

2021-01

Suspicious Money Laundering Detection Using Imbalanced Dataset

Yinur, Zelalem

<http://ir.bdu.edu.et/handle/123456789/12403>

Downloaded from DSpace Repository, DSpace Institution's institutional repository



BAHIR DAR UNIVERSITY

BAHIR DAR INSTITUTE OF TECHNOLOGY

SCHOOL OF RESEARCH AND POSTGRADUATE STUDIES

FACULTY OF COMPUTING

**Suspicious Money Laundering Detection Using Imbalanced
Dataset**

MSc. Thesis in Information Technology

Yinur Zelalem Mengistu

Program: MSc. In Information Technology (IT)

Main Advisor: Seffi Gebeyehu (Assistant Professor)

January 2021

Bahir Dar, Ethiopia

Acknowledgements

“Foremost, I would like to express my sincere gratitude to my advisor, Mr. Seffi Gebeyehu, for his valuable and constructive suggestions during the planning and development of this research work. He consistently guided me in the right direction in this research work.

I would also like to thank Dr. Kadurka Rajasheka, Dr. Mekonnen Wagaw and Mr. Solomon Getahun for their continuous support and for being a part of my thesis committee. I must express my deep gratitude to my wife Frehiwot, my two daughters Herani and Hanan and my younger sister Nuhamin Zelalem for providing me with unfailing support and continuous encouragement throughout my years of study.

Finally, I would like to thank all my friends especially Teshager Worku and Selamawit Mihreet who made my time here in BDU a memorable one.”

Yinur Zelalem

Bahir Dar University

February 2021

BAHIR DAR UNIVERSITY
BAHIR DAR INSTITUTE OF TECHNOLOGY
SCHOOL OF RESEARCH AND GRADUATE STUDIES
FACULTY OF COMPUTING
MSC. THESIS IN INFORMATION TECHNOLOGY

Approval of thesis for defense result

I here confirm that the change requires by the examiners have been carried out and incorporated in the final thesis.

Name of student Yinur Zelalem Mengistu Signature [Signature] Date 3/3/2021

As member of the board of examiners, we examine this thesis entitled "Suspicious Money Laundering Detection Using Imbalanced Dataset" by Yinur Zelalem Mengistu. We hereby certify that the thesis is accepted for fulfilling the requirements for the award of the degree of Master of science in "Information Technology"

Board of Examiners

Name of Adviser	Signature	Date
<u>Selhi S</u>	<u>[Signature]</u>	<u>march 15/2021</u>
Name of External Examiner:	Signature	Date
<u>Zewdie Mossie (PhD)</u>	<u>[Signature]</u>	<u>3/3/2021</u>
Name of internal Examiner	Signature	Date
<u>Mekuria A. (PhD)</u>	<u>[Signature]</u>	<u>15/03/2021</u>
Name of Chairperson	Signature	Date
<u>Mekonnen W. (PhD)</u>	<u>[Signature]</u>	<u>15/03/2021</u>
Name of Chair Holder	Signature	Date
<u>Derajew h</u>	<u>[Signature]</u>	<u>15/03/2021</u>
Name of Faculty Dean	Signature	Date
<u>Daerachew Mekonen</u>	<u>[Signature]</u>	<u>15/03/2021</u>



Faculty Stamp

Abbreviations

AI	Artificial Intelligence
AML	Anti-Money Laundry
CFT	Counter Financing of Terrorism
CNN	Convolutional Neural Network
CRISP-DM	Cross-Industry Standard Process for Data Mining
DM	Data Mining
FATF	Financial Action Task Force
FIC	Financial Intelligence Center
FIU	Financial Intelligence Unit
FOSS	Free and Open Source Software
FSRB	Financial Action Task Force Style Regional Body
GAFISUD	Financial Action Task Force on Money Laundering in South America
GAN	Generative Adversarial Network
GDP	Gross Domestic Product
GIABA	Inter-Governmental Action Group against Money Laundering in West Africa
GCN	Graph Convolutional Networks
GRU	Gated Recurrent Unit
IMF	International Monetary Fund
KYC	Know Your Customer

