Fraud Classification in Imbalanced Big Data with Julia

By

Md. Tarikul Islam, ID: 011 151 013
Dipta Paul, ID: 011 143 008
Md. Ishtiak Hossain, ID: 011 152 056
Md. Asrafollah, ID: 011 142 143
Atikul Islam Sajib, ID: 011 142 099

Submitted in partial fulfilment of the requirements
of the degree of Bachelor of Science in Computer Science and Engineering

Thursday 10 October, 2019

Department of Computer Science and Engineering
United International University

Abstract

In this era of big data, classifying imbalanced real-life data in supervised learning is a challenging research issue. Standard data sampling methods: under-sampling, and over-sampling have several limitations for dealing with big data. Mostly, under-sampling approach removes data points from majority class instances and over-sampling approach engenders artificial minority class instances to make the data balanced. However, we may lose informative information instances using under-sampling approach, and under other conditions over-sampling approach causes overfitting problem. In this research work, we have presented a new cluster-based under-sampling approach by amalgamating ensemble learning (e.g. RandomForest classifier) for classification of imbalanced data that we implemented in Julia. We have collected actual illegal money transaction telecom fraud data, which is highly imbalanced with only 8,213 minority class instances among 63,62,620 instances. The proposed method bifurcates the data into majority class and minority class instances. Then, clusters the majority class instances into several clusters and considers a set of instances from each cluster to create several sub-balanced datasets. Finally, a number of classifiers are generated using these balances datasets and apply majority voting technique for classifying unknown new instances. We have tested the proposed method on separate test dataset that achieved 97% accuracy.

Acknowledgements

This work would have not been possible without the input and support of many people over the last two trimesters. We would like to express my gratitude to everyone who contributed to it in some way or other.

First, we would like to thank my academic advisors, Dr. Dewan Md. Farid, Associate Professor, Department of Computer Science and Engineering, United International University.

Our sincere gratitude goes to Abu Shafin Mohammad Mahdee. We are grateful to Mohammad Zoynul Abedin(Associate Professor, Department of Finance and Banking, Hajee Mohammad Danesh Science & Technology University and Ph.D. candidate at Dalian University of Technology) for providing us the dataset

We are also thankful to Dr. Chowdhury Mofizur Rahman(Professor and Vice Chancellor (In-Charge), Dr. Mohammad Nurul Huda (Professor and Coordinator - MSCSE), Dr. Hasan Sarwar (Professor), Dr Salekul Islam (Professor and Head of the Dept.), Dr. Khondoker Abdullah Al Mamun(Professor and Director - AIMS Lab), Dr. A.K.M. Muzahidul Islam (Professor), Dr. Swakkhar Shatabda(Associate Professor and Undergraduate Program Coordinator), Dr. Md. Saddam Hossain Mukta (Assistant Professor) and some other faculty members who effortlessly working to enlightening our Computer Science and Engineering department.

Last but not the least, We owe to our family including our parents for their unconditional love and immense emotional support.

Publication List

The work related to this research has been published or accepted in journals and conferences as mentioned in the following list:

Conference Papers

1. Md. Tarikul Islam, Md. Ishtiak Hossain, Dewan Md. Farid, Swakkhar Shatabda, Mohammad Zoynul Abedin, Mining Imbalanced Big Data with Julia, Juliacon2019, July 22-26, 2019, Baltimore, MD, USA.

Table of Contents

Table of Contents vi

List of Figures vii

List of Tables viii

List of Algorithms ix

1 Introduction 1
- 1.1 Project Overview . . . 1
- 1.2 Motivation . . . 2
- 1.3 Objectives . . . 3
- 1.4 Organization of the Report . . . 3

2 Background 4
- 2.1 Literature Review . . . 4
- 2.2 Existing Methodologies . . . 7
  - 2.2.1 Data Pre-processing: Sampling Technique . . . 7
  - 2.2.2 Algorithmic Approach . . . 9
- 2.3 Summary . . . 10

3 Project Design 11
- 3.1 Requirement Analysis . . . 11
  - 3.1.1 Purpose . . . 11
  - 3.1.2 Document conventions . . . 11
  - 3.1.3 Intended Audience and Reading Suggestions . . . 11
  - 3.1.4 Goals . . . 12
  - 3.1.5 Overall Description . . . 12
  - 3.1.6 Functional & nonfunctional Requirements . . . 13
- 3.2 Complex Engineering Decisions . . . 14
  - 3.2.1 Programming language . . . 14
  - 3.2.2 Development Tools: IDE . . . 18
  - 3.2.3 Document preparation tools . . . 20

3.3 Proposed Algorithm . . . 22
3.4 Licensing . . . 24
3.5 Summary . . . 24

4 Implementation and Results 26
4.1 Environment Setup . . . 26
4.2 Dataset Description . . . 26
4.2.1 Features and Class Description . . . 27
4.2.2 Dataset Preparation . . . 29
4.3 Performance Evaluation Criteria . . . 29
4.3.1 Confusion Matrix and Performance Metrics . . . 29
4.3.2 ROC Curve . . . 31
4.4 Experimental Results . . . 32
4.4.1 Proposed Hybrid Method . . . 32
4.4.2 AdaBoost . . . 33
4.4.3 Random Forest . . . 34
4.4.4 RUSBoost . . . 34
4.4.5 XGBoost . . . 35
4.4.6 GradientBoost . . . 35
4.4.7 Balanced Bagging . . . 35
4.4.8 SMOTEBoost . . . 36
4.5 Result Comparison . . . 37
4.6 Summary . . . 38

5 Standards, Impacts and Design Constraints 40
5.1 Compliance with the Standards . . . 40
5.1.1 Safety Standard . . . 40
5.1.2 Ethical Standard . . . 40
5.1.3 Technological Standard . . . 41
5.2 Impacts . . . 41
5.2.1 Economic Impact . . . 41
5.2.2 Social Impact . . . 41
5.2.3 Ethical Impact . . . 41
5.2.4 Health and Safety Impact . . . 42
5.2.5 Environmental Impact . . . 42
5.3 Design Constraints . . . 42
5.3.1 Manufacturability Constraint . . . 42
5.3.2 Political Constraint . . . 42
5.3.3 Sustainability Constraint . . . 43
5.4 Summary . . . 43

6 Conclusion **44**
6.1 Summary 44
6.2 Future Work 44

References **47**

List of Figures

1.1 An example of Imbalance dataset 2

2.1 Under-sampling the majority class instances 8
2.2 Over-sampling the minority class instances 9

3.1 Interconnection of API and the System 13
3.2 Simple micro-benchmarks showing Performance comparison of various language. Benchmark execution time comparative to C. (Execution time shorter is better, the performance of C = 1.0) [1]. 17
3.3 Work flow diagram of proposed Hybrid Model for fraud classification 24

4.1 A pie chart showing distribution of Non-Fraud and Fraud instances 27
4.2 A histogram showing frequency distribution of different transaction types. . 28
4.3 Majority and minority class distribution in original, balanced and test dataset. 30
4.4 A perfect model's ROC curve 32
4.5 ROC Curve of different algorithms 38
4.6 Comparison of the hybrid method with others 38

List of Tables

3.1 Programming Language options for Data Analysis. 14
3.2 IDE options for Julia. 18
3.3 Document preparation tool options. 20
3.4 License of selected tools. 24

4.1 Dataset Overview. 27
4.2 Class distribution of *isFraud*. 28
4.3 $D_{Majority}$ and $D_{Minority}$ Sub-datasets. 29
4.4 $D_{Minority}$ Sub-datasets for training and testing. 29
4.5 Distribution of generated clusters. 30
4.6 Majority and minority class distribution in $D_{Balanced}$ and D_{Test}. 30
4.7 Confusion matrix for Fraud Classification. 31
4.8 Confusion matrix for Fraud Classification. 33
4.9 Performance of proposed hybrid method on test set. 33
4.10 Performance of AdaBoost algorithm on test set. 33
4.11 Performance of Random Forest algorithm on test set. 34
4.12 Performance of RUSBoost algorithm on test set. 34
4.13 Performance of XGBoost algorithm on test set. 35
4.14 Performance of Gradient Boost algorithm on test set. 36
4.15 Performance of Balanced Bagging algorithm on test set. 36
4.16 Performance of SMOTE Boost algorithm on test set. 36
4.17 Mean performance result of different algorithm 37

5.1 Technological Standards. 41

List of Algorithms

3.1 Proposed Hybrid Algorithm 23

Chapter 1

Introduction

1.1 Project Overview

Fraud detection is one of the global challenges in the financial sector. Banking and other financial services, provided by organizations are leveraging the rapid development of information technologies more significantly than other industries. Therefore, in both the public and private sector, ensuring security has become a prominent issue, addressed by an emerging field of study called FinTech (Financial Technology) [2]. Fraudsters usually devise their intention in various ways, such as identity theft, phishing, and credit card information stealing, etc. Undoubtedly, a single fraud transaction, that could not be detected can drastically affect in person, an organization and even country level economically. Eventually, to prevent frauds and secure transactions, financial service institutions are investing at an increasing rate to reduce losses. Throughout the advances in data-mining, machine learning and big data analytics, many solutions for detecting frauds have already been proposed. Due to the key behavior of fraudulent events detecting frauds is still a big challenge. In banking, or any other financial services fraud transactions are rare event comparing to the legal transactions. Also, the volume of the data is vast. Moreover, it is highly dynamic in nature and concealed in heterogeneous individuals [3], [4], [5]. So, the class imbalance ratio is high and in fact, most cases it is incredibly high. These situations create the requirement of analyzing and building a robust model that can classify and detect frauds from highly class imbalanced and high dimensional big data. Most traditional machine learning (ML) algorithms used in big data analytics such as decision tree (DT) [6], Naive Bayes (NB) [6], and also k-nearest neighbors (KNN) stand-alone tries to boost classification accuracy [7], , [8] but fails to classify minority class instances accurately [9]. An imbalanced data-set is usually classified as intrinsic and extrinsic [9]. In the case of a data-set where imbalance nature is caused due to the nature of the data space, is called intrinsic. On the other hand, limitation of time and shortage of memory can also create imbalance, can be called extrinsic. The data-set for fraud detection in most cases are mainly intrinsic type.

Approaches towards, learning from imbalanced data commonly fall in two categories:

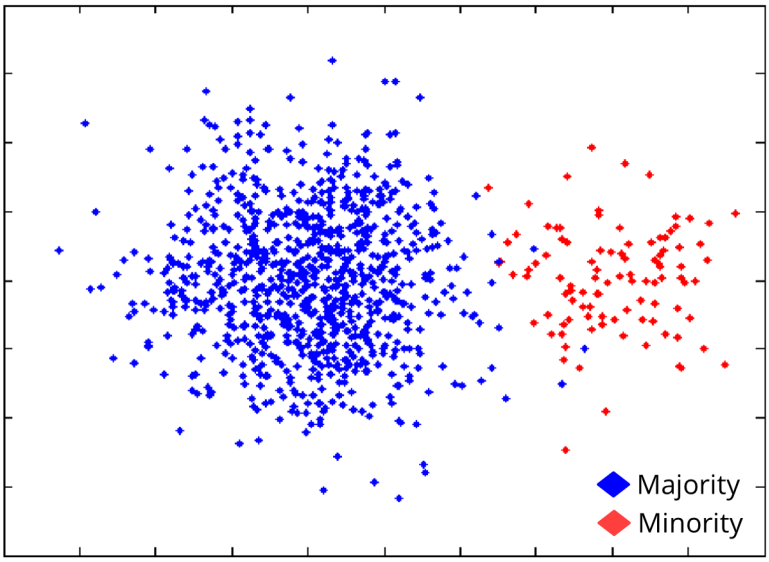

Figure 1.1: An example of Imbalance dataset

1) data balancing technique and 2) algorithmic technique [10] [9]. The data balancing technique is also referred to as an external method, which mainly provides a balanced data-set from an imbalanced data-set so that the existing algorithm can correctly classify both majority and minority instances. The algorithmic technique, also known as an internal method, works on modifying the traditional ML algorithms in such a way that reduces their sensitivity to class imbalance ratio while learning from imbalanced data.

