without SSAE. Furthermore, the LDA, without SSAE, had an F1-score of 78.00%. Thus, for a higher accuracy rate, it is imperative to combine both the LDA and SSAE machine learning techniques when predicting credit card payment defaults.

Jain and Jayabalan (2022) continued their research and compared the accuracy rates with another machine learning technique: Extremely Randomized Trees. They identified that the Extremely Randomized Trees (ERT) machine learning algorithm had the best performance, as it resulted in an accuracy rate of 95.84%, a precision of 94.87%, recall of 85.85%, and an F-score of 90.14 [10].Unfortunately, the models learned patterns that date fifteen years prior to their research. Thus, the predictions may not be able to predict patterns that may have changed over time accurately. Their work suggests that any future research should be performed using the most recent data to ensure the models remain relevant in predicting defaulters, especially with high precision and efficiency [10].

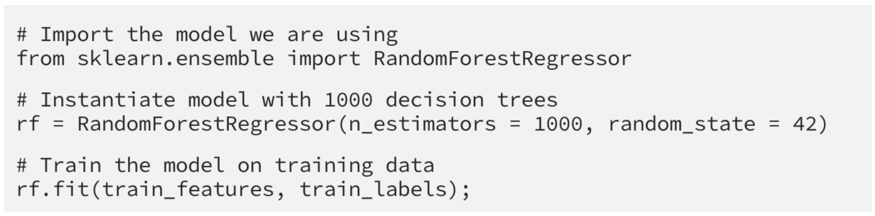
Based on the previous work mentioned above, it appears that both the Random Forest and the Extremely Randomized Trees methods were the most accurate when predicting credit card payment defaults. However, it is important to note the differences between the previous research. For instance, Dominguez (2021) utilized data from over 15 years ago; meanwhile, Jain and Jayabalan (2022) incorporated more recent data. Thus, it is essential to take into account that Random Forest may still be more accurate when using more recent data, as neither of the previous works used both of the machine learning techniques in their research.

3.3 Comparison of Random Forest and Extremely Randomized Trees

As mentioned before, previous works have shown that both the Random Forest and Extremely Randomized Trees methods were the most accurate in predicting credit card payment defaults. Although one notable difference between the research is the datasets they utilized, it is essential to identify which methods would be considered best when predicting credit card payment defaults.

3.3.1 Random Forest

The Random Forest machine learning technique was first developed in 2001 by Leo Breiman [2]. Breiman utilizes the random selection of features machine learning technique, established in 1995, and the bagging sampling approach, established in 1996, to construct a collection of decision trees with controlled variation. This method contains numerous decision trees that produces its prediction based on the average of each trees' prediction.To make a prediction, the random forest algorithm spits out a class prediction, and the class with the most votes becomes the model's prediction.



```
# Import the model we are using
from sklearn.ensemble import RandomForestRegressor

# Instantiate model with 1000 decision trees
rf = RandomForestRegressor(n_estimators = 1000, random_state = 42)

# Train the model on training data
rf.fit(train_features, train_labels);
```

Figure 3.1: Illustration of an example program using the Random Forest technique.

3.3.2 Extremely Randomized Trees (ERT).

The Extremely Randomized Trees (ERT) machine learning technique was first developed in 2006 by Guerts, et al. [5]. As cited in Islam et al. (2018), the Extremely Randomized Trees algorithm is also known as the Extra Trees technique. The ERT technique is described to be an ensemble machine learning algorithm based on decision trees. This algorithm composes a large number of decision trees, where the final decision is obtained taking into account the prediction of every tree as shown in Figure 3.2.

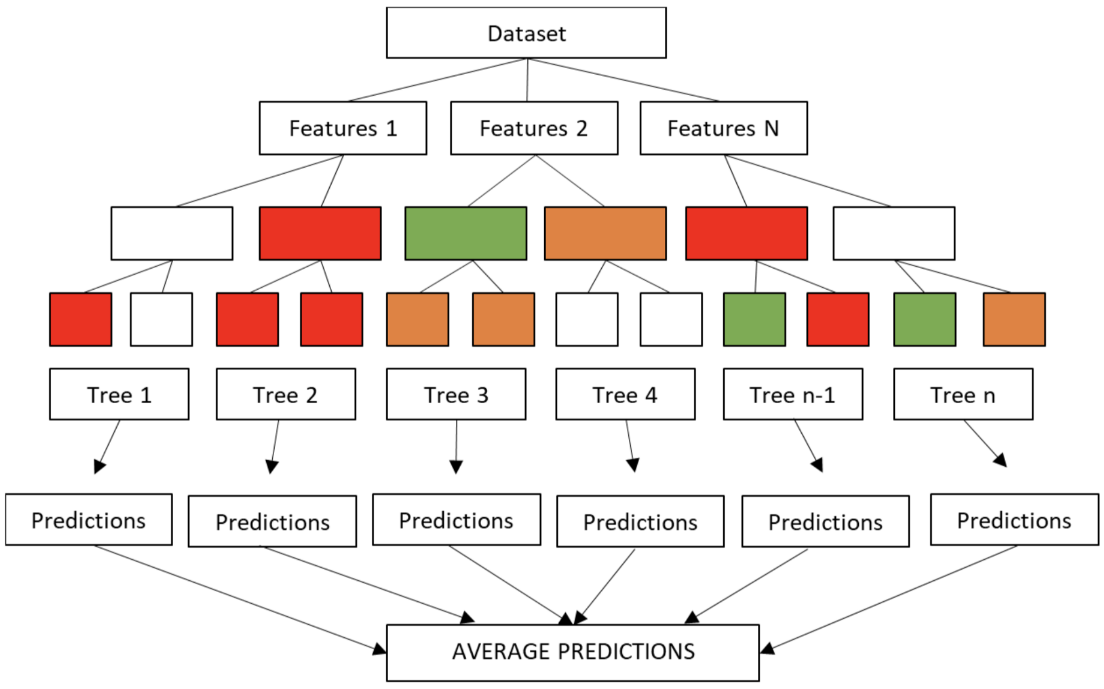


Figure 3.2: Illustration of an example of the Extra Trees Classifer.

Going back to the previous research, it is important to note that although Jain and Jayabalan (2022) discuss the accuracy of the ERT technique, the results were concluded by Islam et al. (2018), as they found that the Extremely Randomized Trees outperform other algorithms in terms of accuracy, precision, recall, and F1-score [8]. Unfortunately, the data used by Islam et al. (2018) originated from 2005 by UC Irvine's Machine Learning Repository, similar to the work of Dominguez (2021).

As both researchers relied on the dataset from 2005, it is important to note that when comparing the research of Islam et al. (2018) and Dominguez (2021), there are multiple machine learning techniques being compared. To reiterate, Dominguez (2021) concluded that the Random Forest method was the most accurate machine learning technique, as the recall score was 0.95 based on his research. Meanwhile, with the same data information, Islam et al. (2018) concluded that the most accurate machine learning technique was the Extremely Randomized Trees method, as the accuracy rate was determined to be 95.84% [8].

Nevertheless, as stated previously, both researchers are using outdated information, as the dataset consists of information from 2005, approximately 17 years ago. As Jain and Jayabalan (2022) suggested, it is imperative to continue performing research on up-to-date data sets to predict credit card payment defaults with high accuracy.

3.3.3 Limitations of Previous Works

Unfortunately, not only is outdated data a limitation of previous works, but there were also other limitations found within the research that needs to be addressed and improved for future studies.

For instance, one of the limitations mentioned in Jain and Jayabalan's (2022) work included the time frame of data from the credit card customers [10]. They observed that in order to have the models generalize the predictions more accurately, they needed at least a full year of data from each credit card customer. Thus, for improvements, future works should utilize this suggestion and use a minimum of one year of data when predicting credit card payment defaults.

Additionally, Jain and Jayabalan (2022) further suggest that future works should implement balancing techniques, such as the Synthetic Minority Oversampling Technique (SMOTE) or Adaptive Synthetic (ADASYN), to ensure that the data points between the class labels are balanced [10].

Finally, Jain and Jayabalan (2022) state that the models from their research learned patterns that date back almost two decades ago. Thus, they suggest future research to utilize more recent data in order to predict defaulters effectively and allow the models to remain relevant [10].

In conclusion, previous research was able to identify machine learning techniques with an accuracy rate of around 95%. However, the data used within these techniques were outdated. Therefore, it is imperative to conduct research with more up-to-date data.

