ASHESI UNIVERSITY COLLEGE

ANALYSIS OF CREDIT CARD FRAUD DETECTION METHODS

UNDERGRADUATE THESIS

B.Sc. Computer Science

Godlove Otoo

2021

ASHESI UNIVERSITY COLLEGE

ANALYSIS OF CREDIT CARD FRAUD DETECTION METHODS

UNDERGRADUATE THESIS

Thesis submitted to the Department of Computer Science, Ashesi University College in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Science.

Godlove Otoo

2021

DECLARATION

I hereby declare that this Undergraduate Thesis is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature: 

……

Candidate's Name: Godlove Otoo

……

Date: 13th May, 2021

……

I hereby declare that preparation and presentation of this Undergraduate Thesis were supervised in accordance with the guidelines on supervision of Undergraduate Thesis laid down by Ashesi University.

Supervisor's Signature: 

……

Supervisor's Name:

Dr Justice K. Appati

……

Date:

13th May 2021

……

Acknowledgements

Many individuals contributed to the successful completion of this project so much so that it would be impossible to mention every one of them. However, I would like to first of all thank the Almighty God for His guidance and strength throughout the project. Indeed, His grace abounded towards me. I would also like to thank my supervisor, Dr Justice K. Appati, for his ever-present counsel and sound academic advice that helped me complete this project. To my friends Daniel, Emmanuel, Ruth and Tamisha and my parents Mr. and Mrs. Otoo, I say a big thank you for your support throughout the journey.

Last but not least, I doff my hat to my grandad, Mr. Emmanuel N. K. Otoo, whose words were a huge motivation for me as I worked on this project – "The heights by which great men reached were not attained by sudden flights but they, while their companions slept were toiling upwards in the night."

Abstract

Technological advancements in the field of finance have brought about several ways to buy and sell via the internet. With the rampant increase in digitization worldwide, more people are leaning towards the use of the internet as the main medium for conducting transactions. Credit cards serve as one of the modes of payment for these transactions and fraudulent activities in the use of these cards have caused a great deal of harm to several institutions in the online market space. Unfortunately, detection of this menace is not a straightforward task as it has two main issues associated with it: (1) The problem of data imbalance in credit card fraud data (2) The profiles of fraudulent and genuine users being dynamic. This project tackles the problem of data imbalance in credit card fraud data by adopting a resampling approach in combination with three different classification algorithms to detect instances of credit card fraud. The dataset used contained 284,315 genuine transactions and 492 fraudulent transactions, making it highly imbalanced. Such data may cause classification algorithms to be biased towards the majority class (the class with genuine users) since they are designed with the assumption that they are working with a fairly equal number of examples for each class. Hence, different resampling techniques were used to resample the data before Logistic Regression, Naïve Bayes and the K-Nearest Neighbor algorithm were used to predict if a transaction was fraudulent or not. The K-Nearest Neighbor classifier obtained the best performance in terms of f1 score and Precision-Recall area under curve (PR AUC) score when it was used with the Neighborhood Cleaning Rule for undersampling. The values obtained for these performance metrics were 82.5% and 81% respectively.

Keywords: *credit card; fraud data; resampling; machine learning*

Table of Content

List of Figures

Chapter 1: Introduction

1.1 Background

Financial transactions are conducted in many ways and on different platforms daily. With numerous technological advancements in the field of finance, these transactions are gradually transitioning from the traditional in-person mode of execution to the internet as their main mode of execution. Buying and selling goods and services online (E-commerce) has become one of the most convenient means for people to conduct financial transactions as it gives them the luxury of ordering items online and having them delivered to their doorstep. In addition to the comfort associated with online transactions, a lot of time is saved when people buy and sell on the internet since buyers can buy from almost anywhere in the world, and sellers can reach a wider audience and make more purchases in a short time. Buyers can also compare prices online and find substitute products with little effort as compared to doing this in-person. This makes E-commerce a more attractive option for conducting financial transactions. On the whole, both buyers and sellers save an appreciable amount of time and money when they conduct their day-to-day transactions online.

As shown above, E-commerce plays a key role in the financial sector and as a result of this, the practice has seen rapid growth in its use since it began in 1995[2]. According to Statista, E-commerce sales online in 2019 was approximately 3.53 trillion US dollars and returns from these sales is expected to rise to approximately 6.54 trillion US dollars in 2022 [4]. An article by John Koetsier reports that total online spending in May was 82.5 billion US dollars indicating an increase of 77% from last year [6]. The author attributed this increase to the coronavirus pandemic and stated that what was observed in May would have taken between four and six years if the rate of growth remained the same as that of the past few years [6].

Credit Cards serve as one of the means by which transactions are conducted online. They are cards that allow users to purchase items and conduct bank transfers on credit; issuers

of these cards expect that the user pays the amount credited in the future. Due to the nature of these cards, there are certain features like the grace period for payment of the user's balance and the credit limit associated with every credit card. It is reported that in the first quarter of the year 2020, three hundred and forty million VISA credit cards were being used in the United States of America while eight hundred and one million of these cards were being used outside the country [9]. These statistics show that there is a huge credit card user base.

With E-commerce and credit card usage growing rapidly, fraudulent use of credit cards for transactions online has also been on the rise. According to Katy Worobec - the Managing Director of the Economic Crime Division at United Kingdom Finance - British businesses suffered a loss of 844 million dollars from card fraud in 2018, out of which 495 million dollars was lost through online transactions [10].

1.2 Problem Statement

Given that revenue in the E-commerce market is projected to show a growth rate of 8.2% and reach an estimated value of 3.2 million US dollars by 2024 [3], an increase in the number of fraudulent activities will do a great deal of harm to business in this market. Hence, the need for detection of the occurrence. Credit Card Fraud Detection involves putting mechanisms in place to identify and flag credit card transactions that are conducted illegally. Different methods such as the use of genetic algorithms and classification algorithms have been applied to credit card fraud data in attempts to detect credit card fraud, however, there are a few problems that show up in the detection process. I highlight two of these problems in this project, but the objective of the research is centered around one of them.

The first is that the profiles of normal and fraudulent users of credit cards are dynamic [5], that is, these profiles change constantly with time, and hence, it is difficult to spell out a fixed set of attributes that distinguishes a normal credit card user from a fraudulent one. In addition to this, transactions conducted by fraudulent and genuine users can be very similar

and fraudsters try their best to remain in this camouflage. In their work on dynamic credit card profiling, Damez *et al* explain that fraudsters are constantly devising new strategies to find their way around anti-fraud policies [7] hence, the detection methods put in place should make room for this dynamic behavior.

The second problem - which is the focus of this research - has to do with an imbalance in the classification of credit card fraud data where the number of genuine transactions greatly outweighs the number of fraudulent transactions. Most classification algorithms assume that the data they are working with has an equal number of examples for each class and there is an equal cost for each misclassification [11]. These assumptions affect their performance on data which has an unequal number of examples for each class. In this project, the degree of inequality considered for a dataset to imbalanced falls in line with the categorizations of imbalance described by Haibo He and Yunqian Ma in their book on Imbalanced learning. They consider a 10:1 distribution of the majority class to the minority class as modestly imbalanced and a 1000:1 imbalance ratio or anything above it as extremely unbalanced [12].

1.3 Objective of Research

The objective of this research is therefore to identify which resampling techniques best fit credit card fraud data and analyze their efficiency when they are used with classification algorithms.

1.4 Research Question

What degree of resampling brings significant improvement in the detection of credit card fraud?

1.5 Outline of Methodology

To tackle the problem of data imbalance, the approach used in this research focuses more on the data pre-processing stage. Different sampling techniques are explored at this stage to provide some form of balance to the dataset and propose a technique which produces the best results when used with a classification algorithm. The dataset used is highly imbalanced. It consists of 284,807 transactions out of which 492 (0.172%) are cases of credit card fraud. It was used by Worldline and the Machine Learning group of ULB (Université Libre de Bruxelles) for a research project on big data mining and fraud detection. At the data pre-processing stage, sampling techniques such as hybrid sampling, oversampling and undersampling techniques will be used to resample the dataset. Feature selection and Dimensionality reduction using Principal Component Analysis have been carried out on the dataset already so the next step will be to train the classification algorithms and test them. Three classification models are used in this research: K-Nearest Neighbor, Logistic Regression and Naïve Bayes. They are trained using 70% of the dataset and tested with the remaining 30%. Their performance will be evaluated with precision, recall, F1 score, and the Precision-Recall Area under curve (PR AUC) score.