LSTM	Long Short-Term Memory
MABS	Multi-Agent-Based Simulation
McLayS	Machine learning cooperating Layering Simulation
MENAFATF	Middle East and North Africa Financial Action Task Force
ML	Money Laundry
MLP	Multi layer perceptron
MONEYVAL	Measures and the Financing of Terrorism
MSE	Mean Squared Error
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SAR	Suspicious Activity Report
SNA	Social Network Analysis
SVM-S	Support Vector machine with spark
SMOTEENN	Synthetic Minority over sampling with Edited nearest Neighbor
UNODC	United Nations Office on Drugs and Crime

Contents

Chapter 1: Introduction	- 1 -
1.1 Background	- 1 -
1.2 Statement of the Problem	- 7 -
1.3 Objectives	- 8 -
1.3.1 General Objective	- 8 -
1.3.2 Specific objectives.....	- 8 -
1.4 Research Questions	- 8 -
1.5 Methodologies	- 8 -
1.6 Model	- 8 -
1.7 Data Sources	- 9 -
1.8 Procedure.....	- 9 -
1.9 Implementation environment	- 9 -
1.10 Scope of the study	- 9 -
1.11 Significance of the study	- 10 -
1.12 Organization of the study	- 10 -
Chapter 2 Literature Review	- 11 -
2.1 Related Work	- 11 -
2.2 Preliminaries	- 16 -
2.3 Deep Learning	- 16 -
2.4 Deep Learning Approach.....	- 17 -
2.5 CNN	- 18 -
2.6 Convolutional Neural Network Layers	- 19 -
2.7 Classification	- 20 -
2.8 Class imbalance problem	- 20 -
2.9 Handling class imbalance problem	- 21 -
2.10 Evaluation Metrics	- 25 -
Chapter 3 Research Methodology	- 27 -
3.1 Proposed Model.....	- 27 -
3.2 Dataset Description.....	- 29 -
3.3 Data splitting.....	- 33 -

3.4 Regularization techniques.....	- 33 -
3.6 Training and testing phase.....	- 36 -
3.7 Performance evaluation of the proposed model	- 36 -
CHaPTETR 4 Experment Result	- 37 -
4.1 Without Resampling and Regularization	- 37 -
4.2 Under sampling.....	- 39 -
4.1.1 Near Miss Under-sampling	- 41 -
4.1.2 Tomek links Under sampling.....	- 43 -
4.1.3 Under-sampling with Edited Nearest Neighbor Rule.....	- 44 -
4.1.4 Under-sampling with Neighbor hood Cleaning Rule	- 46 -
4.2 Over sampling	- 47 -
4.2.1 Random Over Sampling.....	- 47 -
4.2.2 SMOTE over Sampling.....	- 49 -
4.2.3 ADASYN Over Sampling.....	- 51 -
4.2.4 Hybrid (combination of over sampling and Under sampling techniques)	- 53 -
4.2.4.1 SMOTETomek.....	- 53 -
4.2.4.2 SMOTEENN.....	- 54 -
Chapter 5 Conclusion and Future Works	- 59 -
5.1 Conclusion.....	- 59 -
5.2 Recommendation.....	- 60 -
REFERENCE.....	- 61 -
ANNEX.....	- 66 -

Lists of Figure

Figure - 1:	Types of Learning
Figure - 3:	Under Sampling techniques
Figure - 4:	Over sampling techniques
Figure- 5:	Hybrid Techniques
Figure - 6:	Fraud Detection Model
Figure - 7:	Proposed Model (CNN)
Figure - 8:	Flow Charts of CNN Model
Figure-9:	Unknown, Licit verses Illicit graph with values
Figure-10:	Licit verses Illicit graph
Figure-11:	Data Correlation
Figure-12:	Data Splitting
Figure-13:	Dropout Regularization
Figure-14	Best Training Accuracy and Validation accuracy at epoch 3
Figure-15:	Model Accuracy and Loss Without Resampling and Regularization
Figure-16:	Random Under Sampling
Figure-17	Best Training Accuracy and Validation accuracy at epoch 10
Figure-18	Model Accuracy and loss for random under sampling
Figure-19:	Best Training Accuracy and Validation accuracy at epoch 6
Figure-20:	Model Accuracy and Loss for Near Miss under Sampling
Figure-21:	Best Training Accuracy and Validation accuracy at epoch 2
Figure-22:	Model Accuracy and Loss for Tomek Links Under-Sampling
Figure-23:	Best Training Accuracy and Validation accuracy at epoch 2
Figure-24:	Model Accuracy and Loss for Edited Nearest Neighbor Rule
Figure-25:	Best Training Accuracy and Validation accuracy at epoch 3