In this project, a novel sampling strategy using clustering, to utilize the core of the majority instances is introduced, in addition to that, blending with ensemble learning is also proposed to detect fraud resolving data imbalance issue. This sampling method, cluster out only the majority instances into multiple clusters, and fused those clusters with minority samples to create multiple subset of the dataset. Finally, an ensemble of random forest is employed with majority voting in the final prediction at the end. In the distinct test set, we found accuracy of 97.22, with AUC of 96.91 and gmean of 96.78.

1.2 Motivation

Fraudulent activities cause significant loss financially that creates economic hardship. According to the Survey Report by the American Bankers Association(ABA), in 2016, the Banking industry faced approximately $2.2 billion loss in deposit accounts due to frauds [11]. Furthermore, unethical fraudulent activities also responsible for the damage to social status and reputation. Therefore it is an utmost concern to detect every fraud activity to ensure economic and social security.

Moreover, the dataset associated with fraud detection consists of a significant amount of non-frauds and insignificant amount of fraud. This imbalance situation makes it extremely difficult to discover invaluable information from fraud detection dataset.

Organizations like HSBC, World Bank, Amazon spent a large amount of money to detect fraudulent transactions. Hence it is very essential to find out invaluable information from these kinds of imbalanced data.

1.3 Objectives

The proposed project is a robust model that can identify any fraudulent activity and thus can classify a fraud in financial transaction. Though the primary goal is to build a machine learning model that takes any transaction information as input and outputs if the transaction initiator is fraud or not. There are also some other vital targets regarding the project. These associative but important targets are listed below.

- Building a faster model to detect fraud.
- The final model must be as accurate as possible, because any misclassification may harm the business.
- Preventing the stealing of money and valuable data from fraud.
- Ensuring the security and safety of the user's data.

1.4 Organization of the Report

The rest of the chapter of this book is organized as follows:

Chapter 2 provides background works which includes the literature review and existing methodologies.

Chapter 3 analyses the requirements of the project, discusses the complex engineering decisions and introduces with the proposed method.

Chapter 4 discusses the results and experimental analysis.

Chapter 5 describes about the standards that has been followed for this project, the impacts and the constrains of the project.

Chapter 6 presents the conclusions, summaries the thesis contributions, and discusses the future works.

Chapter 2

Background

This chapter discusses background works which include the literature review and existing methodologies.

2.1 Literature Review

One of the most significant difficulties in the banking sector is classifying fraudulent behavior in online banking services. Worldwide, this business is facing significant losses due to frauds. Therefore banks are striving to mitigate their losses resulting from these frauds. Most of the existing work in the domain of fraud detection is on credit card fraud, intrusion detection, and telecommunication fraud. Also, there are very few published papers on online banking fraud detection.

Most common fraud detection tools used by banks are rule-based [12]. Although rule-based algorithms can successfully identify frauds in transactions by matching historical patterns; it suffers from the lack of agility.

General fraud prevention mechanism developed for the credit card is CHIP&PIN. This mechanism fails most often to prevent fraudulent activities. As a result detecting fraud in credit card has been one of the hottest domain of research. An ensemble learning algorithm, combining popular supervised learning algorithms is proposed by Kültür et al. for detecting credit card fraud [13].A real transaction dataset acquired from a leading bank in Turkey is used in this study. Different voting mechanisms are used, and their evaluated performance is outstanding. Options are available for banks, to choose any of these voting mechanisms to achieve their desired false alarm rates.

Sam, Karl, and Bram used the real world data which has been provided by EPI, and they speculate on their paper, the problem of ascertaining fraudulent behavior in a credit card transaction system [14]. In this research, their focus was on two machine learning techniques for the credit card fraud detections problem. One is Artificial Neural Networks(ANN), and another one is Bayesian Belief Networks(BBNs). They have also stated that BBNs is better than ANNs. They have shown that BNNs takes less time to train and gives more accurate prediction whereas ANNs processes the prediction faster

but falls behind of BNNs.

Saad M. Darwish propose a two level intelligent system to detect fraudulent transaction from class imbalanced datasets [15]. This two level system is the amalgamation of k-means clustering and Artificial Bee Colony (ABC) Algorithm, which strengthen the classification rate of the system. A rule engine is an integral part of this system, which screens out whether a transaction authentic or a scam one based on features like, geolocations, frequency of usage, and balance etc. In this work, Semantic and decision-level fusion is used to combine both K-means and ABC algorithms that results in enriching accuracy rate with increased speed of detection convergence. In the training phase, the customers' transaction database is filtered through the rule engine, which generates the critical values of the transaction that finally used as a hint for two-level classifier to identify the customer as fraud or genuine.

Data related to fraud detection is highly imbalanced; predicting fraud in organizations is a challenging task. The reason for the imbalance is the number of legal transactions is very high than fraudulent examples. It results in, predicting majority class to be authentic, achieving high accuracy rate. To overcome this problem of class-imbalanced data, oversampling methods inflate the minority class by re-producing synthetic examples [16].

These techniques focus on the behavior of the rare class instances and use them in the Oversampling process. Sharma proposed a novel approach for synthetic oversampling that uses the fruitful information in the majority class to synthesize rare class data [17]. Measuring Mehalbonis distance from the majority class as the known instances, synthetic data is produced. This method achieved state-of-the-art performance improvement by evaluating 26 benchmark data-sets, over the available oversampling techniques.

In binary class imbalanced problems, a general way to determine classification accuracy is by calculating the geometric mean of the true positive and true negative rates of the classifier. Ludmila I. Kuncheva and others, prove that instance selection can improve geometric mean and also, provide theoretical basement for such amelioration [18]. It is manifested that, in terms of the amount of retained examples, geometric mean is non-monotonic and discourage picking instances systematically. Here in this work, they also show that, straightforward maximization of geometric mean is better choice rather than creating equal distribution of positive and negative class in imbalanced dataset. They studied 12 instance selection techniques in 66 benchmark dataset with two fold stratified cross-validation repeating five times. Though 10-fold cross validation is usually used by default, they use 2-fold, because, creating many folds results in very imbalanced class ratio in each fold. For classification, Nearest Neighbor and Begging Nearest Neighbor is used and finally it is concluded that, randomize instance selection is better approach than controlled instance selection in terms of geometric mean.

Phua, Alahakoon, and Lee introduced Stacking-Bagging [16], an innovative ensemble fraud detection method which is built on subsisting fraud detection research and minority report. They used Naive Bayesian (NB), Backpropagation (BP) and C4.5 algorithms to

train data partitions obtained from a fraud detection data set in automobile insurance. Authors generated the data partitions by doing oversampling with the replacement of minority class instances. They showed that stacking NB, BP and C4.5 classifiers and bagging their predictions achieve the maximum cost savings. In their research, they considered a small dataset of only 11,338 instances. Moreover, minority oversampling with replacement might have resulted in internal overfitting.

Ensemble methods over conventional methods for class imbalanced methods shows significant performance in terms of keeping useful information and predicting minority class without overfitting problems. A novel ensemble method is proposed, which transform the imbalanced dataset into multiple balanced sets by using different balancing techniques, such as random splitting and clustering algorithms. Therefore, it employs multiple classifiers on each balanced dataset, such as Naïve Bayes, c4.5, Random Forest etc. Finally, different ensemble rules are studied in this proposed method like, Max, Min, Product, Maj distance. Along with these ensemble rules they proposes five rules, which are, Max Distance, Din Distance, Maj Distance, and Sum Distance etc. In this work, they evaluated their algorithm with 46 datasets, and compared with other existing methods.

Ensemble methods over conventional methods for class imbalanced methods show significant performance in terms of keeping useful information and predicting minority class without overfitting problems. Thus, A novel ensemble method is proposed by Zhongbin Sun, which transforms the original imbalanced dataset into multiple balanced sets by using different balancing techniques, such as random splitting and clustering algorithms. Therefore, it employs multiple classifiers on each balanced dataset, such as Naïve Bayes, c4.5, Random Forest, etc. Finally, different ensemble rules are studied in this proposed method like Max, Min, Product, Maj distance. Along with these ensemble rules, they propose five rules, which are, Max Distance, Din Distance, Maj Distance, and Sum Distance, etc. In this work, they evaluated their algorithm with 46 datasets and compared with other existing methods [19].

A work based on cluster-based undersampling is proposed by Show-Jane Yen, Yue-Shi Lee [20]. They introduced six methods for making the dataset balance. All of the six methods initially makes cluster from the original dataset, and each cluster may contain minority and majority class instances. And then from those cluster majority class instances is under-sampled using six heuristics for six different methods. Lastly, a combination of all the majority class instances and the under-sampled minority class instances make a balanced dataset. They have shown that their proposed method SBC outperforms Random Undersampling, NearMiss-2, and their other five proposed algorithm. One of the significant limitations of their proposed algorithm SBC is the internal calculation of the number of majority class instances to be selected randomly from each cluster. Other five algorithm suffers from calculating the nearest or farthest neighbor of majority instance and minority instance. Moreover, they did not state what to be done if a cluster only contains the minority class.

Another research work pointed out that, the essence of online fraud reflects the synthetic

abuse of interaction between resources in three worlds: the fraudster's intelligence abuse in the social world, the violence of web technology and Internet banking resources in the cyber world, and the abuse of trading tools and resources in the physical world. Online banking fraud involves multiple resources, including human wisdom, computing tools, web technology, and online business systems: the instant and effective detection of such fraud challenges existing fraud detection techniques and methods [5]. Considering all these situations wei introduced a systematic online banking fraud detection approach in their research. It includes contrast pattern mining, neural network and decision forest, and their individual outputs are integrated with an overall score measuring the risk of an online transaction being illegal or legal. They tested the approach by incorporating the system in a banking system. Moreover, they claimed that their system could perform well with any other existing banking system.

2.2 Existing Methodologies

There is basically two way to work with imbalance dataset. One way is to make the dataset balance by applying some pre-processing technique like oversampling or undersampling and then apply some tradition algorithm like a decision tree to train a model. Another way is to improve the classification algorithm and considering the dataset as it is. In this section, we have discussed some of the existing data prepossessing technique and some existing classification algorithm to handle imbalance big data. We start by displaying an overview of the Sampling technique, followed by a description of each sampling type and also some report on some other method.

2.2.1 Data Pre-processing: Sampling Technique

In machine learning, the issue of bias towards the majority class for data imbalance problems harms the classification performance. Thus it is important to make the dataset balanced before the training process. The most frequently used data pre-processing based balancing technique is sampling. Sampling is the process of rearranging the data sample from the actual datasets. In terms of machine learning, It can be done either by under-sampling the instances of the majority or over-sampling the instances of the minority class or doing a mix of both under-sampling and over-sampling. In terms of statistics, it can be done either by non-statistical estimation or statistical estimation. In the non-statistical evaluation, we randomly draw examples from the real dataset. Statistical estimation involves estimating the parameters of the actual dataset and then bringing the sub-samples. Thus data samples can be created which include all data information from the real dataset. This sampling technique helps to create a balanced sample dataset when the actual data is highly imbalanced.

2.2.1.1 Under-sampling

One of the most common strategies to transform an imbalanced dataset to a balanced dataset is to under-sample the majority class. It is done by excluding some instances of the majority class in various conditions and thresholds. Some of the frequently used under-sampling strategies are as follows:

1. **Random Under-sampling:** One of the most straightforward under sampling strategy is random under-sampling (RUS). It is done by eliminating majority class instances randomly and uniformly. This strategy yields a better result when most of the instances of the majority class are almost similar. Otherwise, it is not a good option for data balancing.