1.6 Justification of Study

Working with imbalanced datasets like credit card fraud datasets can be very challenging because of the aforementioned assumptions that classification models work with. There can be cases where a model ignores the minority class and predicts that all the examples are a part of the majority class [13]. Assuming we have a credit card dataset with 200,000 transactions and only 0.1% of these transactions are cases of fraudulent activities, the classification model may predict that all the transactions are genuine and have close to a 100% accuracy for the majority class. This will mean that the 200 transactions that were not genuine in the dataset will go undetected although we seem to have an almost perfect accuracy for our model. The

two main issues highlighted here are that working with imbalanced datasets is a challenging task and accuracy as a performance metric is not the best in such cases.

Various approaches have been put forward to solve the problem of imbalance in datasets; sampling or data pre-processing, the algorithmic approach, and the feature selection approach [13]. This project focuses mainly on the sampling approach to solve the problem.

1.7 Outline of Study

In chapter two, emphasis is placed on existing works that have been done in attempt to detect credit card fraud. The chapter is divided into two sections - the first section expounds on the use of machine learning with different resampling techniques to predict the occurrence while the second section looks at the performance of classification algorithms that have been used on credit card fraud data.

Chapter three gives an in-depth description of stages that are gone through in this project. It goes into detail on the dataset, the oversampling techniques and the three algorithms that are selected to be used in throughout the research.

In Chapter four, results are discussed. The performance of the classification algorithms in conjunction with the resampling techniques are evaluated using various performance metrics.

Chapter five is the concluding part of the project. It captures the challenges faced, further work and recommendations.

Chapter 2: Literature Review

An appreciable amount of work has been done around credit card fraud detection and different approaches ranging from hard regulations – regulations that involve the use of the law - to soft regulations – regulations that involve technological solutions [1] - have been employed to help curb the practice. This chapter focuses on two soft regulation categories that have been used to detect credit card fraud; the use of machine learning models with different sampling techniques to predict the occurrence and comparative studies on the performance of classification algorithms on credit card fraud data. The works discussed in this section have helped shape the scope of this project.

The first category focuses on the ways by which different machine learning models have been used with sampling techniques to detect credit card fraud, however, the second category is limited to classification models. While the main focus in the first category is solving the problem of imbalance in credit card fraud datasets, the focus of the second category is to identify which classification models perform well in detecting fraud with or without resampling. The second category informed which classification models were used in this research.

2.1 The use of machine learning with different sampling techniques to predict the occurrence

Ronish Shakya [8] performed an analysis on a credit card fraud dataset by using different sampling techniques and machine learning algorithms like logistic regression, random forest, and xgboost on the data. At the feature engineering stage, Ronish Shakya uses Principal Component Analysis (a dimensionality reduction technique) to reduce the large number of variables in the dataset to a smaller subset of feature variables [8]. He further standardizes the dataset, splits it into training and testing data, and finally resamples the dataset using resampling techniques such as random undersampling and oversampling, SMOTE, Tomek

links removal, a combination of SMOTE and Tomek links removal to make the dataset balanced [8]. To govern the training process, Ronish Shakya uses 10-fold cross-validation to tune the hyperparameters of each of the machine learning models before passing the resampled data to each of these models. Ronish Shakya evaluates the performance of each of the models using Precision, Recall, F1 score, area under precision-recall curve, and the area under receiver operating characteristic curve as the performance metrics. She explains that she did not use accuracy in evaluating performance because it gives misleading conclusions when it is used on an imbalanced classification dataset [8]. The results of her research revealed that random forests performed better than the other models when it was used with SMOTE and Tomek links removal. The only limitation of this research was that it was conducted solely for stationary credit card fraud data; credit card fraud is a practice that changes continually with time and hence, a more efficient approach would have been one which can adapt to the dynamic nature of the phenomenon. However, I appreciate the author's feature engineering and resampling approach and hope to build on it in this research.

In their study on the performance of supervised machine learning algorithms for credit card fraud detection, Dhankhad *et al* compare the performance of ten supervised machine learning algorithms and conclude that based on the performance metrics they used, the stacking classifier outperformed all other algorithms [14]. Although their focus was on the comparison of the algorithms, they acknowledged that an imbalance in the dataset can affect the results they obtain from the algorithms. Hence, they performed under-sampling on the data before the supervised learning models were evaluated with accuracy, precision, recall, F1-score, G-Mean, FPR, TPR, and specificity. Comparing this study to Ronish Shakya's paper on the Application of Machine Learning Techniques in Credit Card Fraud Detection, we see that although these works have a similar objective, the former gives more attention to the imbalance in data than the latter. The former was also more detailed in the presentation of its results; as it showed the performance of the algorithms when no sampling had been done and

when different sampling techniques had been applied to the data. Dhankhad *et al* could have been more informative in their approach by showing how the algorithms performed with and without resampling. This would have helped readers know if their proposed super classifier's performance is limited to only the under-sampling which was done or could be used in all instances.

2.2 Comparative Studies on the performance of classification algorithms on credit card fraud data

Awoyemi *et al* [5] carried out a comparative analysis of the performance of the K-nearest neighbor algorithm, Naïve Bayes, and Logistic Regression on credit card fraud data that was resampled using hybrid sampling. The dataset used consisted of 284,807 transactions which took place within two days; out of these, only 0.172% were fraudulent transactions [5]. Principal Component Analysis was then performed on the dataset and a subset of 30 features were used in the study. To bring some form of balance into the data, Awoyemi *et al* applied a hybrid of undersampling and oversampling to achieve two sets of distribution of the data (10:90 and 34:64) before the classification algorithms were used on the data. 70% of the data was used for training and 30% was used for testing. Six performance metrics were used to evaluate the results from the algorithms; accuracy, sensitivity, specificity, precision, Matthews correlation coefficient, and balanced classification rate. The results from their study showed that hybrid sampling improves the performance of the three classification algorithms on the data and that the k-nearest neighbor performed best across the performance metrics. The only time it was outperformed was when accuracy was used to measure its performance on under-sampled data.

Although the central aim of the works mentioned in this chapter is to detect credit card fraud, the focus in the detection process has been on the machine learning algorithms. In some

of them, one or two resampling techniques have been applied to the data used. However, the resampling approaches employed in these cases has not been extensive.

Chapter 3: Methodology

The approach used in this project, as has been mentioned in previous sections, focuses mainly on the extensiveness of resampling and the performance of three classification algorithms on resampled data. To do this, credit card fraud data was downloaded from Kaggle for the analysis to be carried out on, after which the analysis itself was done using five resampling techniques and three classification algorithms.

3.1 Dataset

The dataset used in this research is the Credit Card Fraud Detection dataset provided by Kaggle (an online community of data scientists and machine learning practitioners). It contains 284,807 financial transactions conducted with credit cards, out of which 492 (0.172%) are labeled as fraudulent transactions. It was used by Worldline and the Machine Learning group of ULB (Université Libre de Bruxelles) for a research project on big data mining and fraud detection. All variables in the dataset are numerical; 28 of them were obtained with Principal Component Analysis and the remaining three variables represent Time, Amount, and the Class the transaction belongs to (genuine or fraudulent). Below is a short description of the dataset:

1. **Time**: The number of seconds elapsed between each transaction and the first transaction in the dataset.
2. **V1, V2, …, V28**: Principal Components obtained with Principal Component Analysis.
3. **Amount**: The transaction amount.
4. **Class**: The response variable. It indicates fraudulent transactions with the value 1 and genuine transactions with 0.

3.2 Simulation Environment

To carry out this analysis, Google Colaboratory was used as the development environment. This cloud-based environment provides free access to GPUs and did not require any configuration; it already had installed the needed python libraries.

Scikit learn, Matplotlib, and Imblearn were the main python libraries used in this project. Scikit learn provided access to the classification algorithms used, Matplotlib was used for data visualization and Imblearn served as the package for accessing the resampling techniques used in this project.

3.3 Data Pre-processing stage

At this stage, the dataset is cleaned, and then some features are standardized to bring uniformity in terms of their units of measurement. It is then resampled using the different resampling techniques listed below. It is finally split into the training and testing sets for use by the classification models. Resampling performed during this stage served as the main focus of the experiment.

3.3.1 Data Cleaning and Validation

Missing values in the dataset are identified and other errors are investigated for in the data. This was to ensure that all the values for each column were in line with the specified formats for that column.