3.4 Problem Statement

The credit card has grown to become a substantial way for individuals to borrow and spend money wisely in today's economy. Unfortunately, many people may miss payment deadlines and borrow more funds than they can afford to pay back, which puts credit card companies at a huge financial risk. To prevent such a risk, it is anticipated to accurately identify individuals who are prone to default.

4. METHODOLOGY

This section includes an overall description of the research design, measures, and risk. It also discusses the method used for this project and states how the data was collected, analyzed, and make appropriate conclusions.

4.1 Research Design

The research design of this study is a case study. This study reviewed various types of previous research on machine learning methods, credit card default factors, and credit card default predictions using machine learning methods. Based on this understanding, a case study approach was developed to analyze this real-world situation, predict credit card defaults using available data, understand past research efforts, and address previous issues in prior research. Based on the extent of research, one of the machine learning methods previously mentioned was the most accurate when used to predict credit card defaults. In this study, research on the aforementioned machine learning method was conducted to ensure an understanding of the program. Finally, a conceptual framework was created to provide solutions to past problems regarding the prediction of credit card default using machine learning methods. This study was conducted between September 2022 and April 2023.

4.2 Methodological Approach

This study aimed to analyze available data and create a program utilizing the Extremely Randomized Trees classifier to predict customers likely to default. This study relied on previous research, which used multiple machine learning algorithms when predicting credit card payment defaults. Unfortunately, prior research expressed multiple limitations, such as accessing outdated data and not ensuring the data was balanced. Thus, to continue with previous claims that display the Extremely Randomized Trees approach as the most accurate, this study ensured to utilize more recent data, balance the dataset evenly, as well as share the findings and compare them with other work that utilized other machine learning algorithms.

4.3 Measures

The data that was used throughout this study was provided by the computing-based software utilized throughout this project, Kaggle, and was in the form of a feather file. The dataset consisted of multiple American Express credit card customers and their profile features at each statement date. Each customer is identified using a specific identification number, and the features of each customer are categorized with various variables, which are D_*, S_*, P_*, B_*, and R_*, where the variables are identified as delinquency, spend, payment, balance, and risk, respectively.

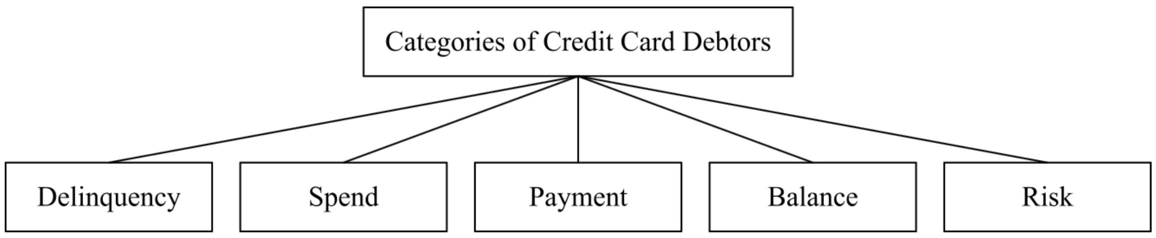


Figure 4.1: Framework illustrating the categories of credit card debtors.

After accessing the AMEX dataset, the Kaggle user then converted the data into a feather file. The original dataset was in the format of a comma-separated value (CSV) file. When conducting this research, it is essential to access and utilize the dataset as a feather file due to various reasons, discussed in a later section.

4.3.1 Risk Factors - The FAIR Model

When accessing and analyzing data regarding credit card payments, it is essential to identify and quantify potential risks using the FAIR model. To begin, we identify the assets, which are the credit card payment data. As stated previously, the data will contain various features of each customer, including their spending, payments, and balances. Due to the information being readily available to the public, threat agents could take actions to gain unauthorized access or infect the system with malware, which ultimately could impact the overall confidentiality of the data. Not only that but depending on the actions of the threat agent, the credit card company could face financial losses, reputable damages, or legal liabilities. Fortunately, the data is encrypted regarding customers to safeguard against any online or cyber vulnerabilities. Each customer is labeled with a customer ID number, which can potentially reduce the risk of data breaches. However, it is significant to find more ways to minimize and avoid unauthorized access.

In conclusion, it is essential to address any risk factors that may come when working with data. Additionally, it is imperative to identify the potential risk factors and its impacts. Finally, it is crucial to reduce to potential risks and prevent data breaches.

4.3.2 Comparison of a CSV File and a Feather File

Now, going back to the reasons why it was essential to use the feather file of the dataset. To begin, the comma-separated value files are known to be a very common flat-file. Essentially, CSV files only contains numbers and/or letters. Additionally, the information found within the CSV files structures the data in the form of a table.

```
import csv

with open('employee_birthday.txt') as csv_file:
    csv_reader = csv.reader(csv_file, delimiter=',')
```

Figure 4.2: Illustration of an example program reading a CSV file.

Meanwhile, feather files are known to be relatively fast at reading data.Feather files aim to push data frames in and out of memory as simple as possible and as quickly as possible, as it is known for its speed.

```
# Result is pandas.DataFrame
read_df = feather.read_feather('/path/to/file')

# Result is pyarrow.Table
read_arrow = feather.read_table('/path/to/file')
```

Figure 4.3: Illustration of an example program reading a feather file.

While feather files being extremely faster than CSV file containing the same data is a notable difference, it is essential to note other significant differences between the files. For instance, CSV files are known to be costly in space and read/write time. Since feather files are lightweight, it takes up less than half of the space compared to CSV files using the same data. Due to a few limitations in this study that will later be discussed, it became obvious and essential to utilize the Kaggle user's dataset since they converted the CSV file into a feather file.

4.3.3 Machine Learning Methods

Based on previous research, the most accurate machine learning method is Extremely Randomized Trees (ERT), as it has over a 95.00% accuracy rate in predicting credit card payment defaults [8]. Naturally, in this project, ERT was the technique used.

4.4 Objectives

The long-term objective of this research was to predict credit card payment defaults using the most accurate machine learning technique. The sub-objectives were as follows:

1. Understand the data provided by the Kaggle user, Munum, which converted the data from the American Express - Default Prediction competition from a CSV file to a feather file;
2. Review different machine learning techniques for better understanding and better performing of the application utilized;
3. Apply the dataset information into the machine learning algorithm that was considered the most accurate based on prior research; and
4. Analyze the outputs of the data and provide findings and understandings of the outputs.

5. PROCEDURE

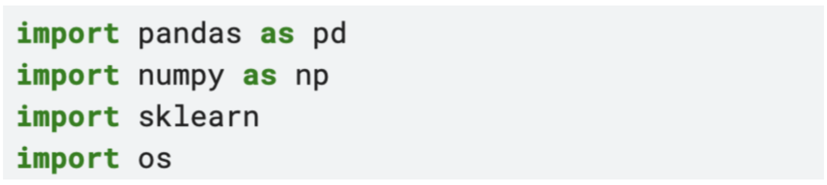
This section discusses which machine learning language was used and the process of this project.

5.1 Machine Learning Language: Python

This project utilized the machine learning language, Python. Python is a high-level general purpose programming language and was developed in the 1970’s by Guido van Rossum. It can be found in various applications, as Python is a universal programming language. Overall, this programming language is an interpreted, interactive, object-oriented, high-level programming language with dynamic semantics. Essentially, this means that Python is a flexible programming language that allows the user to write and create intricate codes to adapt to various situations. Additionally, the Python language allows the user to view the results of their code immediately after typing in the commands, as well as compile the code before running the program. Finally, the Python language was designed to be easily readable and understandable. Thus, this project utilized the Python language.

5.2 Importing Libraries

Before creating the program, it is essential to import specific libraries in order to ensure a smooth and easy process. The first imported libraries in this project are shown below in Figure 5.1.



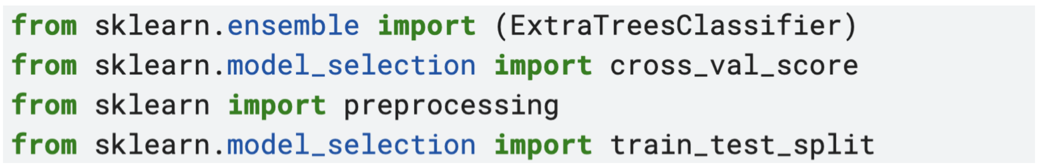
```
import pandas as pd
import numpy as np
import sklearn
import os
```

Figure 5.1: Illustration of the program importing some important libraries.

The “import pandas as pd” code is known for being a popular open source library used for various functions, such as handling and analyzing data. The allows the programmer to access various functions, such as reading files. For example, in this project, this library was utilized to access and read the data.