2. **Cluster-based Undersampling:** The cluster-based under-sampling procedure solves the class imbalanced problem by creating multiple clusters from the majority class sub-datasets and then excluding elements from each cluster. The exclusion is done on different requirements. Cluster-based under-sampling can be done either by excluding the instances that are far from the cluster centroids or the instances that are in the boundary line of the clusters.

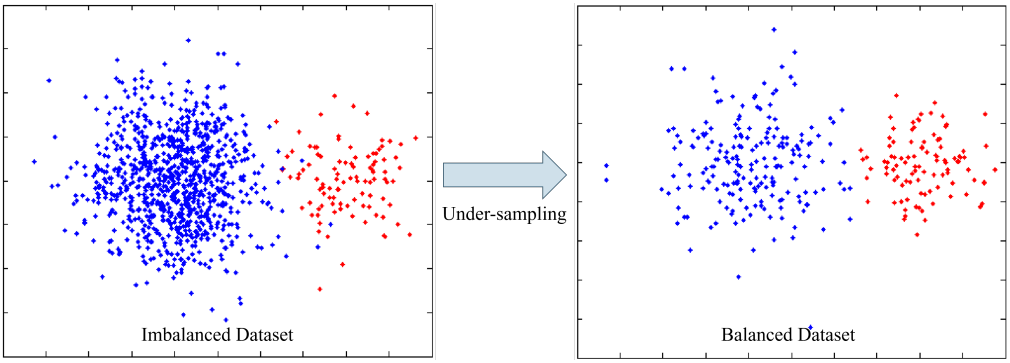

Figure 2.1: Under-sampling the majority class instances

Other undersampling variants are available which are based on two types of noise model. One noise model is assumed that samples near the boundary are noise. In another model, it is expected that the majority of class samples gathered in the same area as minority class samples are noise. Discarding these samples from the data creates a clear boundary that can assist in classification.

The main drawback of doing undersampling is it causes data loss which may be essential for classification.

2.2.1.2 Oversampling

In an imbalanced dataset, Oversampling increases the underrepresented minority class instances until the dataset becomes balanced. Some of the well known oversampling

techniques are as follows.

1. **Random Oversampling:** Random oversampling is a strategy that converts an imbalanced dataset into a balanced dataset by choosing minority class samples randomly and then duplicating them. It might produce good results when the class imbalanced ratio is small.

2. **Cluster-based Oversampling:** In cluster based over-sampling a clustering algorithm is used to create clusters for both minority class and majority class independently. Then using these new cluster new instances is produced until instances of both the classes becomes equal.

3. **SMOTE:** Synthetic Minority Over-sampling Technique simply known as SMOTE creates artificial synthetic samples by randomly sampling the characteristics from occurrences in the minority class. There are also some other variants of SMOTE like MSMOTE, SVM-SMOTE, KMeansSMOTE, SMOTE-NC etc.

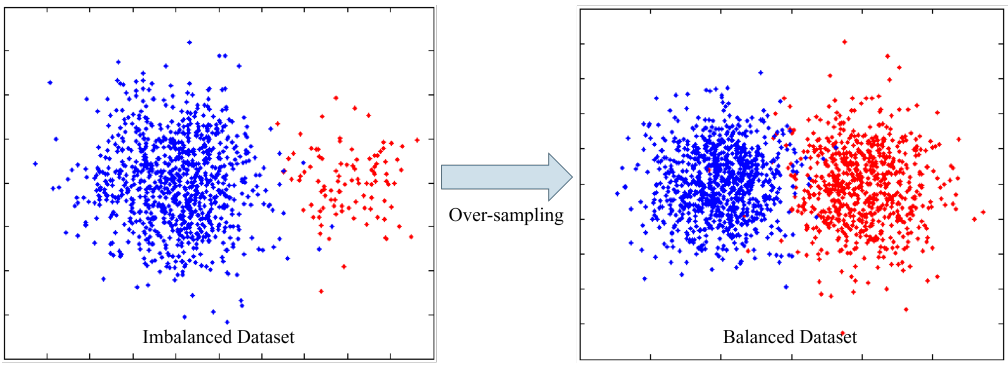

Figure 2.2: Over-sampling the minority class instances

The problem with oversampling is it causes Over-fitting problem. It takes place when we over-sample the minority class before cross-validation.

2.2.2 Algorithmic Approach

2.2.2.1 Bagging

Bagging is a simple and very powerful ensemble method. It is the application of the Bootstrap procedure to a high-variance machine learning algorithm, typically decision trees. When we consider bagging with decision trees, we are less concerned about separate trees overfitting the training dataset. For this reason and efficiency purpose, the separate decision trees are grown deep, and the trees are not pruned. These trees will have both low bias and high variance. These are important components of sub-models when combining predictions using bagging. This can be selected by increasing the number of trees on the run after run until the accuracy begins to stop showing improvement.

2.2.2.2 Boosting

To apply the Bagging, Bootstrap concept is needed. Bootstrap method refers to random sampling with replacement[9]. Bootstrap helps to understand the biases, variance with the big data. Bootstrap makes a random sampling of a small subset of data from the large dataset. This small subset can be replaced. The selection of all the instances in the dataset has equal probability. This technique can help to understand the mean and standard deviation from the dataset adequately. Boosting is a group of algorithms that utilize weighted averages to make weak learners into stronger learners. In boosting each model run independently and group the outputs at the end without partiality to any model, this reason differs boosting from bagging. Each model that runs in boosting, delivers what features the next model will focus on.

Bagging and Boosting reduce the variance of a single estimate as they combine various estimates from several models. Therefore, the result may be a model with a very higher stability.

2.3 Summary

Most popular choices to work with imbalanced dataset are primarily, sampling strategy and designing algorithm using bagging or boosting approach. In sampling strategy, undersampling the majority class is often a wise choice, as oversampling the minority instances may lead to the overfitting problem. Bagging is done bootstrap aggregation of the dataset. Random Forest is a popular algorithm which uses the bagging technique. Boosting algorithms are also effective in terms of class imbalance problems. It re-adjust weights at each iteration, to focus on the misclassified instances of the previous iteration to the next iteration. Adaboost is one of the most popular boosting algorithm.

Chapter 3

Project Design

In this chapter, we analyze the requirement specification of the project and discuss the complex engineering decisions.

3.1 Requirement Analysis

A requirement analysis describes a system and gives an outline to develop it. This requirement analysis describes an API for fraud detection. Moreover, this requirement analysis contains all the instruction of developing the API which includes the purpose of the project, target user of this project, the functionality of this project and other requirements.

3.1.1 Purpose

The purpose of this requirement analysis is to describe the specification of the project which is an API named as "Fraud Detection API v1.0". It will give a clear description of the functions of the project, all functional and nonfunctional requirements of the project, how the project will operate, and the user of the project and how the user will interact with the project. The intended user of this document is its stakeholders and developers of the project.

3.1.2 Document conventions

The entire document is written using Latex. Moreover, the entire font and other settings are the default of latex, though in some instances some package of latex has been used. An IEE format has been followed for documentation of this requirement analysis.

3.1.3 Intended Audience and Reading Suggestions

This document contains some technical term, and the customer and the audience should understand the terms. Moreover, the intended audience who will be reading this document is the audience like developers, project managers, users, testers and documentation writers.

This requirement analysis should be read starting with Introduction. This document is intended for:
Developers: Requirements provided in this document ensures the developer about the fact that they are developing the right project.
Testers: In order to have an accurate list of the functions and features of the system that has to reply according to requirements and to provided diagrams.
Users: To understand the idea of the project and get familiar with the project and suggest other new features that would make it more functionally better.
Document Writers: In order to know what features have to put and in what way they need to be explained. And also to know what security technologies are required and on each users action how the system will response etc.
Advanced end users and system administrators: To know their expectation from the system, what are the right inputs and corresponding outputs and what is the response from the system in error situations.

3.1.4 Goals

Though the primary goal is to build a "Fraud Detection API" which is a machine learning model that takes a transaction information as input and produce output if the transaction initiator is fraud or not, there are also some other important goal of the project. These associative but essential goals are listed below.

- Building a faster model to detect fraud.
- The model needs to be as accurate as possible, because any misclassification may harm the business.
- Preventing stealing of money and valuable data from fraud.
- Ensuring the security and safety of the user's data.

3.1.5 Overall Description

3.1.5.1 Product Perspective

Fraud Detector API 1.0 is a new Application Protocol Interface for detecting Fraud. It will be a subsystem of any larger financial transaction System like the payment system. More specifically as shown in Figure 3.1 it will operate as middleware in-between a larger system.

3.1.5.2 Product Functions

As the user performs any transaction the main system (Part A of the system) will send some information to the middleware and then the middleware with the API will generate a prediction of whether the transaction initiator is fraud or not. And will send an alarm to the main system (Part B of the system).

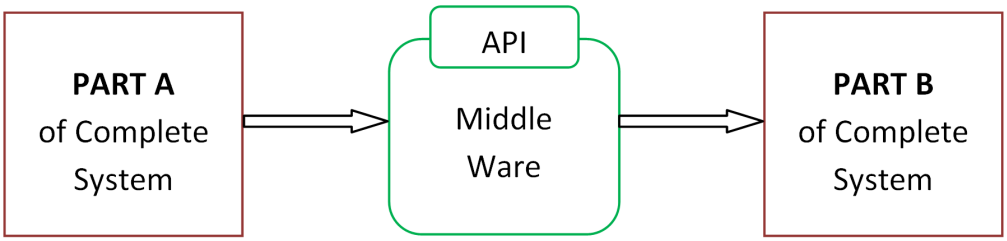

Figure 3.1: Interconnection of API and the System

3.1.5.3 User Classes and Characteristics

This type of systems should be used by any type of financial transaction system as the payment system.

3.1.5.4 Operating Environment

It will be a server-side web API. Thus it can be used by any web-based transaction system. It can be accessed via a web-based transaction system by a browser.

3.1.6 Functional & nonfunctional Requirements

Functional Requirements includes Software Interfaces and Communications Interfaces. On the other hand nonfunctional requirements includes performance, safety and security issues.

3.1.6.1 Performance Requirements

Considering the project, it is based on the client-server architecture where the client will forward a request to the end server and the end server will serve the request. Moreover, at any time as the number of client raise thus the number of the request will rise. Therefore to serve the entire client request server must have the hardware, the software, and the networking capability. The network architecture should be strong enough so that the time between request and the response should be as minimum as possible. Also, the network should be scalable to guarantee the number of clients can be increased as needed.

3.1.6.2 Safety Requirements

One of the safety requirements is power supply should be uninterrupted. The networking devices should be properly connected. And if there are faulty networking devices (such as a router, switch and the hub, etc.), then they should be replaced/ removed as early as possible.

3.1.6.3 Security Requirements

Database access should be restricted for the people who are not required to view the information about users.

3.1.6.4 Software Interfaces

The system makes indirect use of an internet browser along with the internet connection. Must support Operating System: Windows XP/7/8/10.

3.1.6.5 Communications Interfaces

An internet connection is used by the system to connect to the database. The system will use the HTTP protocol as a communication interface to transfer data of financial information between subsystems and API. As financial data is essential thus, it also requires excellent concern to ensure the security of these data.

3.2 Complex Engineering Decisions

It is always necessary to select the right tool for the job. A great combination of the right software tool, right hardware tool and right integration between them eventually leads to a successful project. This project does not require any hardware tool. Software tools for this project include the programming language, development tools such as IDE and document preparation tools. Throughout this section the advantages and disadvantages of some tools, a comparison between these tools and the final decision of the tools will be discussed.

3.2.1 Programming language

There is numerous amount of programming language is available for data analysis and data manipulation work. However, we are considering only the followings as options.

SI	Options
1	Julia
2	Python
3	R

Table 3.1: Programming Language options for Data Analysis.

- **Julia**: Julia is a high-performance and high-level dynamic programming language for numerical computing [1]. It was born in 2012. The language is quickly adopted in the financial field. In addition, the number of packages offered by Julia is increasing rapidly because of the contribution of the Julia developer community.