3.3.2 Data Standardization

Data Standardization involves rescaling the data such that the mean becomes 0 and the standard deviation becomes 1. The formula used to achieve this is shown below:

$$z = \frac{x_i - \mu}{\sigma} \qquad 3.1$$

where $\mathbf{x_i}$ = the value being standardized, $\boldsymbol{\mu}$ = the mean and $\boldsymbol{\sigma}$ = the standard deviation

By doing this, we can now compare features in the dataset which have different units and use them in our classification models. Outliers are also adjusted for during this process and a more normal distribution is achieved. Data Standardization was done as part of the Principal Component Analysis.

3.3.3 Data Resampling

Three categories of resampling are used in this project; Under-sampling, Over-sampling, and a Hybrid of the two. Random Over-sampling and Under-sampling methods were not used as part of the techniques because the former tends to cause overfitting to occur in the classifier while the latter can cause the loss of useful information.

Under-sampling techniques

Under-sampling involves discarding some instances of the majority class to balance the class distribution for the dataset. When this is done randomly, it may result in the loss of useful information that may have contributed to the classification phase of the project [17]. Hence, this project makes use of techniques that aim to retain useful information present in the majority class by using under-sampling techniques that remove redundant noise and/or borderline instances from the dataset. One-Sided Selection for Under-sampling and Neighborhood Cleaning Rule for undersampling are the two under-sampling techniques used in this project. Unlike random under-sampling, they both select which examples to keep and delete from the dataset based on certain heuristics and learning models.

One-Sided Selection for Under-sampling

This under-sampling technique combines Tomek Links and the Condensed Nearest Neighbor (CNN) rule to remove redundant and ambiguous points from the majority class.

Tomek links take care of the noisy and borderline examples while the CNN rule removes the examples which are distant from the decision border. The borderline examples are those which can easily fall on the wrong side of the border with a small amount of noise [18].

The algorithm for this technique is shown below [15]:

1. Let S be the training set.
2. Initially, C contains all positive examples from S and one randomly selected negative example.
3. Classify with the 1-NN rule using the examples in C, and compare the assigned concept labels with the original ones. Move all misclassified examples into C that is now consistent with S while being smaller.
4. Remove from C all negative examples participating in Tomek links. This removes those negative examples that are believed borderline and/or noisy. All positive examples are retained. The resulting set is referred to as T.

Neighborhood Cleaning Rule for Under-sampling

This under-sampling technique combines the Condensed Nearest Neighbor (CNN) rule and the Edited Nearest Neighbor (ENN) rule to remove redundant and ambiguous examples from the majority class. The ENN rule removes examples that are misclassified according to its K-Nearest Neighbor classifier. These examples are those whose class labels differ from the labels of at least two of its nearest neighbors [18]. It also focuses more on cleaning the examples which are retained than on dropping examples from the majority class. The algorithm for this technique is shown below [15]:

1. Split data T into the class of interest C and the rest of the data O.
2. Identify noisy data A_1 in O with the edited nearest neighbor rule.
3. For each class C_i in O

   if ($x \in C_i$ in 3-nearest neighbors of misclassified $y \in C$)

and ($|C_i| \geq 0.5 \bullet |C|$) then $A_2 = \{ x \} \cup A_2$.

4. Reduced data $S = T - (A_1 \cup A_2)$

Over-sampling techniques

Over-sampling involves replicating examples in the minority class or creating new examples from already existing ones to rebalance the dataset. Random over-sampling is, however, not used here because it tends to fit the classifier too closely to the training data and hence, affecting its generalization performance [3]. Instead, Synthetic Minority Over-sampling Technique (SMOTE) is used in this project.

Synthetic Minority Over-sampling Technique (SMOTE)

The Synthetic Minority Over-sampling Technique synthetically generates examples from the minority class to balance the dataset. Synthetic here means that the examples are generated by creating or extrapolating new examples out of the existing ones. The algorithm for this technique is shown below [18]:

1. The total amount of over-sampling is defined as an integer value N. It is usually defined to obtain an approximate 1:1 class distribution
2. An iterative process of several steps is carried out:
   i) A positive instance is selected at random from the training set.
   ii) 5 (by default) of its K-Neighbors are obtained
   iii) N of these K instances are randomly chosen to compute the new instances by interpolation. The difference between the sample under consideration and each neighbor is taken and multiplied by a random number between 0 and 1. It is then added to the previous feature vector.

Hybrid Sampling

Hybrid sampling techniques combine both under-sampling and over-sampling techniques to balance a given dataset. They try to achieve an optimal balance by removing examples in the majority class and replicating some examples in the minority class. In doing this, it neither loses too much information from the under-sampling process nor does it overfit the classifier through the over-sampling process. The hybrid sampling techniques used in this project are SMOTE and Tomek Links under-sampling and SMOTE and Edited Nearest Neighbors under-sampling.

SMOTE and Tomek Links under-sampling

In this technique, SMOTE is first used to oversample the dataset then the pairs of nearest neighbors that have different classes (Tomek Links) are identified and removed. The advantage here is that Tomek Links are removed in both the majority and the minority class and hence, the clusters produced from the dataset are more defined [18] as compared to when the two techniques are used individually.

SMOTE and Edited Nearest Neighbors under-sampling

This technique is similar to the SMOTE and Tomek links under-sampling combination. However, after over-sampling the dataset using SMOTE, the Edited Nearest Neighbor rule performs a more in-depth data cleaning than the Tomek Links removal technique [18].

3.3.4 Data Splitting

The dataset is first randomized then split into two parts: 70% is used for training while the other 30% is for testing the classification algorithms. The training set was again split into

two parts; 70% used for training and 30% used as the validation set. The resampling techniques are applied to the training set before the models are trained.

3.4 Classification Algorithms

The problem of identifying fraud in credit card transactions is a classification problem, that is a problem with continuous independent variables and a categorical dependent variable. Hence, three classification algorithms are used in this project to classify transactions as fraudulent or genuine. They are Logistic Regression, K-Nearest Neighbour, and Naïve Bayes.

Logistic Regression

Logistic regression is a supervised learning algorithm that can be used for binary classification. It is used to predict the probability that a given input belongs to the class labeled 1(fraudulent transactions in the dataset used). This classifier models the data based on a non-linear function called the sigmoid function.

$$\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}} \qquad 3.2$$

where *e* is the base of the natural logarithm and *t* is the value you want to transform. The output value of this function is compared to a threshold value to determine whether the input given belongs to the fraudulent or genuine class.

Naïve Bayes

Naïve Bayes is a supervised non-linear algorithm that classifies input based on Bayes' Theorem. The theorem estimates the probability of a hypothesis given a particular instance, d. from the dataset. In the context of this project, the hypothesis is the class to assign the given credit card transaction to. The most probable hypothesis for that data instance D is selected using the formula below:

$$P(h_j|D) = \frac{\left(P(D|h_j) * P(h_j)\right)}{P(D)} \quad 3.3$$

where P(h|D) is the probability of a hypothesis h given a data instance D,

P(D|h) is the probability of a data instance D given the hypothesis h,

P(h) is the probability of the hypothesis h being true regardless of any data instance,

P(D) is the probability of the data instance regardless of a hypothesis

Naïve Bayes works with the assumption that the variables for a data instance are independent given the class and hence P(D|h) can be written as

$$P(D|h_j) = P(d_1|h_j) * P(d_2|h_j) * P(d_3|h_j) * \ldots * P(d_n|h_j) \quad 3.4$$

$P(D|\ h_j) = {}^{n}\Pi_{i=1}\ P(d_i|\ h_j)$ where i = 1,2,…,n and j = 1,2

The value n in the formula represents the maximum number of features for a data instance. The Bayesian classification rule used to classify the transactions as fraudulent or genuine is shown below:

Assuming that h_1 represents the hypothesis that the transaction is genuine and h_2 represents the hypothesis that it is fraudulent,

If $P(h_1|D) > P(h_2|D)$ then the data instance is classified as belonging to class h_1

If $P(h_1|D) < P(h_2|D)$ then the data instance is classified as belonging to class h_2

K-Nearest Neighbor

The K-Nearest Neighbor algorithm is a supervised machine learning algorithm that can be used for solving both classification and regression problems. It makes no assumptions about a given dataset and stores the entire dataset as its model representation. Hence, there is no training phase in the use of this algorithm.

Due to the sole reliance of the algorithm on the training data as its model, consistency in the data must be maintained to ensure that predictions made are accurate. For a new data

point, prediction of the class it belongs to is made by searching through the training dataset for the K most similar instances [19] and classifying the data point as part of that group. Distances such as the Euclidean distance, Hamming distance, Manhattan distance, and the Minkowski distance are used as measures to determine which group of K instances are most similar to the new data point. Euclidean distance is most suitable when the input variables used are similar in type while Manhattan distance can be used when these variables are not similar in type [19]. The value K determines the number of nearest neighbors the algorithm considers in making predictions for a new data instance. There is no particular way to determine this value, however, cross validation and algorithm tuning can be used as measures to find the optimal value for K.