Next, the "import numpy as np" code is known to perform various mathematical operations on arrays. For instance, in this project, this library was utilized to transpose the array.

The "import sklearn" is a command to import the Scikit-learn library, which can be used to build models for regression, classification, and prediction. In this project, this library was used to import various models, as shown below in Figure 5.2.

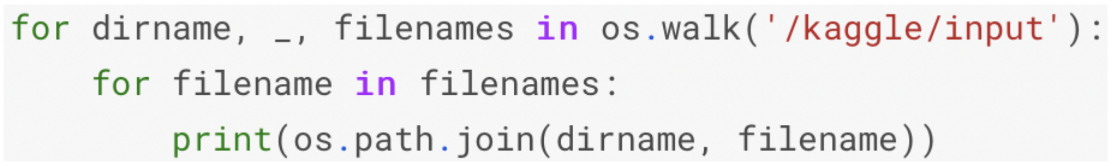


```
from sklearn.ensemble import (ExtraTreesClassifier)
from sklearn.model_selection import cross_val_score
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
```

Figure 5.2: Illustration of the program utilizing "import sklearn" library.

As displayed in the illustration above, the "import sklearn" library command was used to import the machine learning algorithm that was utilized to predict the credit card payment defaults, Extremely Randomized Trees, also known as the Extra Trees. Additionally, this library command was utilized to import the classifier to check the cross validation score. Next, "import sklean" library command was used to import preprocessing, which was used to utilize the label encoder, where the program can understand word labels. Finally, the aforementioned library command was used to import the train_test_split command to split the datasets based on the command. In this project, the datasets were split into 80% train data and 20% test data; the reason for this action will be explained in detail in a later section.

To continue, the "import os" library is used to interact with the underlying operating systems, specifically, the one that the Python program is running on. In this project, the library was utilized to use files directly from Kaggle, shown below in Figure 5.3.

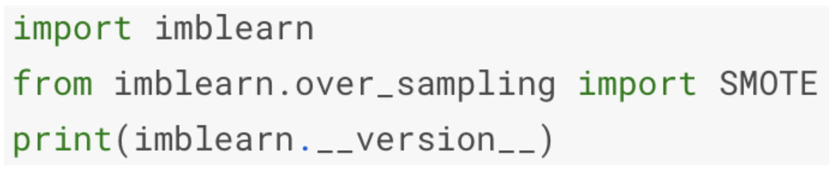


```
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

Figure 5.3: Illustration of utilizing the "import os" library.

As shown in the illustration above, this project utilizes the "import os" to call and use files directly from the Kaggle cloud-based computing software.

Next, this project uses the command library, "import imblearn" in order to utilize one of the data balancing techniques; in this case, SMOTE was imported, as shown below in Figure 5.4.

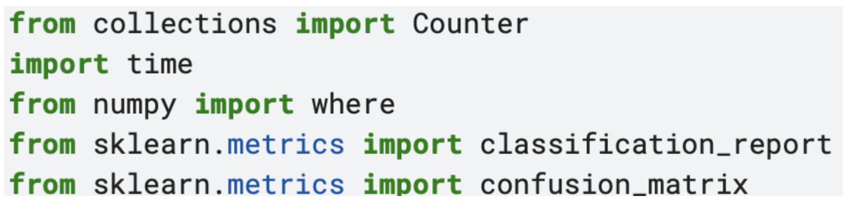


```
import imblearn
from imblearn.over_sampling import SMOTE
print(imblearn.__version__)
```

Figure 5.4: Illustration of utilizing the "import imblearn" library.

The data balancing technique, SMOTE was used to balance the data in this project; the technique is explained in a later section. Meanwhile, the third line of the illustration allows the program to output the version of the imblearn library command. This project uses imblearn version 0.10.1.

Finally, the remaining miscellaneous, yet imperative libraries are displayed below in Figure 5.5.



```
from collections import Counter
import time
from numpy import where
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
```

Figure 5.5: Illustration of utilizing the rest of the library commands.

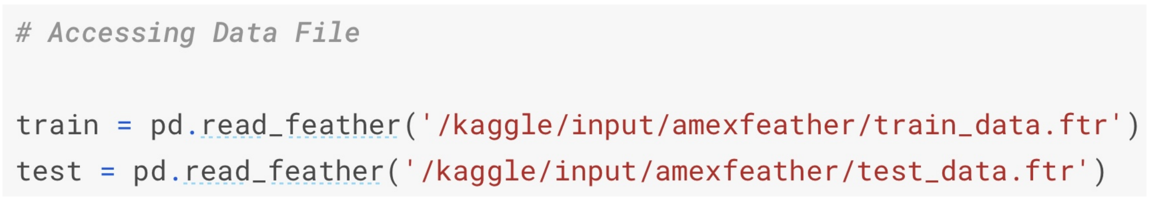
To begin, the "from collections import counter" library command was utilized to count the class distribution before and after using the data balancing technique, SMOTE, to display that the data was in fact imbalanced, as well as balanced after using SMOTE. Next, the "import time" library command was utilized to display how long techniques took to run. For instance, in this program, the SMOTE technique took a total of 8 minutes and 34 seconds to process. Meanwhile, the program to check the cross validation score took approximately 29 minutes and 9 seconds to process.

To continue, the "from numpy import where" library command is used to find the target labels. Then, the "from sklearn.metrics import classification_report" library command was used to display the performance of the Extremely Randomized Trees algorithm. Finally, the "from sklearn.metric import confusion_matrix" was used to display the performance of the algorithm in a matrix form.

In conclusion, various library commands were imported in this project, and each of them had a significant purpose when developing the program to predict credit card payment defaults.

5.3 Creating the Program

In order to accurately predict credit card payment defaults, access to data from various customers with unique features was needed. The data provided by the Kaggle user was collected from the American Express credit card company. To recall, the Kaggle user's dataset converted the Kaggle's American Express - Default Prediction competition dataset into a feather file. Munum's dataset can be accessed through the link: https://www.kaggle.com/datasets/munumbutt/amexfeather.

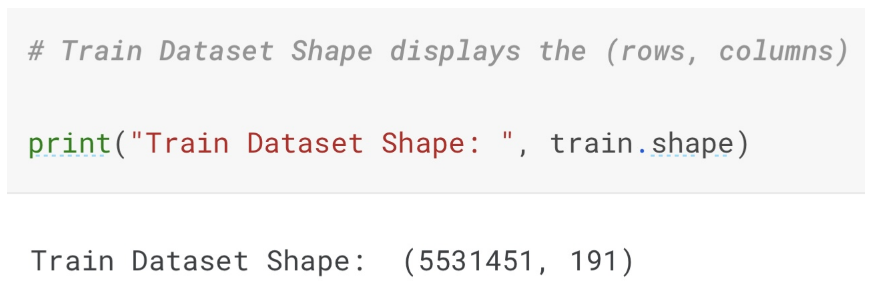


```
# Accessing Data File

train = pd.read_feather('/kaggle/input/amexfeather/train_data.ftr')
test = pd.read_feather('/kaggle/input/amexfeather/test_data.ftr')
```

Figure 5.6: Illustration of the code used to access the train data and test data.

The data provided observed an 18-month performance window between March 2018 and October 2019. Additionally, the data contained categorical variables: 'B_30', 'B_38', 'D_114', 'D_116', 'D_117', 'D_120', 'D_126', 'D_63', 'D_64', 'D_66', 'D_68'. A customer from the American Express credit card is considered a default event when they do not pay their due amount within 120 days from their latest statement date. After evaluating the data set, the likelihood of a customer defaulting on their credit cards using one of the most accurate machine learning techniques will be analyzed for prediction. According to Jain and Jayabalan (2022), the most precise machine learning technique when predicting credit card payment defaults is Extremely Randomized Trees (ERT) [10].

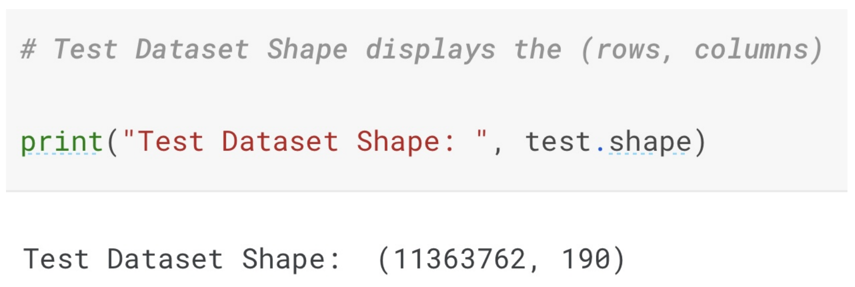
However, before working with the Extremely Randomized Trees technique, the program will need to minimize the feather files as much as possible to ensure sufficient memory throughout the program, as the files appear to be very large.