License:

MIT License and GNU GPL v2;

Advantages

- A program written in Julia is fast. It reaches the speed of C++ and Fortran and other compiled languages. This is important for scientific computing, where the audience might not want to iterate many times on the code base, but simply get a result once on a problem as fast as possible.
- As a high-level language, Julia comes with the dynamism of Python or Ruby, and it also provides the mathematical functions and notations of Matlab or Octave.
- Julia has excellent support for macros that let us manipulate code like data.
- Parallel computing is also enabled in Julia.
- Julia provides the possibility to use the packages of C, Fortran, and Python. Thus it is easy to run existing code in Julia.
- In Julia, there are some packages with GPU-support.

Disadvantages

- Julia has smaller community support, though the community around Julia is enthusiastic and rapidly growing.
- It also has fewer packages or libraries of its own as the language is still developing.
- A few bugs occur when doing data manipulation.

- **Python**: Guido van Rossum introduced Python in 1991 [21]. Since then, it has become an extremely popular general purpose language that is widely used in the data processing. These days it is widely used in Scientific computing [22].

License:

BSD License

Advantages

- Python has huge collection of packages and libraries from web development, through game development, to machine learning. Some of the well known packages are NumPy, Skit-learn etc.

- Python focuses on readability of code. The language is neat, versatile, well-structured, easy to use and easy to learn.
- Python is open source with a large active community.
- This language supports both the procedural and object-oriented programming paradigms. It is also Embedded.

Disadvantages

- Python is an interpreted language; thus a possible disadvantage of Python is its slow speed of execution.
- Python is one of the top choices for the computation of many desktop and server platforms, but for mobile computing, it is considered as a weak language.
- Python is not suitable for parallel computing (multi-processor/multi-core work).

- **R**: R is an open source environment and programming language for statistical computing [23], data analysis and graphics [24] which was first introduced by R. Ihaka and R. Gentleman. It is more used by academics.

License:

GNU GPL v2

Advantages

- R has a great range of useful statistical and graphics packages. Such as dplyr, a package for data manipulation [25] and ggplot2, a graphics package for data analysis [26].
- R language offers a breadth of techniques which are used for data analysis, sampling, and visualization. It has more advanced tools for analysis of statistical data.
- As it is open source anyone can contribute, community support is getting better day by day.

Disadvantages

- R is an interpreted programming language, for this reason in some cases program written in R shows slow execution time than some other languages.
- Though R has some quality packages for statistical usage, but quality of some packages is less than perfect.
- There are almost too many packages, many doing the same things.

- The naming of some package and function can be bewildering. As an example "read_csv" is a reader function that is used to import data. On the other hand there is a import function named "read.csv" in Base R. These two function acts quite differently.
- R is Memory intensive language.

Comparison and Decision

Many languages offer great data analysis tools and other facilities. Now comparing between Python, R and Julia we find that Julia is much faster like C than the other two as shown in Figure 3.2. Three of them are almost the same syntactically. Python and R, both of them offer great packages for data analysis. On the other hand, Julia has lack of packages comparing to both Python and R. As a language Julia offers the general usage of Python and the statistical ease provided by R. Combing these useful elements makes Julia a powerful language for statistical data analysis. The project fraud detection in high dimensional big data demands faster and accurate prediction. Thus in this case fastness in the algorithm and programming language is essential. So we choose Julia as the primary programming language because it is faster and we are considering the lack of package in Julia as a challenge rather than a problem.

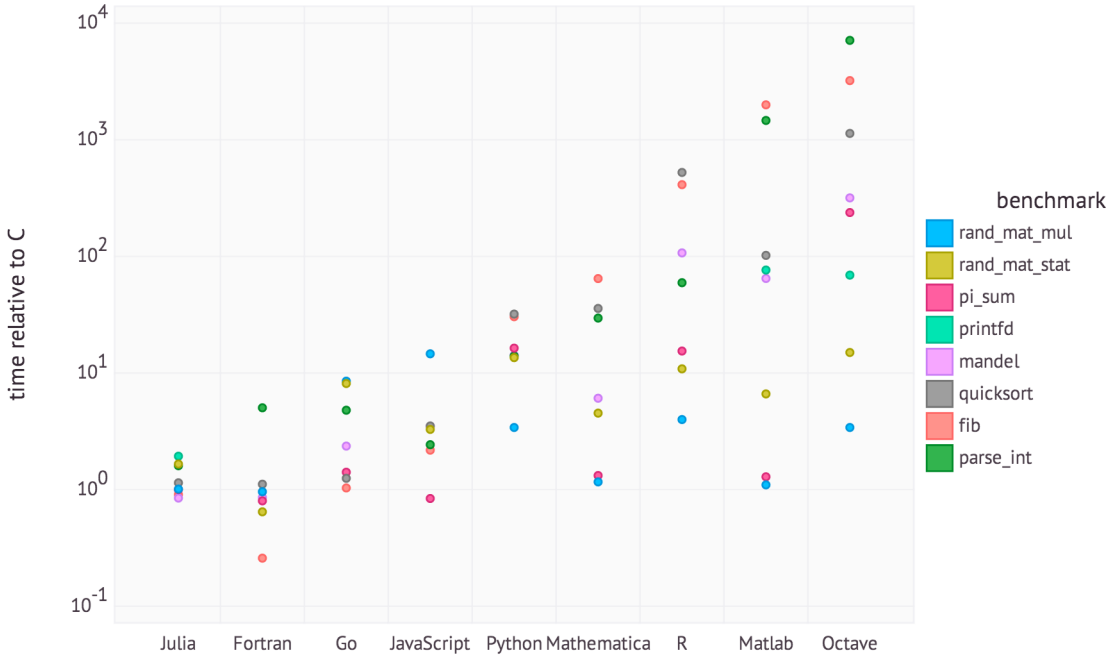

Figure 3.2: Simple micro-benchmarks showing Performance comparison of various language. Benchmark execution time comparative to C. (Execution time shorter is better, the performance of C = 1.0) [1].

3.2.2 Development Tools: IDE

An integrated development environment (IDE) provides developers/programmers an interface to write code and develop tools. It has also used by data scientists and researchers for data analysis and developing a data-driven application. A different program and different programmers prefer different programming languages for coding. And not all the IDE's supports all the available languages. Julia is one of the most powerful languages for data analysis and some of the available IDE's that supports Julia are shown in Table 3.2.

SI	Options
1	Jupyter notebook
2	Vim
3	Juno

Table 3.2: IDE options for Julia.

- **Jupyter notebook**: It is one of most powerful and data scientists favourite development tool for data cleaning and data transformation, statistical modeling, numerical simulation, data visualization, data analysis, machine learning, and more.

  License:

  BSD license (Free and Open source).

  Advantages

  - The UI convenience of Jupyter notebook is great with a clear and interactive environment.
  - It offers selective and cell-level code execution, which is helpful in data science work.
  - Julia has good support for macros that let us manipulate code like data.
  - One of best feature of Jupyter notebook is, it can be run on remote server and use it through web browser.
  - It offers different kernel option, like IJulia and IPython.
  - It supports 40+ programming languages (including Julia, Python, R, and Scala) and most big data tools, such as Apache Spark, NumPy, Pandas, scikit-learn, ggplot2, DataFrames, Flux and TensorFlow.

  Disadvantages

  - Debugging is hard in Jupyter Notebook.

  - Version controlling in Jupyter Notebook is almost impossible. Though JupyterLab solves this problem but it requires extra installation work.
  - It does not show up any suggestion while coding.

- **Juno**: Juno is the biggest name in Julia-specific IDE's. It is built on Atom. It also offers developers a powerful environment for Julia computing and development.

  License:

  MIT License (free software)

  Advantages

  - Juno is customizable like other Atom IDE's.
  - Juno comes with both Julia and Atom packages.
  - Git integration is available, also provide tree-view to see all the history and files.

  Disadvantages

  - Juno is slow like other Atom IDE's.
  - This IDE is not very good for large files. Because there are issues loading and manipulating large files in Juno.

- **Vim**: Vim is a powerful text editor and a successor of Vi text editor written by Bram Moolenaar. It supports dozens of command and modes, which makes it programmers favourite.

  License:

  Free software (Vim License), charityware

  Advantages

  - Vim is a super fast editor. It not only loads up things fast, but user also can work very fast with this editor.
  - It is very, very much customizable. It has hit a number of these top IDE lists for this feature.
  - Vim offers a lot of features, learning curve of vim is high as user will keep discovering new feature after several years of use.
  - Vim has some vibrant, useful and almost endless plugin environment.

Disadvantages

- Vim is hard to use initially as it offers a lot of modes and commands. The learning curve is steep while learning all these modes and commands supported in Vim.
- Configuring vim is a endless process as there is always something that can be updated or improved.
- Many plugins depend on optional Julia, Python and Lua features, which may or may not be included.

Comparison and Decision

There is no IDE that is completely perfect. Each IDE described avobe have their pros and cons. Though these days most of the IDE's are almost same thus choosing the IDE depends on user's previous experience. We are working on Jupyter notebook for python development for long time. We found that we are familiar with Jupyter notebook for Julia development also.
Now comparing Julia, Vim and Juno we found that Vim and Juno both have great customizable ability and Jupyter is lacking behind on that. But Vim is difficult to learn and Juno is slow for big data analysis. On the other hand Jupyter is fast , offers great documentation facilities, also can be run in remote server. And it is also some data scientists favourite. Our familiarity and these advantages makes Jupyter notebook win over Vim and Juno for the development IDE of the project.

3.2.3 Document preparation tools

Documentation is one of the key aspect of a project. There are plenty of options for document writing. But we have considered the following two as the option.

SI	Options
1	LaTex
2	MS Word

Table 3.3: Document preparation tool options.

- **LaTex**: LaTex is a typesetting and document preparation software system for writing formatted document. It comes with some unique features for writing mathematical, scientific and technical document.

License:

LaTeX Project Public License (LPPL)

Advantages

- LaTex offers great packages for typesetting mathematical expressions more beautifully and it can be done much more easily than Word.
- LaTex is completely platform independent.
- LaTex comes with a free software license.
- User just needs to put some scripts and LaTex will do rest of the things that is almost automated. User doesn't need to worry about the layout and other aspects.
- Latex handles the cross-referencing with re-numbering and bibliography smartly.
- Community support from LaTex is a great advantage and there is a collaborative effort to upgrade LaTex with new packages.

Disadvantages

- Fairly steep learning curve
- Collaborators who are not familiar with LaTeX will face difficulty reviewing LaTex manuscripts.
- There are many features that require libraries, sometimes user have to finds those.
- Changing layout is really painful.

- **MS Word**: MS word is one of most popular document preparation application of Microsoft office package. It is widely used in corporate sector.

License:

Trialware, SaaS (Initially free then Requires buying)

Advantages

- Easy to learn, use and Learning curve is flat.
- In most of the situations it happens that the collaborators does not know latex but knows word.
- Spell check and word suggestion is one of the best feature of word.
- New versions of word also suggests crammer correction .

Disadvantages

- Representing complex mathematical expressions and notations are difficult in word. If they can be done they often will look ugly.
- It comes with Trialware license. Software under this licence is initially free for few days but for further use it requires dollars.
- Bibliography handling in word is difficult and sometimes need third party software to handle the task.

Comparison and Decision

Latex is usually considered as the best option for writing scientific content.Wheres word is for general use. Latex handles and presents bibliography, referencing, figures , mathematical expressions and indexing very smartly. On other hand MS Word makes it difficult to handle these things. Moreover latex is free but Word is paid. Thus the conclusion is word is easy to use and simple but the features of latex makes it superior for documentation writing. And that made us choose latex over MS Word.