Chapter 4: Results and Discussion

The results obtained from the classification algorithms and the resampling techniques used on the dataset are outlined below. The performance metrics used to evaluate the performance of these algorithms were precision, recall, F1 score and the precision-recall area under curve (PR AUC) score. These metrics were carefully chosen because of the imbalanced nature of the dataset; metrics like accuracy would have been very misleading due to reasons stated in section 1.5 of this paper.

Precision here is used to measure the number of fraudulent transactions that are correctly identified as being fraudulent out of the transactions predicted as being fraudulent by the model while Recall is used to measure the number of fraudulent transactions correctly identified as being fraudulent out of those that are actually fraudulent in the dataset. With respect to this project, maximizing precision will mean minimizing the number of transactions wrongly predicted as fraudulent while maximizing recall will mean minimizing the number of transactions predicted as genuine which are actually fraudulent. Since we want to do both, we take advantage of the f1 score as it combines both precision and recall in an equally weighted way. The precision-recall curve was also used due to the high imbalance in the dataset.

The plot below (Figure 4.1) shows the distribution of genuine to fraudulent transactions before resampling. Only the training set was used for this plot and hence, the number of transactions in the genuine class was 139,296 while the fraudulent class had 258 transactions.

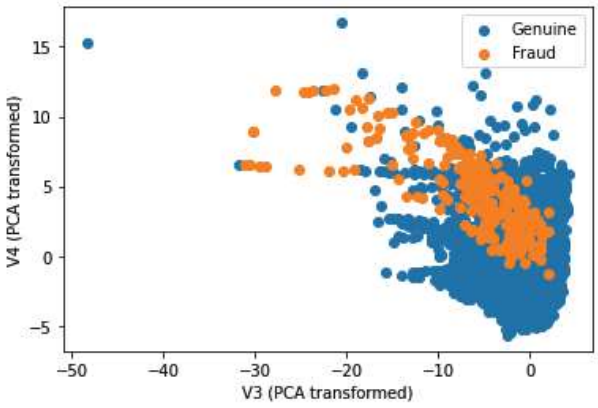


Figure 4.1: Distribution of genuine to fraudulent transactions before resampling

As shown above, the number of genuine transactions in the dataset greatly outweighs the number of fraudulent transactions. The minority to majority ratio is approximately 1:600 making it modestly imbalanced according to Haibo He and Yunqian Ma's categorization [12].

4.1 Results after Resampling

After resampling with five different techniques, the following distributions were obtained:

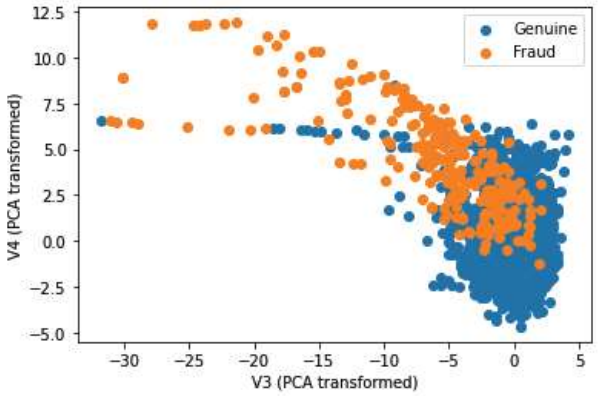


Figure 4.2: One-Sided Selection for undersampling

Through the One-Sided selection for undersampling technique, transactions in the genuine class were reduced to 4,962 while transactions in the fraudulent class remained at 258. Figure 4.2 shows the distribution after resampling using this technique

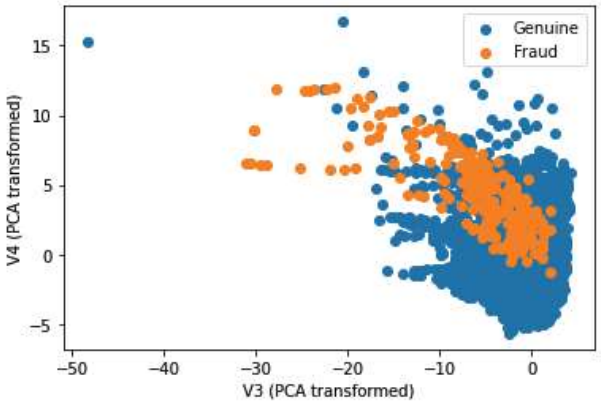


Figure 4.3: Neighborhood Cleaning Rule for resampling

The Neighborhood Cleaning Rule reduced the transactions in the genuine class from 139,296 to 139,109 and kept the fraudulent class at 258. The reason a large number of transactions (as compared to the One-Sided Selection for undersampling) was not removed from the genuine class was because of the technique's approach to removing redundant and ambiguous examples. As mentioned earlier, it focuses more on cleaning the examples which are retained than on dropping more examples from the majority class.

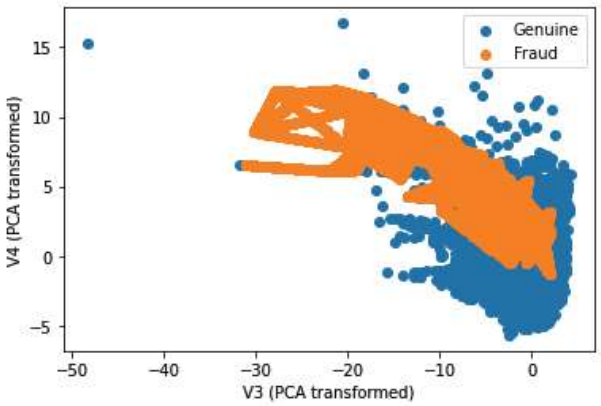


Figure 4.4: Synthetic Minority Over-Sampling Technique

After oversampling with the Synthetic Minority Over-Sampling Technique, 139,296 transactions were obtained for both the genuine and fraudulent classes. Figure 4.4 shows that the resampling technique synthetically generated an equal number of transactions for the minority class – the fraudulent class.

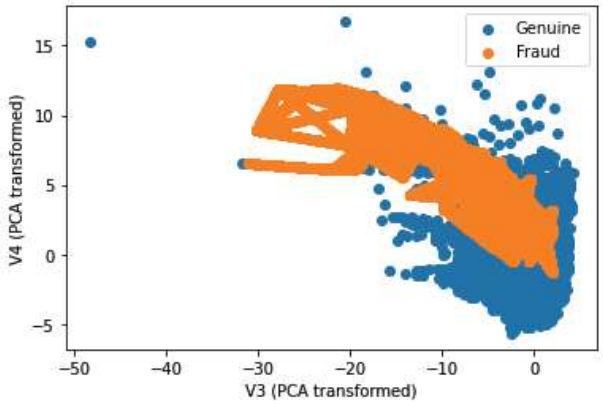


Figure 4.5: SMOTE and Tomek Links undersampling

The SMOTE and Tomek Links undersampling technique as shown in Figure 4.5 performed both undersampling and oversampling of the data. In the end, it gave the same number of transactions for both the genuine and fraudulent classes just as the Synthetic Minority Over-Sampling Technique did. 139,296 transactions for each.

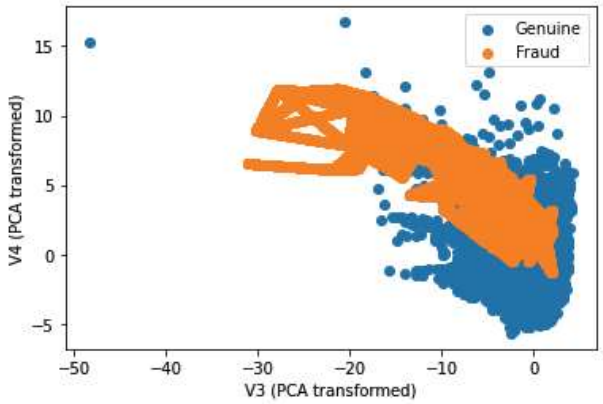


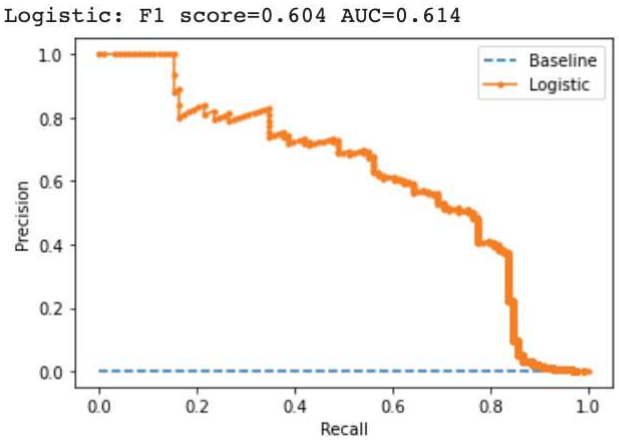
Figure 4.6: SMOTE and Edited Nearest Neighbors undersampling

Figure 4.6 shows a distribution of the genuine to fraudulent transactions after the second hybrid sampling technique was used, however, the difference between this technique and the SMOTE and Tomek Links undersampling is that this technique performs a more in-depth data cleaning on the majority class using the Edited Nearest Neighbor undersampling than the Tomek Links undersampling. This reduced the majority class to 139,039 and kept the minority class at 139,296.