```
# Train Dataset Shape displays the (rows, columns)

print("Train Dataset Shape: ", train.shape)
```

```
Train Dataset Shape:  (5531451, 191)
```

Figure 5.7: Illustration of the code used to display the shape of the train data.



```
# Test Dataset Shape displays the (rows, columns)

print("Test Dataset Shape: ", test.shape)
```

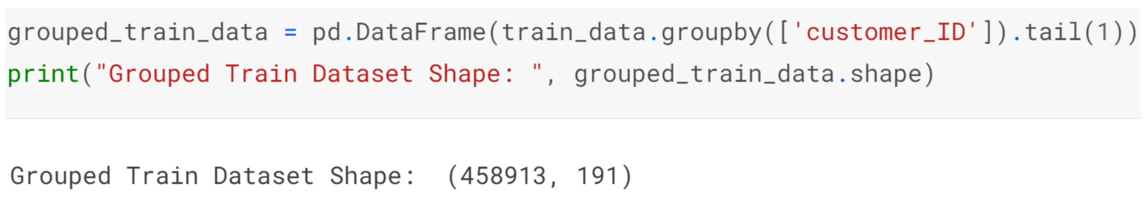
```
Test Dataset Shape:  (11363762, 190)
```

Figure 5.8: Illustration of the code used to display the shape of the test data.

As in Figures 5.7 and 5.8, there are over 5 million rows in the train data and over 11 million rows in the test data. Unfortunately, it seems that minimizing the datasets will be essential (information of the analysis will be elaborated on in later sections).

5.4 Reducing the Data

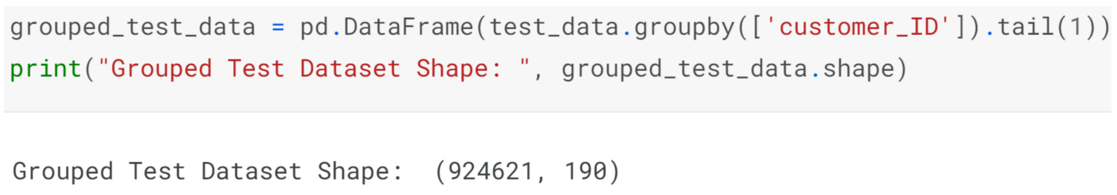
After reading the data, duplicates of customer_IDs were observed. Thus, to first begin minimizing the data, duplicate customer_IDs were grouped.



```
grouped_train_data = pd.DataFrame(train_data.groupby(['customer_ID']).tail(1))
print("Grouped Train Dataset Shape: ", grouped_train_data.shape)
```

```
Grouped Train Dataset Shape:  (458913, 191)
```

Figure 5.9: Illustration of the code used to group duplicate customer_IDs in the train data.



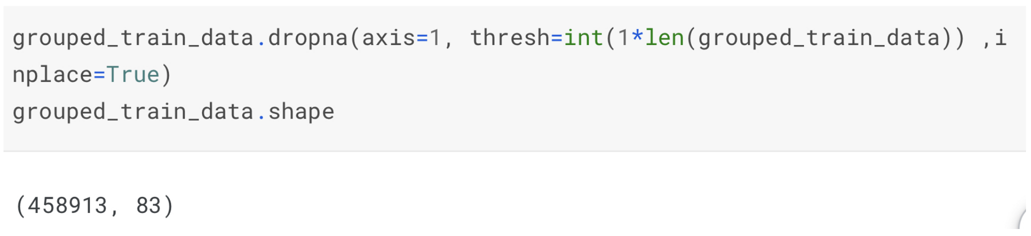
```
grouped_test_data = pd.DataFrame(test_data.groupby(['customer_ID']).tail(1))
print("Grouped Test Dataset Shape: ", grouped_test_data.shape)
```

```
Grouped Test Dataset Shape:  (924621, 190)
```

Figure 5.10: Illustration of the code used to group duplicate customer_IDs in the test data.

As the illustrations show in Figures 5.9 and 5.10, the number of rows in both train and test data and significantly reduced. The original train dataset contained over 5 million rows, and after grouping the customer IDs, the train dataset now contains a little over 450,000 rows. Meanwhile, the original test dataset contained over 11 million rows, and after grouping the customer IDs, the test dataset now contains a little over 920,000 rows.

The next step is to eliminate any columns with missing values. Many columns of the datasets contain missing values, which state NaN. It is imperative to handle missing values in the dataset. There are various ways in which the missing values can be handled. Sainani (2015) mentions that the best strategy to handle missing values is to simply avoid it all together [13]. Therefore, to reduce the data size as well as utilize the best strategy, this project drops the columns containing any missing values.



```
grouped_train_data.dropna(axis=1, thresh=int(1*len(grouped_train_data)) ,i
nplace=True)
grouped_train_data.shape
```

```
(458913, 83)
```

Figure 5.11: Illustration of the code used to drop columns with missing values in the train data.

From the original train dataset, there were 91 columns. Now, after dropping columns with missing values, there are 83 columns included in the train dataset.

It is essential to note that after grouping duplicates of the customer_IDs and dropping multiple columns in the train dataset, the distribution was slightly affected. Fortunately, though, the effect on the datasets was considered relatively insignificant. To elaborate, the initial train data displayed that 75.10% of the customers did not default (target = 0) and that 24.90% of the customers defaulted (target = 1). Whereas after handling the data, the number of customers that did not default decreased by 1.00% to 74.10% (target = 0) and the numbers of customers that defaulted increased by 1.00% to 25.90% (target = 1).

```
train_data['target'].value_counts().plot(kind='pie', title="Distribution of Target in Training Data", legend=True, \
                   autopct='%1.1f%%', explode=(0, 0), \
                   shadow=True, startangle=0)
```

Figure 5.12: Illustration of the code used to display the distribution of the targets *before* grouping and dropping data.

As shown above, Figure 5.12 displays the code utilized to identify the percentages of the target distribution in the training data before grouping and dropping data. Meanwhile, in Figure 5.13, shown below, the output of the code is displayed. Displayed below, in

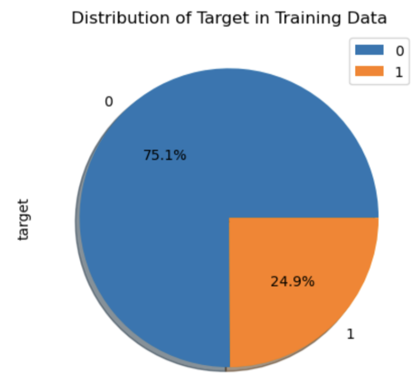


Figure 5.13: Illustration of the output *before* grouping and dropping data.

Figure 5.14, is the code utilized to identify the percentages of the target distribution in the training data after grouping and dropping data.