3.3 Proposed Algorithm

One of the most important steps in developing an effective classification algorithm, is the data preparation. Moreover, in the case of class imbalanced dataset, data preparation step need to be considered with care full manner, to make it a balanced one. In this project, we introduce a sampling approach for balancing the imbalanced dataset. All the instances are splitted into two groups mainly, the majority and the minority based on their class value. Then, majority instances are clustered into several clusters using k-means clustering algorithm. Number of the cluster value is determined by using the elbow method. From each clusteres, informative instances are merged with a sample of minority instances. Informative instances are selected from the cluster centers, boarders and some random selection. This step creates several sub-dataset which are nearly balanced and helps the classifier to identify both majority and minority instances accurately. More details of this step is discussed on section 4.2.2 the Data Preparation.

Finally, the balanced samples created are employed into the Random Forest algorithm to train the ensemble learner. Random Forest is itself an ensemble learner, which creates multiple decision trees and takes the majority vote of each tree. Besides, it reduces the chance of overfitting. In the proposed ensemble method, all the balanced samples are used to generate Random Forests, and their majority votes are taken into the count when testing the model using test dataset. Figure 3.3 depicts the entire procedure of the algorithm.

Algorithm 3.1: Proposed Hybrid Algorithm

Input : Training Set $D_{Imbalanced}$, Clustering algorithm(Kmeans) with C4.5 Algorithm.
Output: Ensemble of trees

Split the $D_{Imbalanced}$ into $D_{Majority}$ and $D_{Minority}$;
Create clustered sub-datasets $C_i = \{C_1, C_2, ..., C_k\}$ applying Kmeans clustering on $D_{Majority}$;
for $i \leftarrow 0$ **to** k **do**
    create several sample sub-datasets $C_{sample_i} = \{C_1,\ C_2,\ ...,\ C_k\}$ with informative instances from C_i;
end
for $i \leftarrow 0$ **to** $k-1$ **do**
    create balanced sets, $D_{Balanced_i} \leftarrow C_{sample_i} \cup D_{Minority}$;
end
for $i \leftarrow 0$ **to** $k-1$ **do**
    generate a tree, DT_i from $D_{Balanced_i}$ applying C4.5 by randomly selected features;
    $error \leftarrow getError(DT_i)$;
    if $error \geq 0.5$ **then**
        return to step 10 ;
    end
end
Apply test data D_{Test} on ensemble of trees using majority voting approach;

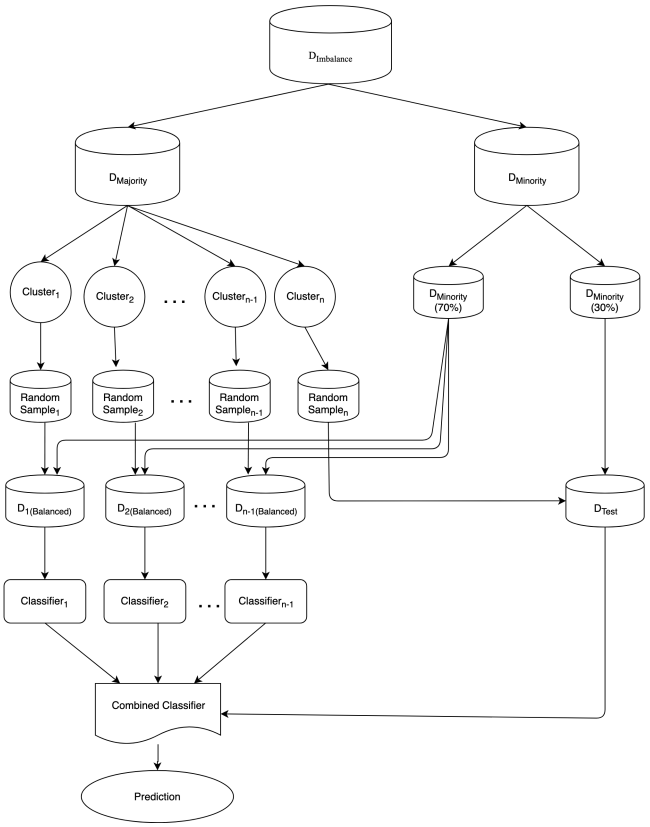

Figure 3.3: Work flow diagram of proposed Hybrid Model for fraud classification

3.4 Licensing

License of Selected tools

SI	Tools	License
1	Julia	MIT License and GNU GPL v2;
2	Jupyter notebook	BSD licenses
3	LaTeX	LaTeX Project Public License (LPPL)

Table 3.4: License of selected tools.

License of the proposed project

– Open source and free software.

3.5 Summary

In this chapter, we discussed briefly the requirement specification of this project and did an extensive analysis of all the available software and hardware tools that are needed to

complete this project. Julia (programming language), Jupyter notebook (code editor) and LaTeX(documentation tool) have been selected for accomplishing this project. Finally, we introduced the proposed hybrid algorithm.

Chapter 4

Implementation and Results

This chapter discusses the environment setup for this research work, the dataset and the experimental results carried out throughout the research on different benchmark algorithms along with the proposed hybrid method.

4.1 Environment Setup

In this project, we used the Julia Programming Language version 1.1 for most of the coding and algorithm implementations. For package and dependency management Conda package manager, and as a code editor Jupiter Notebook is used in this project. Notable libraries and packages used in this project are given below.

- ScikitLearn.jl: a Julia implementation of popular data science library Scikit-Learn
- Clustering.jl: provides all kind of clustering algorithms
- MLDataUtils.jl: supports all kind of data related functionalities
- DecisionTree.jl: provides Decision Tree algorithms such as CART, C4.5 etc.
- PyPlot: a Julia implementation of popular piloting library, that supports different data visualization approaches.

As computing system, intel core i5-7500, 3.4 GHz CPU, 8 gigabytes of primary memory, and 64-bit Linux Operating System (Ubuntu 18.04 LTS) is used in this project.

4.2 Dataset Description

There is a lack of publicly available money transaction dataset to work in the domain of fraud detection. In this paper, we have used the dataset created by *PaySim* simulator which is a synthetic dataset reflecting the real mobile money transactions dataset using multiagent-based simulation [27]. This dataset has over 6 million (6,362,620) transactions with ten features and one target class. The target class has two label representing

fraudulent and non-fraudulent transactions. Also, the class imbalance ratio of this dataset is very high such that, It has only 8,213 instances of illegal transactions and other 6,354,407 instances belong to legal transactions. Table 4.1, provides the brief overview of the dataset and In Fig. 4.1, a pie chart shows the imbalanced distribution of instances of each class values.

Table 4.1: Dataset Overview.

Parameter	Explaination and Values
Dataset Nature	Highly Class Imbalanced
Number of Instances	6,362,620
Number of Features	10
Missing Value	0
Class Value	2
Feature Types	Numerical and Categorical
Learning type	Binary Classification

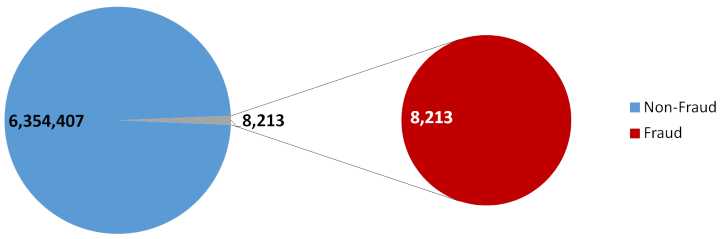

Figure 4.1: A pie chart showing distribution of Non-Fraud and Fraud instances

4.2.1 Features and Class Description

The dataset consists of 10 features and one target class.

- **Step:** Step represents unit time in terms of real world. In this dataset, step = 1 means, its one hour in real word. There are total step of 744, which is 30 days in real world.
- **Type:** All transactions are categorized into five types. Such as, Cashi In, cash Out, Debit, Payment and Transfer. Figure 4.2, shows the frequency distribution of each types.
- **Amount:** Amount refers to the amount of money used in transaction in local currency.
- **nameOrig:** The customers name, who initiated the transaction.

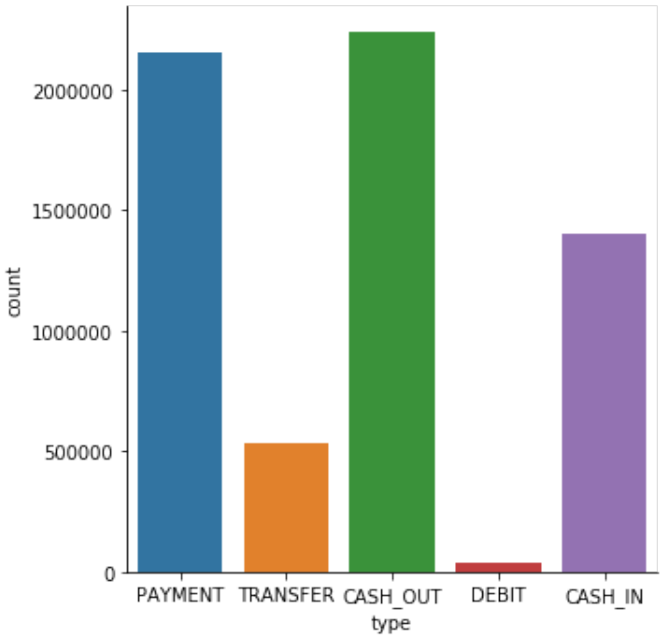

Figure 4.2: A histogram showing frequency distribution of different transaction types.

- **oldbalanceOrig:** The customers starting balance prior to the transaction.
- **newbalanceOrig:** The customers balance after the transaction.
- **nameDest:** The name of the person who is receiving the transaction.
- **oldbalanceDest:** Initial balance of the customer who is receiving money.
- **newbalanceDest:** Balance of the receiver, after the transaction is completed.
- **isFlaggedFraud:** Any effort to transfer more than 200.000, is considered as an illegal attempt in this dataset.
- **isFraud:** isFraud substantiate whether the transaction is fraud or non-fraud. Fraudulent activity like, trying to get financial gain by fetching entire amount in the account are considered as fraud in this dataset. Table 4.2, provides the distribution of instances to each class value and their binary representation used in the dataset.

Class Value	Binary Representation	Number of Instances
Fraud	1	8,213
Non-Fraud	0	6,354,407

Table 4.2: Class distribution of *isFraud*.

4.2.2 Dataset Preparation

Since this dataset is highly class imbalanced, we split the original dataset based on majority and minority classes. Let's denote the original dataset as $D_{Imbalance}$. First of all, $D_{Imbalance}$ is partitioned into two sub-datasets, separating majority classes of non-fraudulent examples, denoted by $D_{Majority}$ and, the rest fraudulent examples of the minority classes, indicated by $D_{Minority}$. Secondly, the k-means clustering algorithm is employed into $D_{Majority}$ to produce clusters of non-fraudulent instances. Besides, $D_{Minority}$ is divided into two subsets of 70% to use in building a training set and the remaining 30% to creat the test set. Finally, from each cluster, random instances are chosen to combine with $D_{Minority}$ forming nearly balanced sample, $D_{nBalanced}$ where the majority classes are 60%, and the minority classes are 40%.
To evaluate the model, a test dataset, D_{Test} is produced in a similar manner, where the majority class is 70%, and the minority class is 30%. The test dataset is intentionally kept imbalanced to evaluate how the algorithm performs in a real-world scenario. Table 4.3, gives splitted amount of instances of $D_{Majority}$ and $D_{Minority}$ and Table 4.4, gives the amount of instances splitted from $D_{Minority}$ for Training and Testing.

Sub-datasets	**Number of Instances**
$D_{Majority}$	6,354,407
$D_{Minority}$	8,213

Table 4.3: $D_{Majority}$ and $D_{Minority}$ Sub-datasets.

$D_{Minority}$	**Number of Instances**
Training	5,749
Testing	2,464

Table 4.4: $D_{Minority}$ Sub-datasets for training and testing.

Table 4.5, provides the number of instances in each cluster with their cluster ID. Subsequently, Figure 4.3, shows the majority and minority class distribution in original, balanced and test dataset. As shown in Figure 4.6, each $D_{nBalanced}$ contains same amount of instance with an equal number of the majority class instances and an equal number of minority class instances.