4.2 Use of Classification algorithms on Resampled Data

4.2.1 Logistic Regression

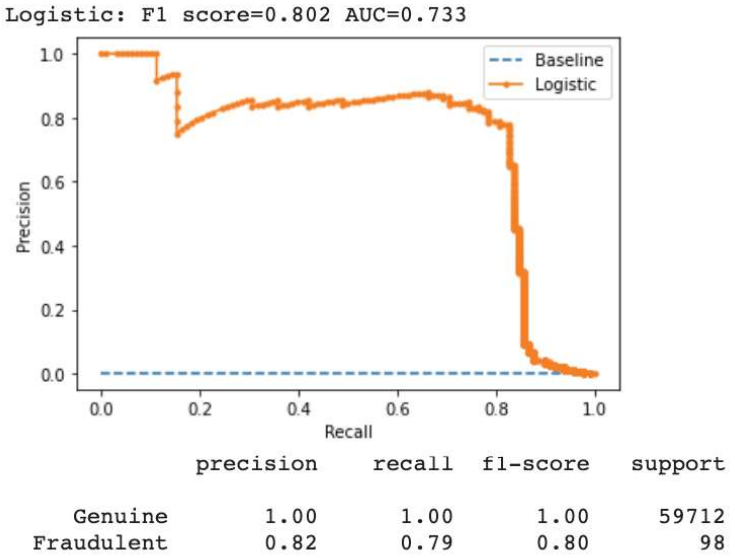
Logistic Regression was the first classification algorithm used on the resampled datasets. It performed best with the dataset that had been undersampled using the Neighborhood Cleaning Rule for undersampling, obtaining an f1 score of 0.802 and a PR AUC score of 0.733. Its worst performance was with the dataset resampled with the SMOTE and Edited Nearest Neighbors undersampling technique obtaining an f1 score of 0.093 and a PR AUC score of 0.782.



	precision	recall	f1-score	support
Genuine	1.00	1.00	1.00	59712
Fraudulent	0.54	0.69	0.60	98

Figure 4.7: Logistic Regression and One-Sided Selection for undersampling

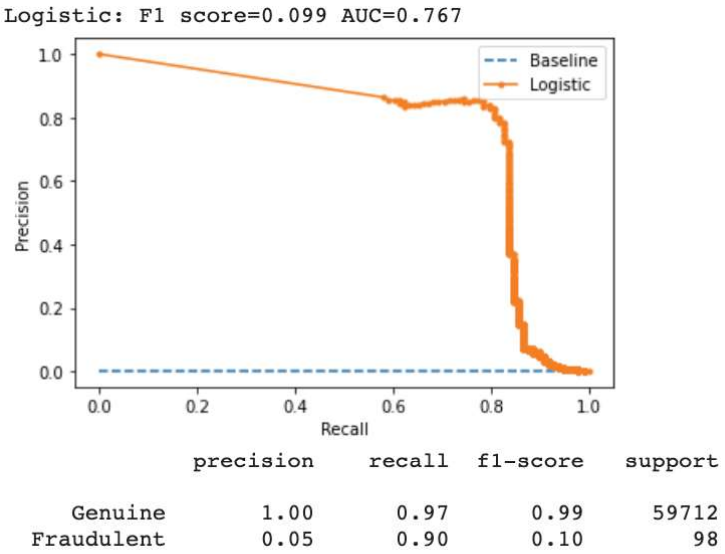
As shown above (Figure 4.7), the logistic regression model obtained a precision of 0.54, a recall of 0.69, and f1 score of 0.604, and a PR AUC score of 0.614 for the fraudulent class. This is not a very satisfactory performance.



	precision	recall	f1-score	support
Genuine	1.00	1.00	1.00	59712
Fraudulent	0.82	0.79	0.80	98

Figure 4.8: Logistic Regression and Neighborhood Cleaning Rule for undersampling

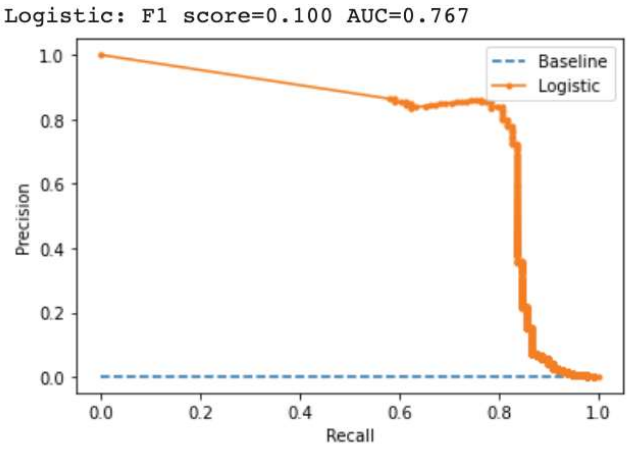
The Logistic Regression model performed best with this undersampling technique. It obtained a precision of 0.82, a recall of 0.79, an f1 score of 0.802, and a PR AUC score of 0.733. The precision, recall, f1 score, and PR AUC score were all higher than the previous undersampling technique used. However, the SMOTE technique obtained a higher recall and PR AUC score for the fraudulent class than this resampling technique. That notwithstanding, this resampling technique's performance with the logistic regression model is better than the SMOTE's performance since both precision and recall are essential for this project. Figure 4.8 depicts these results.



	precision	recall	f1-score	support
Genuine	1.00	0.97	0.99	59712
Fraudulent	0.05	0.90	0.10	98

Figure 4.9: Logistic Regression and Synthetic Minority Over-sampling technique (SMOTE)

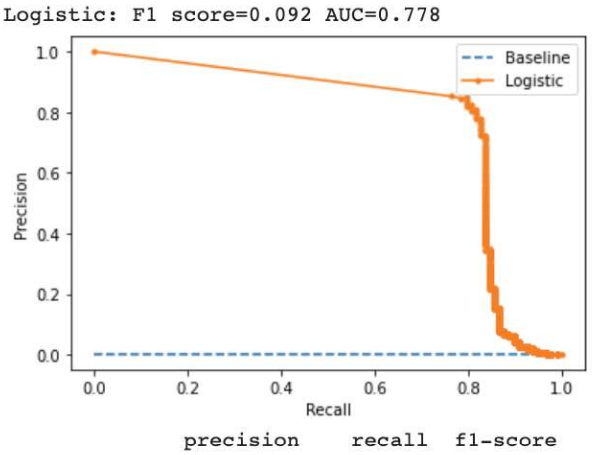
With a precision score of 0.05 and f1 score of 0.099 for the minority class, the SMOTE resampling technique's performance when the logistic regression model was used was also not satisfactory. In that, out of the transactions it predicts as being fraudulent, it is able to obtain a high percentage of them that are actually fraudulent. However, it is not able to detect a high percentage of fraudulent transactions from the total number of fraudulent transactions in the dataset. This is shown in Figure 4.9.



	precision	recall	f1-score	support
Genuine	1.00	0.97	0.99	59712
Fraudulent	0.05	0.90	0.10	98

Figure 4.10: Logistic Regression + SMOTE and Tomek Links undersampling

Similar to the SMOTE's performance, Figure 4.10 shows that this resampling technique obtained a precision of 0.05, recall of 0.90, f1 score of 0.100 and a PR AUC score of 0.767. This tells us that the oversampling done by SMOTE was dominant in the hybrid sampling process.