```
dropping_train['target'].value_counts().plot(kind='pie', title="Distribution of Target in Training Data", legend=True, \
                   autopct='%1.1f%%', explode=(0, 0), \
                   shadow=True, startangle=0)
```

Figure 5.14: Illustration of the code used to display the percentages of the targets *after* grouping and dropping data.

Meanwhile, in Figure 5.15, shown below, the output of the code is displayed.

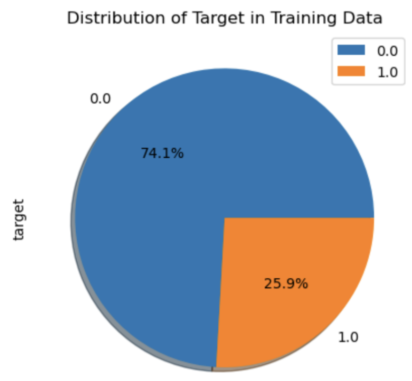


Figure 5.15: Illustration of the output *after* grouping and dropping data.

Finally, to continue reducing the dataset and ensuring the dataset can be read and worked on using the ERT classifier, we dropped two columns, labeled as "customer_IDs" and "S_2." This following code, found in Figure 5.16, was used for both training and test data.

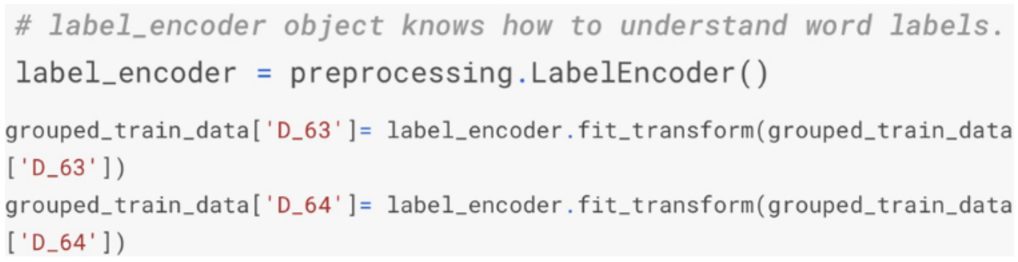
```
dropping_train = grouped_train_data.drop(labels=['customer_ID','S_2'],axis
=1)
dropping_train.head(458913)
```

Figure 5.16: Illustration of the code used to drop the columns in the train data.

5.5 Handling Categorical Variables

In order to continue working with the dataset, it is important to handle the categorical variables. According to Tsunoda, Amasaki, and Monden (2012), there are various methods to handle the categorical variables that can have a positive effect on the model's accuracy [16]. Specifically, past research has claimed that the most effective handling method was label encoding [9]. Therefore, this project will utilize the label encoding method to handle the categorical variables in the dataset.

To recall, the categorical variables in this dataset are 'B_30', 'B_38', 'D_114', 'D_116', 'D_117', 'D_120', 'D_126', 'D_63', 'D_64', 'D_66', 'D_68'. However, due to dropping columns with missing values, the only categorical variables that remain are 'D_63' and 'D_64'. Thus, to ensure the data can continue being worked on and read, we will be encoding the labels.



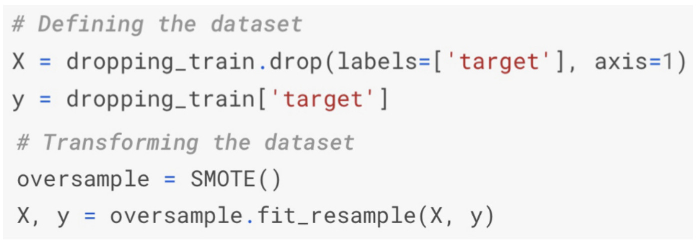
```
# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()
grouped_train_data['D_63']= label_encoder.fit_transform(grouped_train_data
['D_63'])
grouped_train_data['D_64']= label_encoder.fit_transform(grouped_train_data
['D_64'])
```

Figure 5.17: Illustration of the code used to encode the categorical variables.

5.6 Balancing the Data

Next, we will need to balance the data; as mentioned in previous research, one of their limitations in their work was not balancing the data [10]. Past research has discussed that balancing the data overall improves the performance of the machine learning algorithm [1]. For example, in this study, the ERT technique was used. Without balancing the data beforehand, the ERT technique will output inaccurate predictions. Thus, they suggest utilizing one of the balancing techniques, including the synthetic minority oversampling technique (SMOTE) [10].

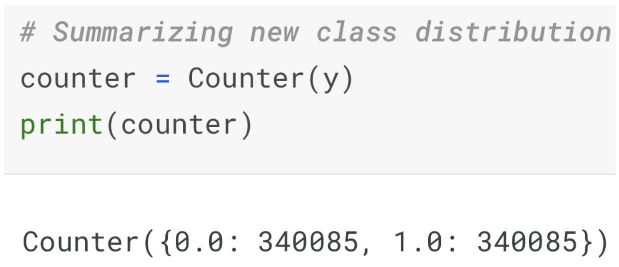
SMOTE balances the data by selecting a random example from the minority class, and identifying multiple examples from the k-Nearest Neighbor. Typically, the value of the k-Nearest Neighbor, written as k, is equal to 5, as it has the highest k accuracy [6]. Then, a random example is chosen from the neighborhood, also known as the feature space, to create a synthetic example between two other points in the feature space. According to Brownlee (2021), using SMOTE to balance the data is effective, as it creates synthetic examples that are relatively close to existing examples from the minority class. However, it is important to note that the downside of using SMOTE to balance data is that it does not consider the majority classes.



```
# Defining the dataset
X = dropping_train.drop(labels=['target'], axis=1)
y = dropping_train['target']
# Transforming the dataset
oversample = SMOTE()
X, y = oversample.fit_resample(X, y)
```

Figure 5.18: Illustration of the code used to utilize SMOTE.

In this study, SMOTE was utilized to ensure even class distribution. Before using the balancing technique, the number of target = 0 was over 340,000, while the number of target = 1 was under 120,000, which shows that the data is significantly unbalanced. After using SMOTE, both targets were equal.



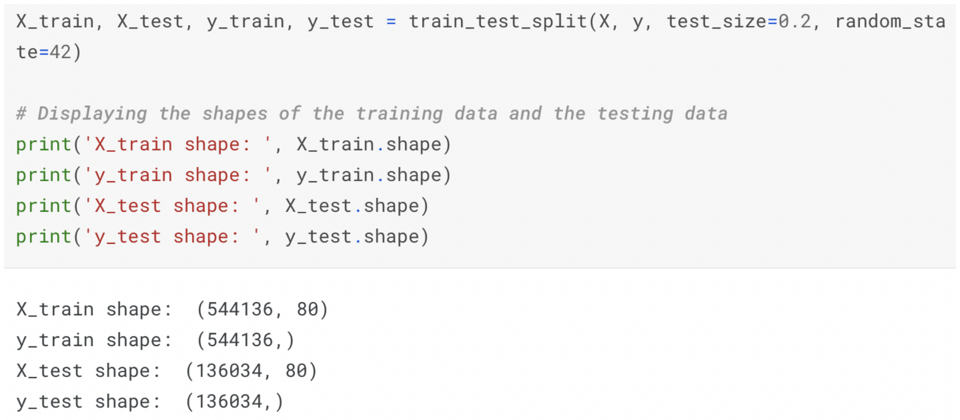
```python
# Summarizing new class distribution
counter = Counter(y)
print(counter)
```

```
Counter({0.0: 340085, 1.0: 340085})
```

Figure 5.19: Illustration of the code used to display the data is balanced.

5.7 Splitting the Data: 80% Train, 20% Test

After ensuring that the dataset is balance, the next step is to split the datasets. When the splitting datasets, there are different ways to do so. For instance, the datasets could be split 50% and 50%, or 70% and 30%; the way it is split depends on the purpose of the program. Generally, splitting the train and test data by 80% and 20%, respectively, tends to be a common practice, specifically in economics. which is known as the Pareto Principle. Thus, this project utilized the Pareto Principle to split the data; the code to split the data is shown in the illustration below.



```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
te=42)

# Displaying the shapes of the training data and the testing data
print('X_train shape: ', X_train.shape)
print('y_train shape: ', y_train.shape)
print('X_test shape: ', X_test.shape)
print('y_test shape: ', y_test.shape)
```

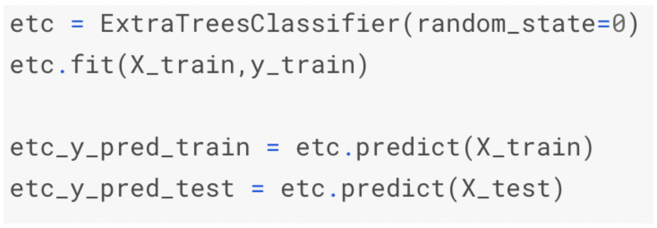
```
X_train shape:  (544136, 80)
y_train shape:  (544136,)
X_test shape:  (136034, 80)
y_test shape:  (136034,)
```

Figure 5.20: Illustration of the code used to split the data.

This principle was introduced in 1906 and first used in macroeconomics by Vilfredo Pareto, an Italian economist. The Pareto Principle states that 80% of the outputs result from 20% of all inputs for any given event. Furthermore, this principle works by identifying the best information from the data and using them efficiently to create maximum value. There have been quite a few success stories regarding the utilization of the 80% of train data and 20% of test data split, despite the lack of thorough scientific investigation proving or disproving the validity of the split. Again, there are different ways that the datasets could be split. However, as this split is considered the most common practice and is mostly used for economics, this work utilized the Pareto Principle to split the datasets.

5.8 Building and Training the Model

Now, to begin discussing the process of building the model, this work was established to utilize the Extremely Randomized Trees technique. It is important to remember that the technique is also referred to as the Extra Trees classifier, as the illustration displayed below states "ExtraTreesClassifier" when building the model.



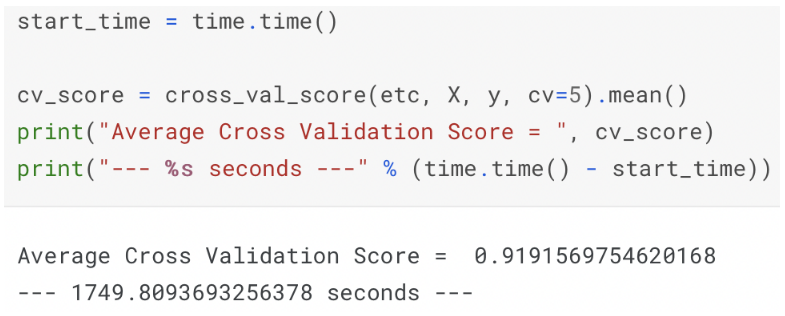
```
etc = ExtraTreesClassifier(random_state=0)
etc.fit(X_train,y_train)