4.3 Performance Evaluation Criteria

4.3.1 Confusion Matrix and Performance Metrics

In machine learning, measuring the performance of models is an essential task for understanding the effectiveness of the model. A confusion matrix comes handy in this job. A **confusion**

Cluster ID	Number of Instances
C_1	5,605,297
C_2	2,93,853
C_3	29,689
C_4	4,23,866

Table 4.5: Distribution of generated clusters.

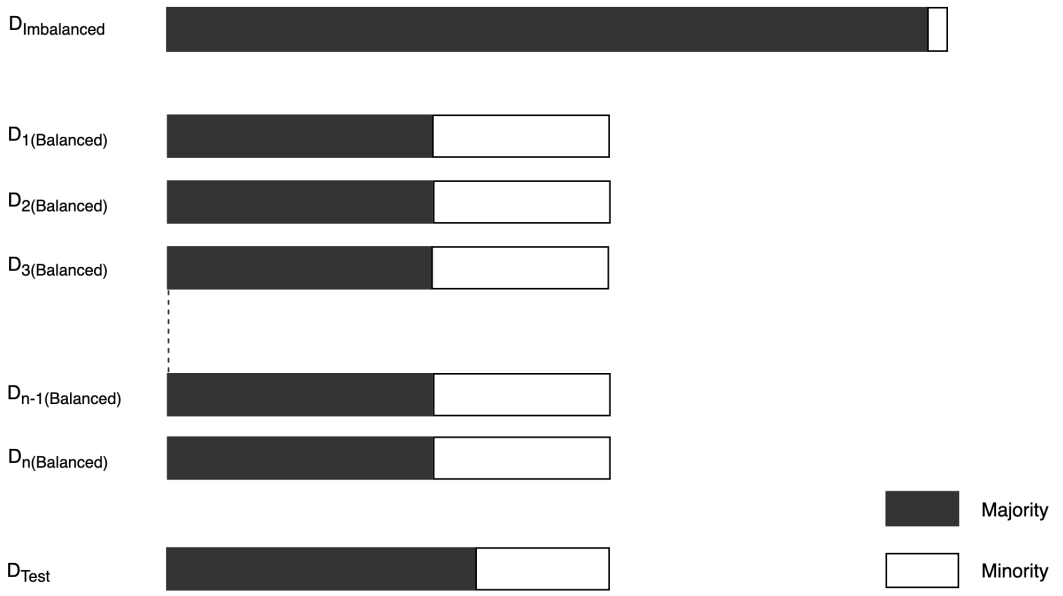

Figure 4.3: Majority and minority class distribution in original, balanced and test dataset.

Sample	Total Instances	Majority Instances	Minority Instances
$D_{1(\text{Balanced})}$	14,372	8,623 (60%)	5,749 (40%)
D_{Test}	8,213	5,749 (70%)	2,464 (30%)

Table 4.6: Majority and minority class distribution in D_{Balanced} and D_{Test}.

matrix is a specific table layout that summarizes the performance of a classification model. It is typically used to evaluate the performance of a supervised learning algorithm. Table 4.7 shows a confusion matrix for a binary classifier, where each column and each row corresponds to the actual class and predicted class respectively. Also, both actual and predicted class value can be either positive class Fraud or negative class non-Fraud. The cells of the matrix contain the number of correctly classified and misclassified instances. True Positive (TP) represent the cases when actual and prediction are the same with positive class Fraud. False Positive (FP) are the cases when the actual class value is negative class non-Fraud, but the predicted class value is is positive class Fraud. False Negative (FN) are the occurrences when the class value is actually negative class Fraud,

but the Prediction is positive class non-Fraud. True negative (TN) are the events when both actual and prediction is the same with negative class non-Fraud.

		Actual		
		Fraud	Non-Fraud	Total
Predicted	Fraud	TP	FP	$TP + FP$
	Non-Fraud	FN	TN	$FN + TN$
	Total	$TP + FN$	$FP + TN$	N

Table 4.7: Confusion matrix for Fraud Classification.

A confusion matrix is conducive to measure accuracy, **True Positive(TP) rate**, **True Negative(TN) rate**, **False Positive(FP) rate**, **Precision**, **Recall**, **F1-score**, and other performance metrics. The values of these performance parameters are calculated using the equation defined by eqs. (4.1)–(4.10)

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)} \tag{4.1}$$

$$TP\,rate\ or\ Recall\ or\ Sensitivity = \frac{TP}{(TP + FN)} \tag{4.2}$$

$$TN\,rate\ or\ Specificity = \frac{TN}{(FP + TN)} \tag{4.3}$$

$$FP\,rate = \frac{FP}{(FP + TN)} \tag{4.4}$$

$$FN\,rate = \frac{FN}{(FN + TP)} \tag{4.5}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{4.6}$$

$$F1\_score = \frac{2 \times TP}{(2 \times TP + FP + FN)} = \frac{2 \times Precision \times Sensitivity}{(Precision + Sensitivity)} \tag{4.7}$$

$$Gmean = \sqrt{Sensitivity \times Specificity} = \sqrt{TP\,rate \times TN\,rate} \tag{4.8}$$

$$AUC\_ROC = \frac{Sensitivity + Specificty}{2} = \frac{TP\,rate + TN\,rate}{2} \tag{4.9}$$

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP) \times (TP + FN) \times (TP + FP) \times (TP + FN)}} \tag{4.10}$$

4.3.2 ROC Curve

ROC curve also known as **AUC-ROC** (Area Under Curve-Receiver Operating Characteristics) curve is an important performance measurement metric for examining any multi-class classification model's efficiency. ROC depicts a probability curve whereas the AUC score gives an understanding of how much the model can accurately predict positives as positives

and negatives as negatives. A higher AUC score means the model can distinguish between different classes and able to predict correctly. The highest value of AUC can be 1 and the lowest value of AUC can be zero. The probability curve of the ROC curve is represented by a graph where the true positive rate is placed along the y-axis and the false positive rate is placed along the x-axis. A perfect classification model would have an AUC score near to 1 where the TPR $\approx$ 1 and FPR $\approx$ 0. Figure 4.4 shows a graph containing a ROC curve and AUC score of perfect classfication model.

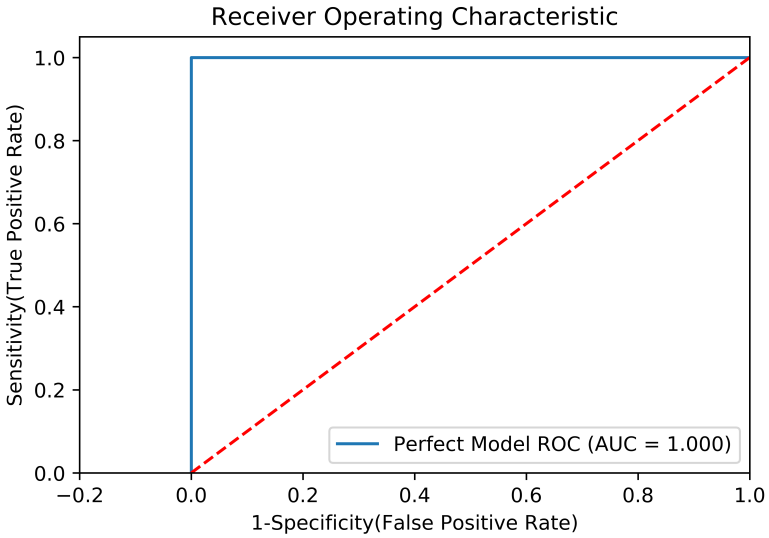

Figure 4.4: A perfect model's ROC curve

4.4 Experimental Results

This section represents the results of the research work carried out throughout the research. We have experimented with the proposed hybrid method along with 9 other benchmark machine learning algorithms for classifying fraud on the PaySim dataset.

4.4.1 Proposed Hybrid Method

The confusion matrix of Table 4.8 shows a classification report of the proposed hybrid method. Also, Table 4.9 shows the score of different performance parameters. Each of these experiments is performed 5 times and their mean scores are depicted in Table 4.9.

		Actual		
		Fraud	Non-Fraud	Total
Predicted	Fraud	2457	221	2464
	Non-Fraud	7	5528	5749
	Total	2678	5535	8213

Table 4.8: Confusion matrix for Fraud Classification.

Matrices	Iteration1	Iteration2	Iteration3	Iteration4	Iteration5	Average
sensitivity	0.9975	0.9927	0.9983	0.9959	0.9963	0.9961
specificity	0.9605	0.7873	0.9998	0.9963	0.9657	0.9419
accuracy	0.9716	0.8490	0.9994	0.9962	0.9749	0.9582
mcc	0.9360	0.7191	0.9985	0.9910	0.9429	0.9175
auc	0.9790	0.8900	0.9991	0.9961	0.9810	0.9691
gmean	0.9789	0.8840	0.9991	0.9961	0.9809	0.9678
precision	0.9155	0.6670	0.9996	0.9915	0.9257	0.8999
f1_score	0.9547	0.7979	0.9990	0.9937	0.9597	0.9410

Table 4.9: Performance of proposed hybrid method on test set.

4.4.2 AdaBoost

The table 4.10 shows the performance score of AdaBoost classifier with independent test set.

- **Input Parameter:** n_estimators = 50, learning_rate = 1.0, algorithm ='SAMME.R', random_state = RANDOM_STATE
  *For each iteration RANDOM_STATE $\in \{42, 7, 13, 23, 37\}$

Matrices	Iteration1	Iteration2	Iteration3	Iteration4	Iteration5	Average
sensitivity	0.6049	0.6003	0.5795	0.6006	0.6195	0.6010
specificity	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
accuracy	0.9994	0.9994	0.9994	0.9994	0.9995	0.9994
mcc	0.7337	0.7278	0.7320	0.7406	0.7571	0.7383
auc	0.8024	0.8001	0.7897	0.8002	0.8097	0.8004
gmean	0.7777	0.7747	0.7612	0.7749	0.7871	0.7751
precision	0.8906	0.8831	0.9253	0.9139	0.9259	0.9077
f1_score	0.7205	0.7147	0.7127	0.7248	0.7423	0.7230

Table 4.10: Performance of AdaBoost algorithm on test set.

4.4.3 Random Forest

The table 4.11 shows the performance score of Random Forest with independent test set.

- **Input Parameter:** base_estimator = DecisionTreeClassifier(max_depth = 1), n_estimators = 50, learning_rate = 1.0, algorithm = 'SAMME.R', sampling_strategy = 'auto', replacement = False, random_state = RANDOM_STATE
  *For each iteration RANDOM_STATE $\in \{42, 7, 13, 23, 37\}$

Matrices	Iteration1	Iteration2	Iteration3	Iteration4	Iteration5	Average
sensitivity	0.7789	0.7975	0.7762	0.7759	0.7781	0.7813
specificity	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
accuracy	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997
mcc	0.8729	0.8822	0.8708	0.8716	0.8714	0.8738
auc	0.8894	0.8987	0.8881	0.8879	0.8890	0.8906
gmean	0.8825	0.8930	0.8810	0.8808	0.8821	0.8839
precision	0.9785	0.9761	0.9772	0.9795	0.9763	0.9775
f1_score	0.8674	0.8778	0.8652	0.8659	0.8660	0.8684

Table 4.11: Performance of Random Forest algorithm on test set.

4.4.4 RUSBoost

The table 4.12 shows the performance score of RUSBoost with independent test set.