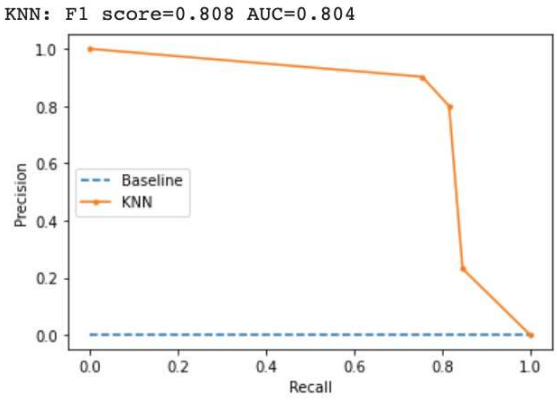
	precision	recall	f1-score	support
Genuine	1.00	0.97	0.99	59712
Fraudulent	0.05	0.90	0.09	98

Figure 4.11: Logistic Regression + SMOTE and Edited Nearest Neighbors undersampling

In Figure 4.11, the logistic regression model obtained the same precision and recall score as the previous hybrid resampling technique but a worse f1 score. This was surprising because the Edited Nearest Neighbors undersampling is said to perform a more in-depth data cleaning on the majority class than the Tomek Links undersampling technique.

4.2.2 K Nearest Neighbor (KNN)

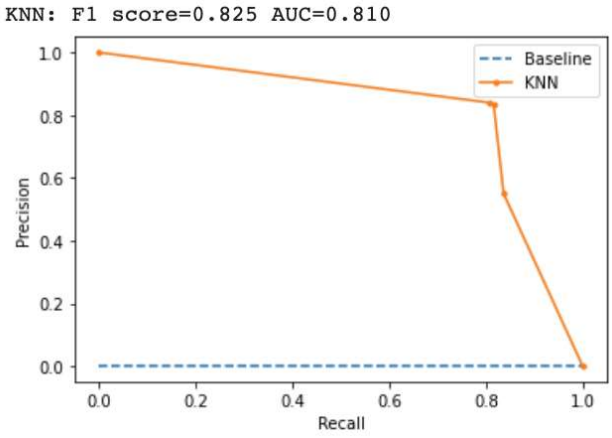
In general, the K Nearest Neighbor algorithm was the best performing algorithm out of the three algorithms used. Akin to the Logistic Regression model, it performed best with the Neighborhood Cleaning Rule for undersampling, obtaining an f1 score 0.825, a precision of 0.83, a recall of 0.82 and a PR AUC score of 0.810 for the fraudulent class. The number of neighbors used for this model was 3.



	precision	recall	f1-score	support
Genuine	1.00	1.00	1.00	59712
Fraudulent	0.80	0.82	0.81	98

Figure 4.12: K Nearest Neighbor and One-Sided Selection for undersampling

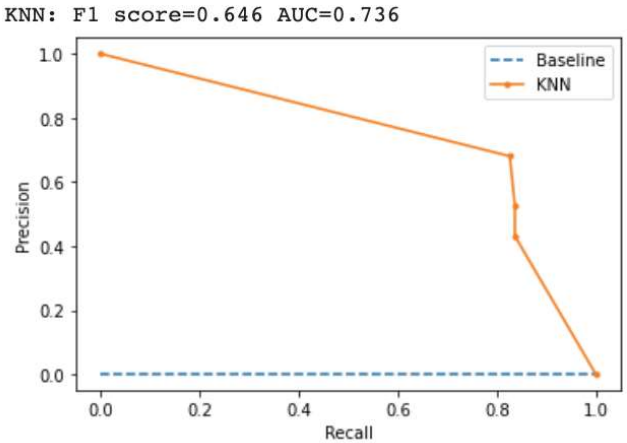
With an f1 score of 0.808, precision of 0.80, recall of 0.82 and PR AUC score of 0.804 for the fraudulent class, the KNN model performed better with the One-Sided Selection for undersampling technique as compared to the Logistic Regression model. It obtained a higher f1 score, PR AUC and recall than the Logistic Regression model's best scores (Figure 4.12).



	precision	recall	f1-score	support
Genuine	1.00	1.00	1.00	59712
Fraudulent	0.83	0.82	0.82	98

Figure 4.13: K Nearest Neighbor and Neighborhood Cleaning Rule for undersampling

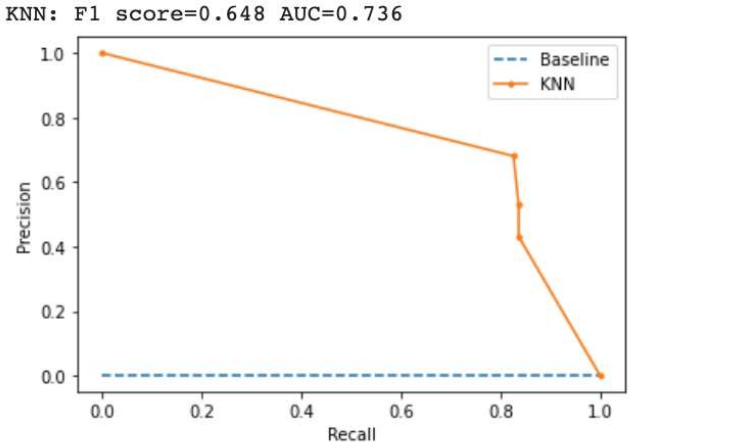
The KNN model obtained its best performance and the best performance out of all the models when it was used with this resampling technique. It obtained an f1 score of 0.825, PR AUC score of 0.810, precision of 0.83 and recall of 0.82 for the fraudulent class (Figure 4.13).



	precision	recall	f1-score	support
Genuine	1.00	1.00	1.00	59712
Fraudulent	0.53	0.84	0.65	98

Figure 4.14: K Nearest Neighbor and Synthetic Minority Over-sampling technique (SMOTE)

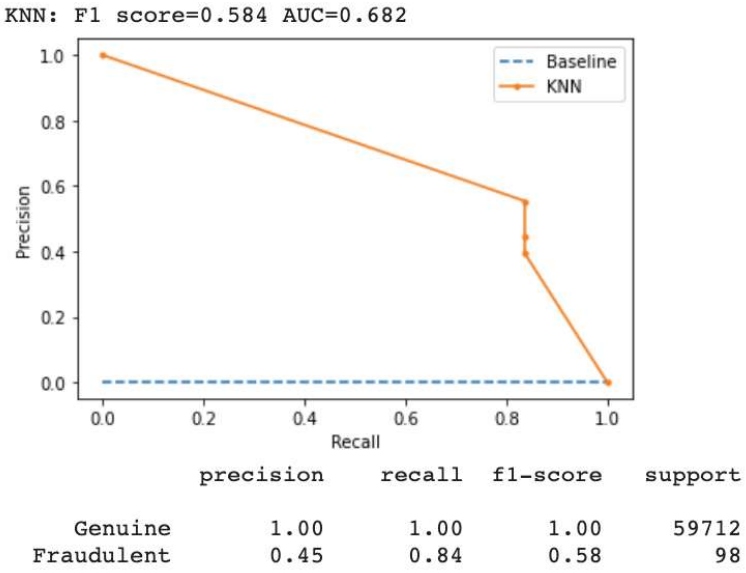
The performance of the KNN model on data that had been oversampled using the SMOTE technique was better than the Logistic Regression's performance although the scores obtained were not good enough. Figure 4.14 shows the performance of the KNN model on the oversampled data.



	precision	recall	f1-score	support
Genuine	1.00	1.00	1.00	59712
Fraudulent	0.53	0.84	0.65	98

Figure 4.15: K Nearest Neighbor + SMOTE and Tomek Links undersampling

Figure 4.15 shows that once again, the performance of the model with this hybrid sampling technique is similar to the model's performance with the SMOTE technique with the only difference being the f1 score. This was also the case for the Logistic Regression model.



	precision	recall	f1-score	support
Genuine	1.00	1.00	1.00	59712
Fraudulent	0.45	0.84	0.58	98

Figure 4.16: K Nearest Neighbor + SMOTE and Edited Nearest Neighbors undersampling

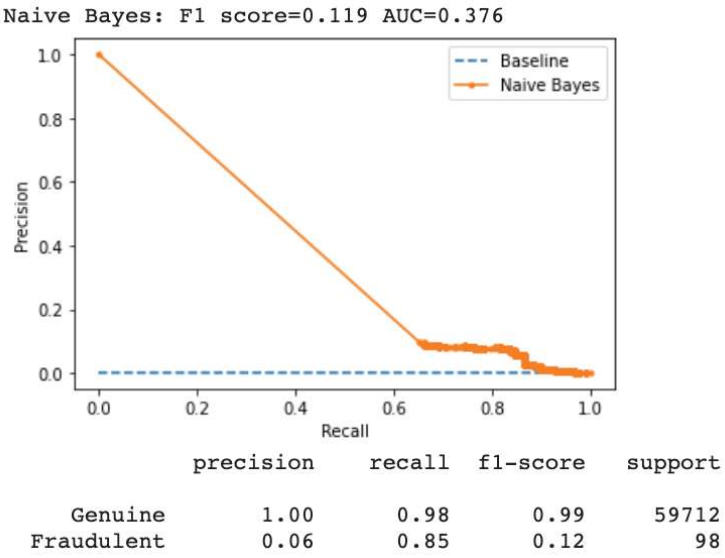
Again, the KNN model performed better than the Logistic regression model when the data was resampled using this hybrid sampling technique. The f1 score increased from 0.092 to 0.584 while the precision increased from 0.05 to 0.45 (Figure 4.16). This is a lower f1 score as compared to the performance of the model with the SMOTE and Tomek Links undersampling technique. This was also the case for the Logistic Regression model.