etc_y_pred_train = etc.predict(X_train)
etc_y_pred_test = etc.predict(X_test)
```

Figure 5.21: Illustration of the code used to build the model.

Additionally, while building the model, it is important to fit the model as well, which is shown in the illustration above. When fitting the model, the Extra Trees classifier generates its trees and combines the predictions. As discussed previously, the ERT technique generates a large number of decision trees. After generating a certain number of trees based on available features, the ERT technique then generates predictions based on the specific feature in a specific tree. Finally, the ERT technique will take the average of each trees' predictions to output its overall prediction.

5.8.1 Cross Validation Technique Score

Next, to check and ensure that the model is effective, the cross validation technique was utilized in this project. There are various cross validation techniques that could be used. As shown below in Figure 5.20, this project utilized $cv = 5$, which indicates that the the k-fold cross validation technique was used. As mentioned by Refaeilzadeh et al. (2009), the k, or cv, in k-fold cross validation represents the number of equally sized partitions in the data [12]. This project splits the data into five equal segments, or folds.



```
start_time = time.time()

cv_score = cross_val_score(etc, X, y, cv=5).mean()
print("Average Cross Validation Score = ", cv_score)
print("--- %s seconds ---" % (time.time() - start_time))
```

```
Average Cross Validation Score =  0.9191569754620168
--- 1749.8093693256378 seconds ---
```

Figure 5.22: Illustration of the code used to utilize the cross validation technique.

It is imperative to mention that the cross validation technique is commonly used in various machine learning programs. The overall purpose of the cross validation technique is to estimate the skill of the used machine learning algorithm, in this case, Extra Trees, on unseen data.

Therefore, the cross validation score of 0.92, as displayed above in Figure 5.20, of this project showcases that the machine learning algorithm used performed well and is good in generalizing new, unseen data, which is the test data.

In conclusion, when utilizing machine learning techniques to predict credit card payment defaults, it is imperative to follow a procedure. The first step when creating a program is to import the necessary libraries. Importing libraries prevents rewriting specific codes repeatedly (e.g., writing pd rather than pandas each time it is used) and allows the utilization of various techniques within the program (e.g., utilizing SMOTE to balance the data). After importing the libraries, the next step is to create a computer program to predict the credit card payment defaults. The process begins by accessing the data files that will be used throughout the program. In this case, the data that was accessed contained information about American Express customers. Next, in this program, the handling of the datasets was essential due to various limitations, which will be discussed in a later section. Then, it is important to keep in mind that some datasets contain categorical variables, which may cause issues in the machine learning algorithm, as the variables are not numerical. Thus, it is imperative to handle the categorical variables. The next step is to ensure that the datasets are balanced. To recall, one of the limitations in the previous works was not balancing the data. Therefore, to improve past research, it was essential to balance the datasets. Finally, the programmer can begin building the model. While building the model, it is imperative to split the data and verify the performance of the model by utilizing the cross validation technique.

6. RESULTS AND DISCUSSIONS

This section provides the results and overall findings of this study, as well as provides a discussion on the analysis and interpretation of the results.

6.1 Precision, Recall, F1-Score, and Accuracy Scores

As stated prior, the overall goal of this study was to reduce the financial risk in credit card companies by accurately predicting individuals who are likely to default on their credit card payments. Therefore, it is essential to discover and understand the performance of the chosen machine learning model in terms of its precision scores, recall scores, F1-scores, and accuracy scores when predicting credit card payment defaults.

First, the precision of the model, as mentioned previously, essentially is one indicator of the Extremely Randomized Trees' performance and is typically the most essential indicator. Overall, it is the division of correctly predicted positives by the total number of predicted positives. More specifically, it is the division of the number of customers who are likely to default that the model correctly predicted by the number of customers who are likely to default the model predicted. In this study, the precision scores are provided in Table 6.1.

	Precision
0.0	0.96
1.0	0.89
Accuracy	0.92
Macro Avg	0.92
Weighted Avg	0.92

Table 6.1: Table of Precision Scores in respect to targets, accuracy, macro average, and weighted average.

As shown above in Table 6.1, the machine learning algorithm predicted the number of customers not likely to default at an extremely high rate, as the precision score was 0.96. Additionally, the machine learning algorithm predicted the number of customers likely to default at a high rate, as the precision score was 0.89. Although the precision score was much higher for the individuals unlikely to default compared to the precision score for the individuals likely to default, it is to be expected. As mentioned in another section, the precision is calculated solely based on the positive predictions (i.e., the individuals not likely to default). Therefore, the precision score should have been higher for the individuals not likely to default. Nevertheless, the average of these precision scores was high at 0.92, which showcases that the Extremely Randomized Trees model, in terms of precision, is highly accurate at predicting individuals who are likely to default on their credit card payments.

Next, the recall score of the model, as mentioned previously, essentially is another indicator of the model's performance; it is the division of correctly classified positives by the total number of classified positives. It is essential, as it reduces the number of false negatives. The recall scores of the ERT technique for this study are provided in Table 6.2.

	Recall
0.0	0.88
1.0	0.96
Accuracy	0.92
Macro Avg	0.92
Weighted Avg	0.92

Table 6.2: Table of Recall Scores in respect to targets, accuracy, macro average, and weighted average.

As shown above in Table 6.2, the machine learning algorithm predicted the number of customers not likely to default at a high rate, as the recall score was 0.88. Additionally, the machine learning algorithm predicted the number of customers likely to default at an extremely high rate, as the recall score was 0.96. Similar to the precision score, it is expected that the recall scores are higher for the individuals who are likely to default on their credit card payments, as it determines the model is making fewer false negative predictions. Therefore, the recall score should have been higher for the individuals likely to default. However, the average of these recall scores was high at 0.92, which showcases that the ERT model, in terms of recall, is highly accurate at predicting individuals who are likely to default on their credit card payments.

Additionally, due to the differences in the precision and recall scores, it is important to note that, typically, the higher the precision score, the lower the recall score, and vice versa. Either way, however, the average scores for both the precision and recall are at 0.92, which means that the ERT is highly accurate at predicting individuals likely and unlikely to default.

Now, the F1-Score of the model, as mentioned previously, is the harmonica mean of precision and recall values of the classifier. The F1-Scores are provided in Table 6.3.

	F1-Score
0.0	0.92
1.0	0.92
Accuracy	0.92
Macro Avg	0.92
Weighted Avg	0.92

Table 6.3: Table of F1-Scores in respect to targets, accuracy, macro average, and weighted average.

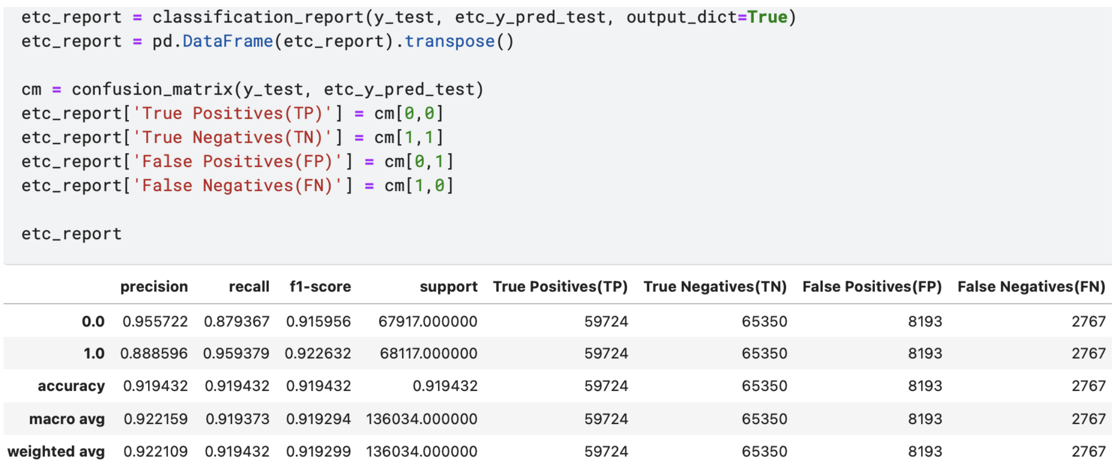
As shown above in Table 6.3, the F1-Score of the machine learning algorithm used in this project displays both the numbers of customers unlikely and likely to default at a high rate, as the F1-Score was 0.92. This score showcases that the ERT model is highly accurate at predicting individuals who are likely to default on their credit card payments.

Finally, it is essential to discover and understand the overall scores and information on the ERT technique; thus, the scores of each indicator are provided in Table 6.4 below.

	Precision	Recall	F1- Score
0.0	0.96	0.88	0.92
1.0	0.89	0.96	0.92
Accuracy	0.92	0.92	0.92
Macro Avg	0.92	0.92	0.92
Weighted Avg	0.92	0.92	0.92

Table 6.4: Table of all scores in respect to targets, accuracy, macro average, and weighted average.

The information was provided by utilizing the code shown below in Figure 6.1 and its output. From Figure 6.1, we can see that there are True Positive, True Negative, False Positive, and False Negative values. Each of these values indicate the model's prediction of positive and negative classes. Specifically, the true positive value indicates the model's correct prediction of positive classes, in this case, customers who are not likely to default; meanwhile the true negative value indicates the model's correct prediction of negative classes, in this case, customers who are likely to default. Similarly, the false positive value indicates the model's incorrect prediction of positive classes; meanwhile, the false negative value indicates the model's incorrect prediction of negative classes. Each of these values are then incorporated into the formulas to identify the results of the precision score, recall score, and F1-score. Again, the results shown in Table 6.4 were the outputs from the code in Figure 6.1, where it calculated the values of TP, TN, FP, and FN and used the necessary formulas to overall discover the performance of the machine learning algorithm.