- **Input Parameter:** base_estimator = DecisionTreeClassifier(max_depth = 1), n_estimators = 50, learning_rate = 1.0, algorithm = 'SAMME.R', sampling_strategy = 'auto', replacement = False, random_state = RANDOM_STATE
  *For each iteration RANDOM_STATE $\in \{42, 7, 13, 23, 37\}$

Matrices	Iteration1	Iteration2	Iteration3	Iteration4	Iteration5	Average
sensitivity	0.9446	0.9164	0.9223	0.9165	0.8899	0.9179
specificity	0.9980	0.9983	0.9980	0.9983	0.9981	0.9981
accuracy	0.9979	0.9982	0.9979	0.9982	0.9980	0.9980
mcc	0.6020	0.6074	0.5947	0.6074	0.5817	0.5986
auc	0.9713	0.9573	0.9602	0.9574	0.9440	0.9581
gmean	0.9710	0.9565	0.9594	0.9565	0.9425	0.9572
precision	0.3846	0.4035	0.3844	0.4033	0.3811	0.3914
f1_score	0.5466	0.5603	0.5427	0.5601	0.5337	0.5487

Table 4.12: Performance of RUSBoost algorithm on test set.

4.4.5 XGBoost

The table 4.13 shows the performance score of XGBoost with independent test set.

- **Input Parameter:** max_depth = 3, learning_rate = 0.1, n_estimators = 100, verbosity = 1, silent = None, objective = 'binary:logistic', booster = 'gbtree', n_jobs = 1, nthread = None, gamma = 0, min_child_weight = 1, max_delta_step = 0, subsample = 1, colsample_bytree = 1, colsample_bylevel = 1, colsample_bynode = 1, reg_alpha = 0, reg_lambda = 1, scale_pos_weight = 1, base_score = 0.5, random_state = RANDOM_STATE, seed = None, missing = None
  *For each iteration RANDOM_STATE $\in \{42, 7, 13, 23, 37\}$

Matrices	Iteration1	Iteration2	Iteration3	Iteration4	Iteration5	Average
sensitivity	0.7005	0.7110	0.6927	0.6838	0.7077	0.6991
specificity	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
accuracy	0.9995	0.9996	0.9995	0.9995	0.9996	0.9995
mcc	0.8291	0.8366	0.8232	0.8221	0.8336	0.8289
auc	0.8502	0.8555	0.8464	0.8419	0.8538	0.8496
gmean	0.8370	0.8432	0.8323	0.8269	0.8412	0.8361
precision	0.9816	0.9847	0.9786	0.9887	0.9825	0.9832
f1_score	0.8176	0.8258	0.8112	0.8084	0.8227	0.8172

Table 4.13: Performance of XGBoost algorithm on test set.

4.4.6 GradientBoost

The table 4.14 shows the performance score of GradientBoost with independent test set.

- **Input Parameter:** loss = 'deviance', learning_rate = 0.1, n_estimators = 100, subsample = 1.0, criterion = 'friedman_mse", min_samples_split = 2, min_samples_leaf = 1, max_depth = 3, random_state = RANDOM_STATE
  *For each iteration RANDOM_STATE $\in \{42, 7, 13, 23, 37\}$

4.4.7 Balanced Bagging

The table 4.15 shows the performance score of Balanced Bagging with independent test set.

- **Input Parameter:** n_estimators = 10, base_estimator = DecisionTreeClassifier(), max_samples = 1.0, sampling_strategy = 'auto', replacement = False, n_jobs = 1, random_state = RANDOM_STATE, verbose = 0
  *For each iteration RANDOM_STATE $\in \{42, 7, 13, 23, 37\}$

Matrices	Iteration1	Iteration2	Iteration3	Iteration4	Iteration5	Average
sensitivity	0.6132	0.6234	0.6085	0.4875	0.6633	0.5991
specificity	0.9999	0.9998	0.9999	0.9999	0.9999	0.9998
accuracy	0.9994	0.9993	0.9994	0.9993	0.9995	0.9993
mcc	0.7662	0.7145	0.7697	0.6851	0.7968	0.7464
auc	0.8066	0.8116	0.8042	0.7437	0.8316	0.7995
gmean	0.7831	0.7895	0.7800	0.6982	0.8144	0.7730
precision	0.9579	0.8196	0.9742	0.9635	0.9575	0.9345
f1_score	0.7478	0.7081	0.7491	0.6474	0.7837	0.7272

Table 4.14: Performance of Gradient Boost algorithm on test set.

Matrices	Iteration1	Iteration2	Iteration3	Iteration4	Iteration5	Average
sensitivity	0.9955	0.9911	0.9953	0.9968	0.9934	0.9944
specificity	0.9921	0.9934	0.9927	0.9926	0.9928	0.9927
accuracy	0.9921	0.9934	0.9927	0.9926	0.9928	0.9927
mcc	0.3718	0.3947	0.3787	0.3858	0.3853	0.3833
auc	0.9938	0.9923	0.9940	0.9947	0.9931	0.9936
gmean	0.9938	0.9923	0.9940	0.9947	0.9931	0.9936
precision	0.1400	0.1583	0.1452	0.1505	0.1505	0.1489
f1_score	0.2454	0.2729	0.2534	0.2615	0.2614	0.2589

Table 4.15: Performance of Balanced Bagging algorithm on test set.

4.4.8 SMOTEBoost

The table 4.16 shows the performance score of SMOTEBoost with independent test set.

- **Input Parameter:** n_estimators = 50, learning_rate = 1.0, ratio="auto", algorithm = 'SAMME.R', random_state = RANDOM_STATE
  *For each iteration RANDOM_STATE $\in \{42, 7, 13, 23, 37\}$

Matrices	Iteration1	Iteration2	Iteration3	Iteration4	Iteration5	Average
sensitivity	0.6246	0.5794	0.6309	0.5971	0.5791	0.6022
specificity	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
accuracy	0.9994	0.9993	0.9994	0.9994	0.9993	0.9994
mcc	0.7459	0.7265	0.7580	0.7401	0.7310	0.7403
auc	0.8123	0.7897	0.8154	0.7985	0.7895	0.8011
gmean	0.7903	0.7612	0.7943	0.7727	0.7610	0.7759
precision	0.8914	0.9116	0.9113	0.9179	0.9235	0.9111
f1_score	0.7346	0.7085	0.7456	0.7236	0.7118	0.7248

Table 4.16: Performance of SMOTE Boost algorithm on test set.

4.5 Result Comparison

Table 4.17 presents a comparison of different algorithms on various performance parameters. All the scores are the mean value of 5 iterations. The highest mean in each parameter is highlighted in bold.

Algorithm	sensitivity	specificity	accuracy	mcc	auc	gmean	precision	fscore
AdaBoost	0.6010	0.9999	0.9994	0.7383	0.8004	0.7751	0.9077	0.7230
Random Forest	0.7812	0.9999	**0.9996**	0.8737	0.8906	0.8838	0.9775	0.8684
RUSBoost	0.9179	0.9982	0.9981	0.5986	0.9581	0.9572	0.3914	0.5487
Balanced Random Forest	0.9948	0.9857	0.9858	0.2831	0.9903	0.9903	0.0818	0.1511
XGBoost	0.6991	0.9999	**0.9996**	0.8289	0.8496	0.8361	**0.9832**	0.8172
Gradient Boost	0.5992	**0.9999**	0.9994	0.7464	0.7996	0.7730	0.9345	0.7272
Easy Ensemble	0.9867	0.9702	0.9702	0.1968	0.9785	0.9784	0.0405	0.0778
Balanced Bagging	0.9945	0.9927	0.9927	0.3833	**0.993**	**0.993**	0.1489	0.2589
SMOTE Boost	0.6022	0.9999	0.9994	0.7403	0.8011	0.7759	0.9111	0.7248
Hybrid Method	**0.9962**	0.9419	0.9582	**0.917**	0.9691	0.9678	0.8999	**0.941**

Table 4.17: Mean performance result of different algorithm

The roc curves of proposed hybrid method along with 9 other state of the art algorithm shown in Figure 4.5 represents that the proposed hybrid method is as good as the other benchmark algorithms. The hybrid method achieves 0.979 AUC score, whereas the highest is 0.991 acquired by Balanced Random Forest.

A overall performance comparison of proposed hybrid method along with 9 other benchmark algorithm is shown in Figure 4.6. It shows that the overall performance of the hybrid method is superior to others for classifying imbalanced data.

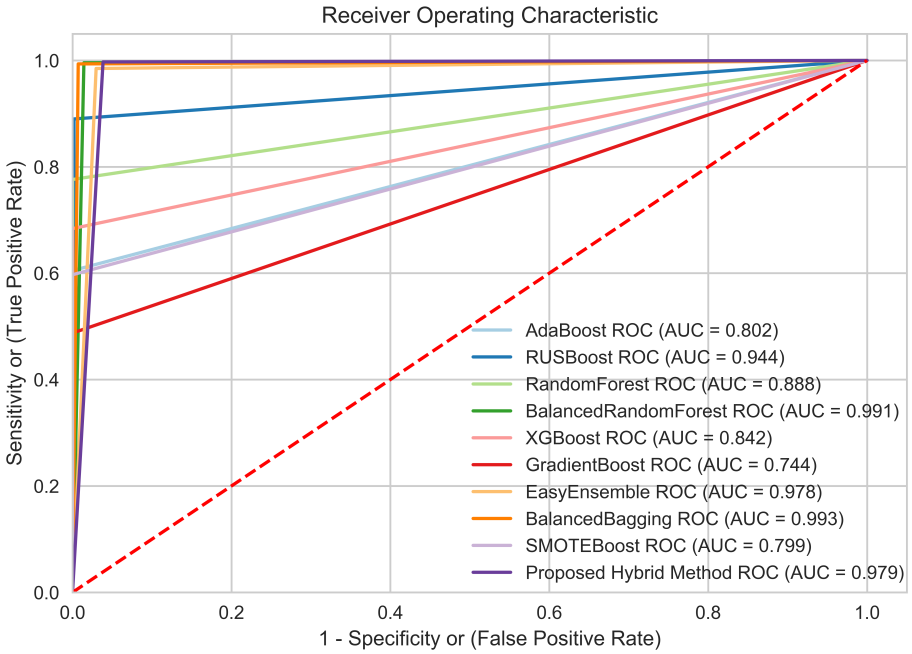

Figure 4.5: ROC Curve of different algorithms

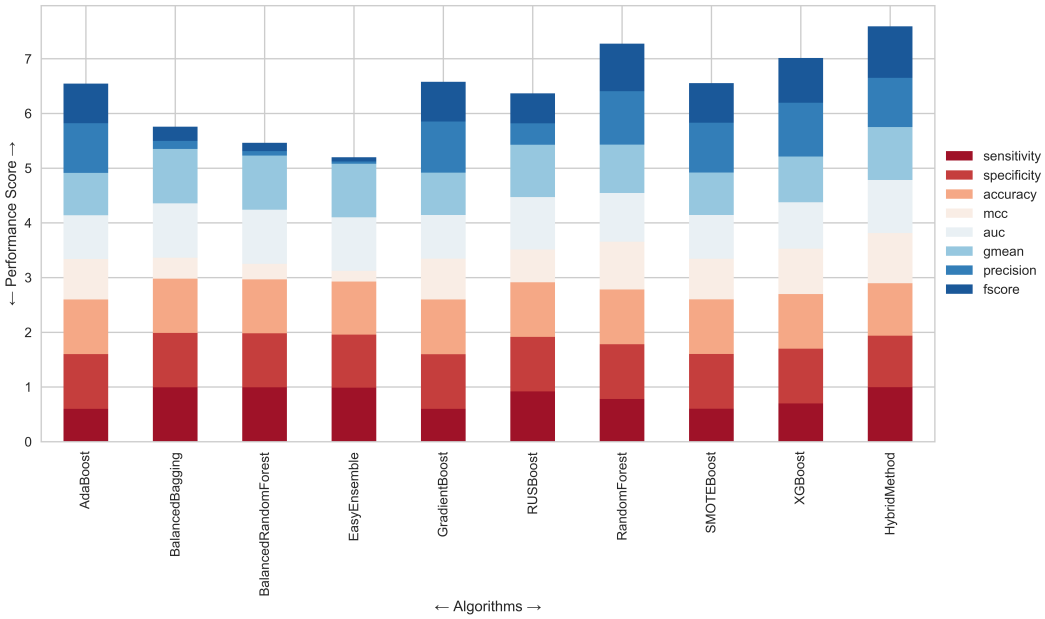

Figure 4.6: Comparison of the hybrid method with others

4.6 Summary

In this chapter we have discussed about the experimental analysis of this project. A Confusion matrix is computed from the models prediction and original label of the test set instances. It provides a comprehensive statistics of the performance by calculating number of true positive, true negative, false positive and false positives. Furthermore,

other sophisticated performance parameters like, precision, recall and f1-scores are also calculated using outcome of confusion matrix. Moreover, other benchmark algorithms which are good for imbalanced dataset, are used to compare performance with our method. To find out these mentioned parameters, in this project, a test set of 8213 instances kept isolated from the training time. We used Julia implementation of scikit-learn package for all kind of experimental analysis in this project. Also, from data analysis to pre-processing and designing model, is written in the Julia Language.