Although the performance was not good enough, it confirms that the KNN model performs better than the Logistic Regression model.

4.2.3 Naïve Bayes

The Naïve Bayes classifier was the worst performing algorithm. It performed best on the data that had been resampled using the One-Sided Selection for undersampling obtaining an f1 score of 0.119, a PR AUC score of 0. 376, a precision of 0.06, and a recall of 0.85 for the fraudulent class. This poor performance can be attributed to the lack of independence between the features in the dataset.

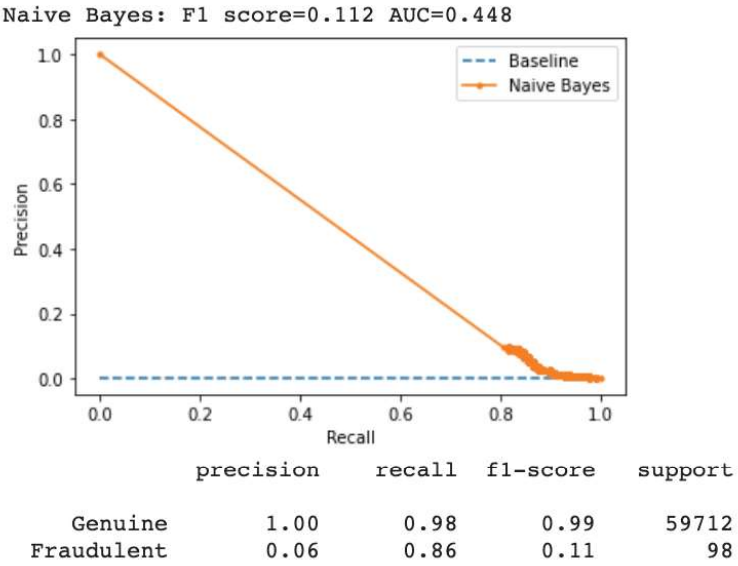
The Gaussian Naïve Bayes classifier was used because the features of the dataset are numeric.



	precision	recall	f1-score	support
Genuine	1.00	0.98	0.99	59712
Fraudulent	0.06	0.85	0.12	98

Figure 4.17: Naïve Bayes and One-Sided Selection for undersampling

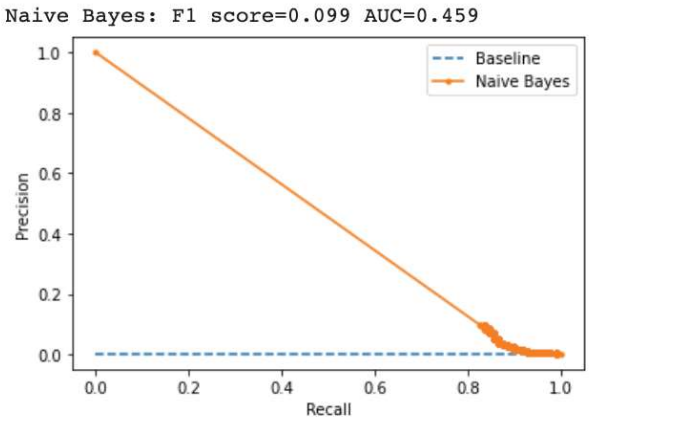
Although a very poor performance, this was the Naïve Bayes model's best performance. Figure 4.17 shows that the model obtained an f1 score of 0.119, PR AUC score of 0.376, precision of 0.06, and recall of 0.85 for the fraudulent class are worse scores than the worst performance of the KNN model. This automatically makes the model the worst performing model out of the three models used in this project.



	precision	recall	f1-score	support
Genuine	1.00	0.98	0.99	59712
Fraudulent	0.06	0.86	0.11	98

Figure 4.18: Naïve Bayes and Neighborhood Cleaning Rule for undersampling

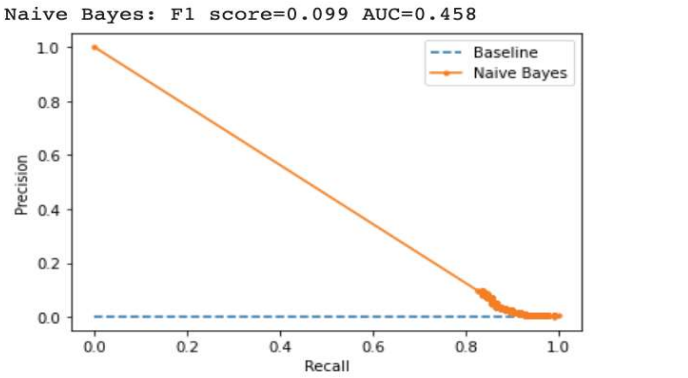
This was the only time when the undersampling technique used with an algorithm did not produce the best results. This time the results produced were second to the performance of the Naïve Bayes model on the data resampled using the One-Sided Selection for undersampling. The only difference between the performance of the model with this resampling technique and the One-Sided selection for undersampling is the lower f1 score.



	precision	recall	f1-score	support
Genuine	1.00	0.97	0.99	59712
Fraudulent	0.05	0.86	0.10	98

Figure 4.19: Naïve Bayes and Synthetic Minority Over-sampling technique (SMOTE)

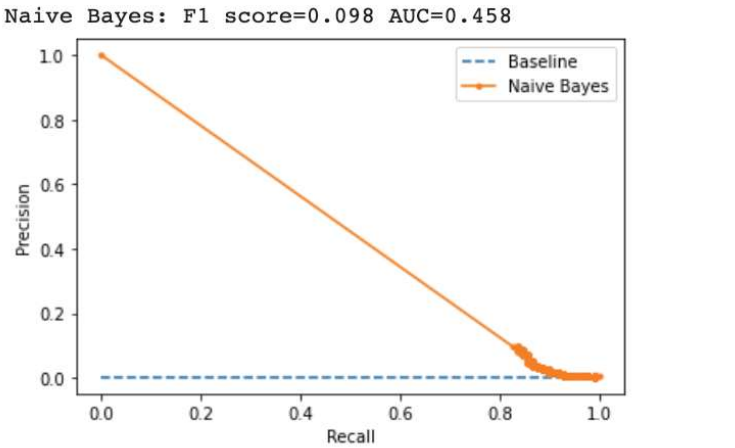
Figure 4.19 depicts that the Naïve Bayes model performance with the SMOTE resampling technique is similar to the Logistic regression model's performance with the same resampling technique. They both obtained an f1 score of 0.099 and a precision of 0.05 for the minority class. The logistic regression model however obtained a higher recall and PR AUC score than the Naïve Bayes model.



	precision	recall	f1-score	support
Genuine	1.00	0.97	0.99	59712
Fraudulent	0.05	0.86	0.10	98

Figure 4.20: Naïve Bayes + SMOTE and Tomek Links undersampling

As expected, and just like the KNN and Logistic regression models, the performance of the Naïve Bayes classifier on data that was resampled using this hybrid technique was similar to its performance on data that was oversampled using SMOTE. The only difference was with the PR AUC of 0.458 with the hybrid sampling approach and 0.459 with the SMOTE approach. Figure 4.20 shows the results obtained.



	precision	recall	f1-score	support
Genuine	1.00	0.97	0.99	59712
Fraudulent	0.05	0.86	0.10	98

Figure 4.21: Naïve Bayes + SMOTE and Edited Nearest Neighbors undersampling

The Naïve Bayes model used with this resampling technique gives a similar result to its performance on the SMOTE and Tomek Links undersampling resampled data. It gives an f1 score of 0.098, a PR AUC score of 0.458, precision of 0.05, and a recall of 0.86 for the fraudulent class here is the 0.001 reduction in the f1 score.