```
etc_report = classification_report(y_test, etc_y_pred_test, output_dict=True)
etc_report = pd.DataFrame(etc_report).transpose()

cm = confusion_matrix(y_test, etc_y_pred_test)
etc_report['True Positives(TP)'] = cm[0,0]
etc_report['True Negatives(TN)'] = cm[1,1]
etc_report['False Positives(FP)'] = cm[0,1]
etc_report['False Negatives(FN)'] = cm[1,0]

etc_report
```

	precision	recall	f1-score	support	True Positives(TP)	True Negatives(TN)	False Positives(FP)	False Negatives(FN)
0.0	0.955722	0.879367	0.915956	67917.000000	59724	65350	8193	2767
1.0	0.888596	0.959379	0.922632	68117.000000	59724	65350	8193	2767
accuracy	0.919432	0.919432	0.919432	0.919432	59724	65350	8193	2767
macro avg	0.922159	0.919373	0.919294	136034.000000	59724	65350	8193	2767
weighted avg	0.922109	0.919432	0.919299	136034.000000	59724	65350	8193	2767

Figure 6.1: Illustration of the code used to display the scores and the output.

Now that we have determined that the machine learning model will perform very well on unseen data, it is now time to utilize the test data to make the predictions.

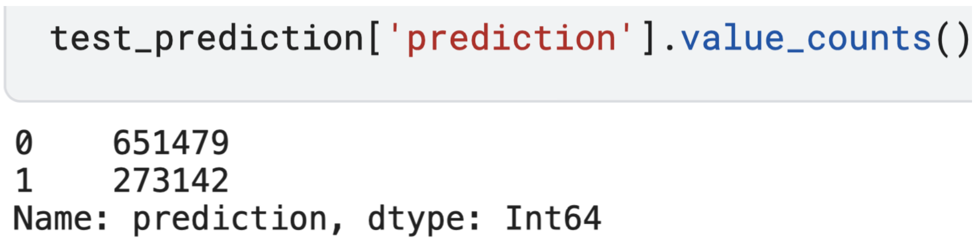
6.2 Making the Predictions

Before we can utilize the test data to make the predictions, it is important to be aware of the test data shape. After handling the missing values, the test data we worked with contained 924,621 rows and 80 columns (shape of test data displayed as: (924621, 80)).

To make a prediction on the test data, we will utilize the predict function code and apply it to the test data. Then, we will identify the number of customers not likely to default as well as the number of customers likely to default. The results are shown below in Figure 6.2.

Based on the results above in Figure 6.2, the machine learning model predicted 651,479 of the 924,621 customers will not default, which is equal to 70.50%. Meanwhile the model predicted that 273,142 customers will default, which is equal to 29.50%.

As the distribution of the target values in the train data was displayed in a pie chart,