Chapter 5

Standards, Impacts and Design Constraints

This chapter describes about the standards to be followed for this project, the impacts and the constrains of the project.

5.1 Compliance with the Standards

This project follows some Safety, Ethical and Technological standards. All of these standards are discussed in this section.

5.1.1 Safety Standard

(i) **UL 639** : UL Standard for Safety Intrusion-Detection Units.

Our project which is based on Fraud-detection, it confirms the standard requirements for UL 639.

5.1.2 Ethical Standard

(i) **ISO/IEC 12207.**
ISO/IEC 12207 promotes not to develop any software which may harm mankind, society, and the environment by economically, physically or in any other form. Our project ensures all the requirements of ISO/IEC 12207.

(ii) **ACM Code of Ethics**
The principles of ACM Code of Ethics is to promote the use the skill for the well being of society, avoiding acts that cause negative consequences, to be honest, and trustworthy with personal acts and to be fair to everyone. Our project also abides by the principles of the ACM Code of Ethics.

5.1.3 Technological Standard

As we are using python programming language for our project, we have used PEP 8 as our coding standard. Latex is used to write all the documents related to our project. And we have used CSV formatted data-set. Table 5.1 shows the complete list of technological standards.

Technology	Standard
Programming Language	Julia and Python
Coding Standard	PEP 8(Python)
Markup language	Latex(Documentation)
Data Formats	Comma Separated Value, Microsoft Excel, ARFF

Table 5.1: Technological Standards.

5.2 Impacts

There are several ways this project establish impacts on the economy, health, society etc. Throughout this section we discussed about these impacts.

5.2.1 Economic Impact

Fraudulent activities may cause significant financial loss to any organization. This fraudulent behavior in the financial transaction will be identifiable by this project, and it can be expected that this project will prevent the stealing of thousands of cores of money. Moreover, real-time fraud detection is time-consuming, but our proposed project will make fraud detection much more faster, comfortable and automated.

5.2.2 Social Impact

Social status and reputation is very essential for any banking or corporate sector. However fraud activities against them can damage their social status and reputation. This project can be used in any banking sectors or any corporate sectors where fraud- detection systems are necessary. During this project, our primary focus will be detecting fraudulent behavior in the banking sector or any other financial transaction system. When this project will be used in every banking system or any financial transaction system, we prospect it will have an outstanding social impact by detecting fraud.

5.2.3 Ethical Impact

Fraud transaction itself is an unethical activity, which will be reduced to a tolerable level by the project we have proposed. Most of the fraud events that occur in financial transactions directly or indirectly affect the mass people, our project will lessen this by protecting their

property reserved in banks and different financial organizations. At a national level, this is also a matter of concern to protect national resources reserved in banks. The law enforcement agencies will be able to detect these intruders and take proper action against them. Thus it can be said that the ethical impact of the project is positive.

5.2.4 Health and Safety Impact

This project does not leave any adverse impact to health and safety, instead, it aims to protect people's property by providing better security at a transaction level, which indirectly ensures sound mental and physical health to the customers.

5.2.5 Environmental Impact

Our designed project does not affect our ecosystem or consume any energy from our ecosystem, or it is not a threat to our environment. Thus, it can be stated that our project does not impact the environment.

5.3 Design Constraints

This project also faces some constraints such as manufacturability and sustainability. All these constraints are discussed in this section.

5.3.1 Manufacturability Constraint

The major manufacturability constraint of this project is the highly imbalanced dataset, which we use to train our model to build our designed project. The main problem of imbalanced datasets is calculating the actual performance of the algorithm. Machine learning algorithms produce weak classifiers when they are trained up with highly imbalanced data. Those algorithms show bias to the majority class by overfitting and treat the instances of minority class as noise. Though there are some standard classifier algorithms, for example, Logistic Regression, Naive Bayes Classifier, and Decision Trees, but there is a likelihood of the wrong classification of the minority class.

5.3.2 Political Constraint

The primary constraint of our designed project is if the Government restrict the sharing policy of personal accounting information. Another thing is if the Government changes its security requirement for a valid transaction. Though right now the government provides a standard security bridge for any transaction. Till now, an authorized company can classify there every transaction fraud or not by their security system. That is why we do not need to wait for the Government's approval.

5.3.3 Sustainability Constraint

If the security requirement of a transaction is changed, then the project may fail to detect fraud which can cost personals and businesses a huge amount of money. Security patch must be updated after a certain period of time.

The massive imbalance in class might deform the importance of certain correlations which evaluate to our class variable.

5.4 Summary

In this chapter, different aspects of the project's Standards, Impacts and Constraints have been discussed. In the standard section, we selected "UL 639" as the safety standard, "ISO/IEC 12207" as the ethical standard. Additionally Julia, Python, Latex, etc have been chosen as the technological standard. In the other two section we pointed out all the Impacts and Constrains. Although there are some constraints, afterward the project completion the impact on the economy, ethics and safety are huge.

Chapter 6

Conclusion

This chapter summaries the thesis contributions and also discusses the future works.

6.1 Summary

Distinguishing fraud and non-fraud transactions is a challenging task in machine learning and data mining, as the dataset comes with high-class imbalance ratio. Therefore, most algorithms show biases in predicting as because of the insignificant amount of fraud in the dataset. It is a severe concern to classify instances which are illegal as accurate as possible, rather than misclassifying a legal one as a fraud. Among all the techniques, under-sampling is one of the popular choices to deal with imbalanced data. The dataset we used to train and test the model, is a high dimensional big data, having over 6 million examples with high-class imbalance proportion. We proposed a new cluster-based under-sampling technique, which clusters the majority class instances and marge with minority class instances to produce balanced samples. Along with balancing, the ensemble classifier was designed with Random-Forest classifier to utilize cases from all the segment of the majority class instances. Thus, it shows a significant result in detecting illegal transactions with 97.24% accuracy, 99.72% precision, 91.75% recall and 95.57% f1-score.

6.2 Future Work

In future work, we shall continue the investigation on the performance of the proposed approach in an excellent manner with other established under-sampling techniques. Moreover, modifying the core algorithm can open a new door in classifying illegal transactions as an illegal one with minimal error.

References

[1] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[2] Keke Gai, Meikang Qiu, Xiaotong Sun, and Hui Zhao. Security and privacy issues: A survey on fintech. In *International Conference on Smart Computing and Communication*, pages 236–247. Springer, 2016.

[3] Clifton Phua, Damminda Alahakoon, and Vincent Lee. Minority report in fraud detection: Classification of skewed data. *SIGKDD Explorations*, 6, 2004.

[4] Andrea Dal Pozzolo, Olivier Caelen, Yann-Aël Le Borgne, Serge Waterschoot, and Gianluca Bontempi. Learned lessons in credit card fraud detection from a practitioner perspective. *Expert Syst. Appl.*, 41:4915–4928, 2014.

[5] Wei Wei, Jinjiu Li, Longbing Cao, Yuming Ou, and Jiahang Chen. Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web*, 16:449–475, 2012.

[6] Dewan Md. Farid, Li Zhang, Chowdhury Mofizur Rahman, M. Alamgir Hossain, and Rebecca Strachan. Hybrid decision tree and naïve bayes classifiers for multi-class classification tasks. *Expert Syst. Appl.*, 41:1937–1946, 2014.

[7] Dewan Md. Farid, M. Abdulla Al Mamun, Bernard Manderick, and Ann Nowé. An adaptive rule-based classifier for mining big biological data. *Expert Syst. Appl.*, 64:305–316, 2016.

[8] Dewan Md Farid, Li Zhang, Alamgir Hossain, Chowdhury Mofizur Rahman, Rebecca Strachan, Graham Sexton, and Keshav Dahal. An adaptive ensemble classifier for mining concept drifting data streams. *Expert Systems with Applications*, 40(15):5895–5906, 2013.

[9] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[10] Farshid Rayhan, Sajid Ahmed, Asif Mahbub, Rafsan Jani, Swakkhar Shatabda, and Dewan Md Farid. Cusboost: cluster-based under-sampling with boosting for

imbalanced classification. In *2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS)*, pages 1–5. IEEE, 2017.

[11] American Bankers Association. 2017 aba deposit account fraud survey, January 2018. [Online; accessed 4-April-2019].

[12] Ashutosh Deshmukh and TLN Talluru. A rule based fuzzy reasoning system for assessing the risk of management fraud. In *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, volume 1, pages 669–673, 1997.

[13] Yiğit Kültür and Mehmet Ufuk Çağlayan. Hybrid approaches for detecting credit card fraud. *Expert Systems*, 34(2):e12191, 2017.

[14] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, and Bernard Manderick. Credit card fraud detection using bayesian and neural networks. In *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*, pages 261–270, 2002.

[15] Saad Darwish. An intelligent credit card fraud detection approach based on semantic fusion of two classifiers. *Soft Computing*, 04 2019.

[16] Clifton Phua, Damminda Alahakoon, and Vincent Cheng-Siong Lee. Minority report in fraud detection: classification of skewed data. *SIGKDD Explorations*, 6:50–59, 2004.

[17] Shiven Sharma, Colin Bellinger, Bartosz Krawczyk, Osmar R. Zaiane, and Nathalie Japkowicz. Synthetic oversampling with the majority class: A new perspective on handling extreme imbalance. *2018 IEEE International Conference on Data Mining (ICDM)*, pages 447–456, 2018.

[18] Ludmila I Kuncheva, Álvar Arnaiz-González, José-Francisco Díez-Pastor, and Iain AD Gunn. Instance selection improves geometric mean accuracy: a study on imbalanced data classification. *Progress in Artificial Intelligence*, 8(2):215–228, 2019.

[19] Zhongbin Sun, Qinbao Song, Xiaoyan Zhu, Heli Sun, Baowen Xu, and Yuming Zhou. A novel ensemble method for classifying imbalanced data. *Pattern Recognition*, 48(5):1623–1637, 2015.

[20] Show-Jane Yen and Yue-Shi Lee. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, 36(3):5718–5727, 2009.

[21] Guido Van Rossum. Python 0. 1991.

[22] Travis E Oliphant. Python for scientific computing. *Computing in Science & Engineering*, 9(3):10–20, 2007.

[23] R Core Team et al. R: A language and environment for statistical computing. 2013.

[24] Ross Ihaka and Robert Gentleman. R: a language for data analysis and graphics. *Journal of computational and graphical statistics*, 5(3):299–314, 1996.

[25] Hadley Wickham, Romain Francois, Lionel Henry, Kirill Müller, et al. dplyr: A grammar of data manipulation. *R package version 0.4*, 3, 2015.

[26] Hadley Wickham. *ggplot2: elegant graphics for data analysis*. Springer, 2016.

[27] Edgar Alonso Lopez-Rojas, Ahmad Elmir, and Stefan Axelsson. Paysim: a financial mobile money simulator for fraud detection. In *28th European Modeling and Simulation Symposium 2016)*, 2016.