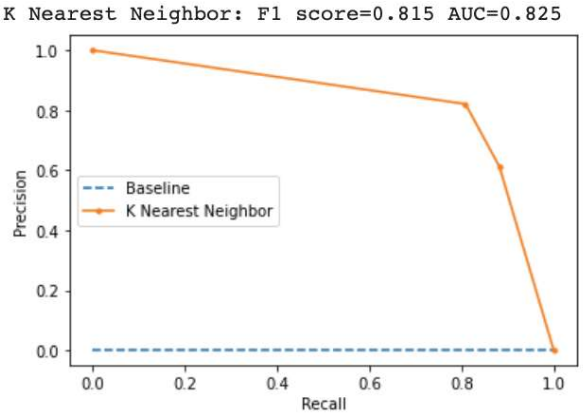
4.2.4 K Nearest Neighbor Classifier and Neighborhood Cleaning Rule for undersampling + Hyperparameter Tuning

The K Nearest Neighbor Classifier had the best performance when it was used on data that had been resampled using the Neighborhood Cleaning Rule for undersampling; it obtained an f1 score of 0.825, PR AUC score of 0.810, precision of 0.83, and recall of 0.82 for the fraudulent class. To try and improve the performance of this model, hyperparameter tuning

using random search was used. Initially, Grid search was the hyperparameter tuning algorithm that was used but due to its computational expense, Random search was used instead. After 10 iterations through a range of 1 to 16 possible nearest neighbors, it obtained a score of 0.9995981825958327 for the best KNN tuned model and an optimal number of two nearest neighbors.

Best params: {'weights': 'uniform', 'n_neighbors': 2}

The hyperparameter tuned model was tested on the test set to compare its performance with the baseline KNN model. The data resampled using the Neighborhood Cleaning Rule for undersampling was used for this test.



	precision	recall	f1-score	support
Genuine	1.00	1.00	1.00	85307
Fraudulent	0.82	0.81	0.81	136

Figure 4.22: Hyperparameter tuned KNN model + Neighborhood Cleaning Rule for undersampling

It is clear from Figure 4.22 that the hyperparameter tuned KNN model did not significantly improve the performance of the baseline KNN model. This suggests that the default parameters used by the KNN model are a better fit for the data resampled using the Neighborhood Cleaning Rule for undersampling.

Chapter 5: Conclusion and Further Work

This project focused on tackling the problem of imbalance in credit card fraud data by applying machine learning algorithms to data that had been resampled using Undersampling, Oversampling and Hybrid sampling techniques. The resampling techniques used under these three categories are techniques that employ an extensive approach in making decisions regarding which examples to add or remove from the dataset. The dataset that was used was the Credit Card Fraud Detection dataset provided by Kaggle. It contained 284,807 financial transactions conducted with credit cards, out of which 492 (0.172%) are labeled as fraudulent transactions making it moderately imbalanced according to Haibo He and Yunqian Ma [17].

The approach to tackling this problem involved analyzing the performance of three classification algorithms (Logistic Regression, K Nearest Neighbor Classifier and Naïve Bayes) when they were applied to credit card fraud data that had been resampling using One-Sided Selection for undersampling, Neighborhood Cleaning Rule for undersampling, Synthetic Minority Over-sampling technique (SMOTE), SMOTE and Tomek Links undersampling and SMOTE and Edited Nearest Neighbors undersampling. The results obtained from analyzing the performance of these algorithms suggest that the K Nearest Neighbor Classifier was the best performing model when it was applied in data that had been resampled using the Neighborhood Cleaning Rule for undersampling. Hyperparameter tuning of this model, however, did not significantly improve its performance.

The main limitation of this research has to do with how realistic undersampling credit card fraud data is in a real-world setting. Undersampling using the Neighborhood Cleaning Rule will definitely lead to a loss of some of the transactions from the genuine class although the algorithm is circumspect about which transaction to remove. This will mean that the organization will lose some of its client's information which I believe is a risk most organizations will not be willing to take.

In terms of future work, this project only focuses on one of the problems associated with credit card fraud detection and hence, will only be effective on the basis that the problem of imbalance in credit card fraud detection was the only problem in the detection of credit card fraud. This is, however, not the case. The dynamic nature of credit card fraudsters makes the occurrence a daunting task to tackle as they always come up with new ways to engage in fraudulent activities. A solution that incorporates the evolving nature of the occurrence will be an improvement to the approach employed in this project.

References

[1] Anita Carolina. 2012. ONLINE CREDIT CARD FRAUD: AN EMERGING CRIME IN THE INFORMATION TECHNOLOGY. *Jurnal InFestasi* 8, 2 (December 2012).

[2] Abdul Gaffar Khan. 2016. Electronic Commerce: A Study on Benefits and Challenges in an Emerging Economy. *Electronic Commerce* (2016), 5.

[3] eCommerce - worldwide | Statista Market Forecast. *Statista*. Retrieved October 14, 2020 from https://www.statista.com/outlook/243/100/ecommerce/worldwide

[4] Global retail e-commerce market size 2014-2023. *Statista*. Retrieved October 13, 2020 from https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/

[5] J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare. 2017. Credit card fraud detection using machine learning techniques: A comerrative analysis. In *2017 International Conference on Computing Networking and Informatics (ICCNI)*, 1–9. DOI:https://doi.org/10.1109/ICCNI.2017.8123782

[6] John Koetsier. COVID-19 Accelerated E-Commerce Growth '4 To 6 Years.' *Forbes*. Retrieved October 13, 2020 from https://www.forbes.com/sites/johnkoetsier/2020/06/12/covid-19-accelerated-e-commerce-growth-4-to-6-years/

[7] Marc Damez, Marie-Jeanne Lesot, and Adrien Revault d'Allonnes. 2012. Dynamic Credit-Card Fraud Profiling. In *The 9th International Conference on Modeling Decisions for Artificial Intelligence (MDAI)* (Lecture Notes in Computer Science), Springer, Girona, Catalonia, Spain, 234–245. DOI:https://doi.org/10.1007/978-3-642-34620-0_22

[8] Ronish Shakya. 2018. Application of Machine Learning Techniques in Credit Card Fraud Detection. University of Nevada. Retrieved from https://digitalscholarship.unlv.edu/thesesdissertations/3454/?utm_source=digitalschol

arship.unlv.edu%2Fthesesdissertations%2F3454&utm_medium=PDF&utm_campaign=PDFCoverPages

[9] VISA: quantity of credit cards in circulation 2020. *Statista*. Retrieved October 14, 2020 from https://www.statista.com/statistics/618115/number-of-visa-credit-cards-worldwide-by-region/

[10] World Business Report - The challenge of credit card fraud - BBC Sounds. Retrieved October 14, 2020 from https://www.bbc.co.uk/sounds/play/w172wx8tq61vjy2

[11] Yanmin Sun, Andrew K. C. Wong, and Mohamed S. Kamel. 2009. CLASSIFICATION OF IMBALANCED DATA: A REVIEW. *Int. J. Patt. Recogn. Artif. Intell.* 23, 04 (June 2009), 687–719. DOI:https://doi.org/10.1142/S0218001409007326

[12] Haibo He and Yunqian Ma (Eds.). 2013. *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons, Inc, Hoboken, New Jersey.

[13] Mr Rushi Longadge, Snehlata S Dongre, and Dr Latesh Malik. 2013. Class Imbalance Problem in Data Mining: Review. 2, 1 (2013), 6.

[14] S. Dhankhad, E. Mohammed, and B. Far. 2018. Supervised Machine Learning Algorithms for Credit Card Fraudulent Transaction Detection: A Comparative Study. In *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, 122–125. DOI:https://doi.org/10.1109/IRI.2018.00025

[15] Jason Brownlee. 2020. Undersampling Algorithms for Imbalanced Classification. *Machine Learning Mastery*. Retrieved November 17, 2020 from https://machinelearningmastery.com/undersampling-algorithms-for-imbalanced-classification/

[16] Jason Brownlee. 2020. SMOTE for Imbalanced Classification with Python. *Machine Learning Mastery*. Retrieved November 17, 2020 from

https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

[17] Haibo He and Yunqian Ma (Eds.). 2013. *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons, Inc, Hoboken, New Jersey.

[18] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. 2018. *Learning from Imbalanced Data Sets*. Springer International Publishing, Cham. DOI:https://doi.org/10.1007/978-3-319-98074-4

[19] Jason Brownlee. Master Machine Learning Algorithms. 163.