```
test_prediction['prediction'].value_counts()
```

```
0    651479
1    273142
Name: prediction, dtype: Int64
```

Figure 6.2: Illustration of the code used to display the dataset and its output.

the predictions of the target values of the test data is displayed in a pie chart, as shown below in Figure 6.3, to make a more logical comparison.

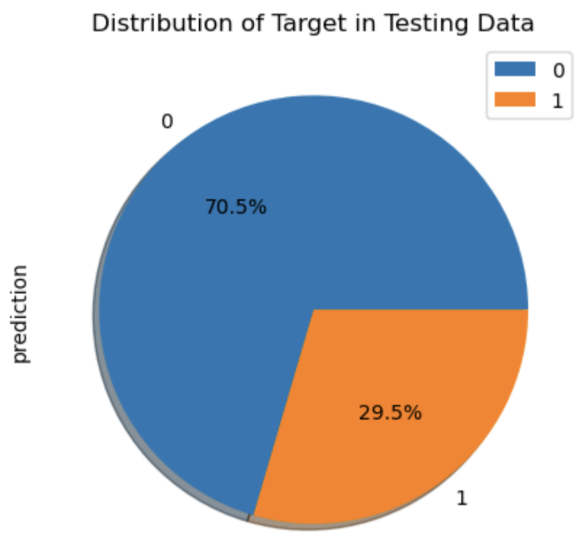


Figure 6.3: Illustration of the percentages of the target distribution.

Referring back to Figure 5.15, it is essential to note that the train data displayed 74.10% of targets = 0 (customers not likely to default) and 25.90% of targets = 1 (customers likely to default), while the test data in Figure 6.3 displayed 70.50% of targets = 0 and 29.50% of targets = 1.

We can see that there is an approximately 4% difference when comparing the percentages of the targets in the train data and the percentages of the predictions in the test data. It is essential to keep in mind that there are different reasons for this slight difference. For instance, the machine learning model is overfitting to the train data. However, as mentioned previously, this project utilized the cross validation technique to reduce the risk of overfitting, so it is presumed that the differences in the percentages are due to natural variance in the data or the specific samples used in the test dataset rather than overfitting. The scores of performance metrics (precision, recall, F1-score, and accuracy) support this presumption, as Table 6.4 displays that the machine learning model is highly accurate. Therefore, we know that the Extremely Randomized Trees technique was able to accurately predict the individuals likely to default on their credit card payments.

6.3 Overall Findings

Thus, the overall findings of this project, based on the scores displayed above, showcases that the process performed to handle the train data did not have much impact on the original data and the machine learning algorithm's performance, as it was evaluated at 0.92, which means it is an accurate machine learning model. To recall, the process to handle the train data included grouping duplicates of the customer_IDs, dropping columns with missing values, and handling categorical variables. Thus, the methods to reduce the train dataset is considered sufficient.

However, this project experienced various limitations, such as the lack of computational resources. Thus, even though the results of the performance metrics display that the machine learning model is highly accurate, the process to reduce the train data and the test data make have caused the scores of the performance metrics to lower. This topic is elaborated in the next section.

7. LIMITATIONS

This section discusses the various limitations that were encountered to analyze the data within this study.

7.1 Limited Access to Computational Resources

Unfortunately, this study accessed and utilized data that was computationally expensive. The data was too large to download onto a local drive. Thus, to be able to access and use the data, cloud-based computing software has been utilized.

7.1.1 Cloud Based Computing Software: Kaggle

The data accessed throughout this project was provided by one of Kaggle's users. Unfortunately, the data could not be downloaded and saved locally due to the size, as there were over a total of 16 million rows within the dataset. Thus, when utilizing Kaggle, the data was stored and used throughout the program rather than through local software.

7.1.2 Insufficient Memory

The next issue in this program was having insufficient memory. When working on the program, the pop-up notification "Your notebook tried to allocate more memory than available. It has been restarted" would appear numerously. Again, the size of the data caused this issue. Thus, the data needed to be handled and compressed. As discussed previously, the process to decrease the data size occurred when working with both the train and test datasets. Although the process did not significantly impact accuracy rates of the precision, recall, and F1-scores for the train data, it is displayed that the process did significantly affect the accuracy rate of the precision, recall, and F1-scores for the test data.

Additionally, when dealing with insufficient memory, not only the data was impacted but so was the various techniques utilized in the project. For instance, this project utilized SMOTE to ensure the data was balanced and ERT to make the prediction. However, due to inadequate memory, the processing times for these machine learning techniques were very slow. The program takes approximately 51 minutes to run, even after reducing the data remarkably.

In conclusion, not having enough memory for this study overall caused slow processing times, multiple system crashes, and at times, made the analysis impossible. This limitation essentially affected the quality of the prediction outputs.

Therefore, the lack of access to overall sufficient computational resources resulted in reducing the size of the datasets significantly, and ultimately lowered the quality of the prediction.

In conclusion, this research encountered a few limitations. However, future directions, discussed in a later section, showcases how to improve this research and the next steps to take to ensure improvement.

8. FUTURE DIRECTIONS

In order for improvement within this discussion, the limitations discussed above need to be addressed and solutions must be provided. For instance, as not having enough memory to access a large amount of data was the most significant limitation of this study, it is important to know various techniques that can reduce the dataset without losing imperative information, such as the Principal Component Analysis (PCA) or Singular Value Decomposition (SVD). These two methods can preserve most of the essential information while also reducing the dimensionality of the data. Additionally, if using cloud-based computing, such as Kaggle or Amazon SageMaker, it is recommended to increase the storage/memory space in order to ensure there is enough for large data. Therefore, this work will be continued by utilizing various techniques to ensure improvement.

Additionally, future plans for this work include presenting in a national conference and publishing the work. Therefore, the future directions of this research include investigating potential conferences as well as researching and identifying publishers, particularly those who have published similar works and specialize in this field of study. Additionally, it would be imperative to discuss this process with experienced individuals and seek guidance.

In conclusion, it is imperative to note that there is room for improvement in this study and plans have been made to continue expanding on this research. As a result, future work of this project will address the limitations previously mentioned through various techniques in order to ensure more accurate information.

9. CONCLUSIONS

There has been a lot of research contributing to the machine learning world, specifically in regard to predicting credit card payment defaults. Data is continuously being released for public use as well as competition use. Many individuals have utilized various machine learning techniques to predict credit card payment defaults; in fact, there has been previous research that has shown the Extremely Randomized Trees algorithm being the most accurate machine learning technique to predict credit card payment defaults. However, the drawback of this claim was that the previous work used outdated data. It is essential to acknowledge that human behavior is consistently changing, day by day and year by year. Therefore, this project determined to use more recent data to discover whether the claim that the Extremely Randomized Trees technique is still the most accurate machine learning algorithm to predict credit card payment defaults. However, when using the more recent data, some obstacles occurred, which could have impacted the results of this project. Nevertheless, this research ensured to describe the limitations, as well as the risks of handling such a large dataset. Additionally, this project also ensured to discuss in detail how the program was created and explained the process thoroughly.

Furthermore, it is imperative to note that although this project experienced limitations, the claim that the Extremely Randomized Trees technique could still be relevant. For instance, when discussing the accuracy rates of the ERT technique utilized throughout this project, it was apparent that they were closely similar to the accuracy rates of the LGBM technique used by another programmer. However, due to minor processes that reduced the dataset, the overall test accuracy rate was not as high as the scores of the precision, recall, and F1-score. Therefore, it is essential to continue with this research and ensure access to sufficient computational resources to either support or counter the claim that the Extremely Randomized Trees technique is still the most relevant machine learning algorithm to predict credit card payment defaults. In the meantime, it is essential to mention that, based on a leaderboard, recent work has shown other machine learning algorithms as the most accurate.

In conclusion, predicting credit card payment defaults by utilizing machine learning techniques is a topic that will need to be continuously discussed and researched through future works, as more and more data is available in addition to the fear that changes in the data analysis field that inevitably enormous. The reactive approach is based on analysis of default or non-default of credit card payment. An attempt is made to understand the factors associated with the the defaulters, and then to put measures in place to prevent future defaults anticipated in the current structure.

10. REFERENCES

[1] T. M. ALAM, K. SHAUKAT, I. A. HAMEED, S. LUO, M. U. SARWAR, S. SHABBIR, J. LI, & M. KHUSHI, (2020), *An Investigation of Credit Card Default Prediction in the Imbalanced Datasets*, IEEE Access: Practical Innovations, Open Solutions, 8, 201173-201198. doi:10.1109/ACCESS.2020.3033784.

[2] L. BREIMAN, (2001), *Random Forests,* Machine Learning 45, 5–32. https://doi.org/10.1023/A:1010933404324

[3] M. DOMINGUEZ, (2021), *Predicting credit card defaults with machine learning*, Medium: The Startup.

[4] L. F. DUNN, I. A. MIRZAIE, (2022), *Gender Differences in Consumer Debt Stress: Impacts on Job Performance, Family Life and Health*, J Fam Econ Iss. https://doi.org/10.1007/s10834-022-09862-z

[5] P. GEURTS, D. ERNST, & L. WEHENKEL, (2006), *Extremely randomized trees,* Machine learning 63.1: 3-42.

[6] H. HENDERI, T. WAHYUNINGSIH, & E. RAHWANTO, (2021), *Comparison of Min-Max normalization and Z-Score Normalization in the K-nearest neighbor (kNN) Algorithm to Test the Accuracy of Types of Breast Cancer*, International Journal of Informatics and Information Systems, 4(1), 13-20. doi:https://doi.org/10.47738/ijiis.v4i1.73

[7] F.A.I.R. INSTITUTE, (n.d.), *Quantitative information risk management: The Fair Institute*, Quantitative Information Risk Management — The FAIR Institute.

[8] S. R. ISLAM, W. EBERLE, AND S. K. GHAFOOR, (2018), *Credit Default Mining using Combined Machine Learning and Heuristic Approach*, ArXiv.

[9] E. JACKSON R. AGRAWAL, (2019), *Performance Evaluation of Different Feature Encoding Schemes on Cybersecurity Logs*, SoutheastCon, Huntsville, AL, USA, 2019, pp. 1-9, doi: 10.1109/SoutheastCon42311.2019.9020560.

[10] S. V. JAIN, & M. JAYABALAN, (2022), *Applying Machine Learning Methods for Credit Card Payment Default Prediction With Cost Savings*, Biomedical and Business Applications Using Artificial Neural Networks and Machine Learning: 285-305.

[11] G. LI, (2018), *Gender-related differences in credit use and credit scores*, The Fed - Gender-Related Differences in Credit Use and Credit Scores.

[12] P. REFAEILZADEH, L TANG, H. LIU, (2009), *Cross-Validation*, In: LIU, L., Ö ZSU, M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9 565

[13] K. SAINANI, (2015), *Dealing with Missing Data*, PM& R 7.9: 990-994, ISSN 1934-1482, https://doi.org/10.1016/j.pmrj.2015.07.011.

[14] Y. SAYJADAH, I. A. T. HASHEM, F. ALOTAIBI, & K. A. KASMIRAN, (2018), *Credit Card Default Prediction using Machine Learning Techniques*, Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA), 1-4.

[15] S. L. STOLBA, (2020), *Married consumers have higher credit scores and debt than single adults*, In Experian.

[16] M. TSUNADA, S. AMASAKI, & A. MONDEN, (2012), HANDLING CATEGORICAL VARIABLES IN EFFORT ESTIMATION, In Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement (ESEM '12). Association for Computing Machinery, New York, NY, USA, 99–102. https://doi.org/10.1145/2372251.2372267

[17] M. A. ULLAH, M. M. ALAM, S. SULTANA, & R. S. TOMA, (2018), *Predicting Default Payment of Credit Card Users: Applying Data Mining Techniques*, 2018 International Conference on Innovations in Science, Engineering and Technology (ICISET), 355-360.

11. VITA

Madison Renee Guerra graduated Summa Cum Laude with her Bachelor of Arts degree in Mathematics at Texas A&M International University in May 2021. During her undergraduate studies, Madison was a member of the University Honors Program, where she completed various courses with honors credit and worked on an undergraduate thesis entitled "Set Constructions: The Construction of the Set of Natural Numbers and the Set of Integers".