Figure-26:	Model accuracy and loss for neighborhood cleaning rule
Figure-27:	Random Over Sampling
Figure-28:	Best Training Accuracy and Validation accuracy at epoch 10
Figure-29:	Model accuracy and loss for over sampling
Figure-30:	Best Training Accuracy and Validation accuracy at epoch 10
Figure-31:	Model Accuracy and Loss for SMOTE
Figure-32:	Best Training Accuracy and Validation accuracy at epoch 15
Figure-33:	Model Accuracy and Loss for ADASYN Sampling
Figure-34:	Best Training Accuracy and Validation accuracy at epoch 15
Figure-35:	Model Accuracy and Loss for SMOTETomke
Figure-36:	Best Training Accuracy and Validation Accuracy at epoch 15
Figure-38:	Best Training Accuracy and Validation accuracy at epoch 15

Lists of Table

Table-1:	Summery of Related work
Table-2:	Variable Description
Table-3:	Accuracy Result Without Resampling and Regularization
Table-4:	Random Under Sampling Result
Table-5:	Near Miss Under Sampling Result
Table-6:	Under-Sampling with Edited Nearest Neighbor Rule Result
Table-7:	Under-sampling with Neighborhood Cleaning Rule
Table-8:	Random Over Sampling Result
Table-9:	SMOTE Over Sampling Result
Table-10:	ADASYN Over Sampling Result
Table-11:	Hybrid Over Sampling Result
Table-12:	SMOTE & Edited nearest Neighbor Result
Table-13:	SMOTEENN Result
Table-14:	Summerized Results

Abstract

Money laundering is a serious worldwide criminal activity in financial institutions and a process in which illegitimate funds and assets are converted into legitimate funds and assets. These illegitimate funds can be found from serious crimes, such as drug trafficking, smuggling, political corruption, terrorist activity, financial fraud and so on. Suspicious money laundering transaction detection is very difficult when the dataset is imbalanced and the aim of this thesis work is, balancing the imbalance data using various techniques and increasing the performance accuracy of suspicious money laundering transactions detection using deep learning. The proposed algorithm evaluated by using an Elliptical dataset that has been developed for money laundering detection in 2019 due to the secrecy of Financial Intelligence Unit (FIU) data. This data set located from Kaggle.com and has 203,769 rows with 166 columns, from this amount of data two percent (4,545) are labeled class-1 (illicit). Twenty-one percent (42,019) are labelled class-2 (licit). The remaining transactions are not labeled whether licit or illicit but have other features. The dataset is highly imbalanced and to curve this problem, various data resampling techniques such as Random under-sampling, Random oversampling and Hybrid techniques have been done. In addition to this, we used Dropout and L2 regularization techniques during model training to constraint the model to fit our data accurately and not overfit. For experimental purpose we used python with keras library because we are familiar to this tool and it is current-state-of-the-art. The proposed technique Convolutional Neural Network (CNN) in combination with different resampling techniques have been applied to detect whether the transactions are illicit or licit and the performance of the model is evaluated based accuracy result. The experimental result showed that the hybrid resampling technique so called Synthetic Minority Oversampling with Edited Nearest Neighbor (SMOTEENN) with L2 kernel regularize has a better result than the rest of resampling techniques and the accuracy improved from 96.1 to 98.93.

Chapter 1: Introduction

1.1 Background

There is no universal definition for money laundering because of the complexity of its nature. With this many researchers followed a various way of expression, for instance, (Le Khac,2010) stated on the following way “Money laundering (ML) is a process to make illegitimate income appear legitimate; this is also the process by which criminals attempt to hide the true origin and ownership of the proceeds of their criminal activity”. On the other hand (Lorenz, 2020) has defined like this, Money laundering is a high-impact problem on a global scale. Criminals obtain money illegally from serious crimes and inject it into the financial system as seemingly legitimate funds. Besides this, (Jin & Qu,2019) described in this way “Money laundering is an attempt by criminals to legalize illegal income, which mainly refers to the disguise of the source and nature of illegal income through various means. In today's economic globalization and financial integration, money laundering activities are extremely harmful to the country's economy, finance, and even political order and social stability”. According to (Zhang & Trubey, 2018), Money laundering is the criminal practice of processing illegal money through a series of transactions to gain access to the legal financial system. It is a type of financial crime with potentially devastating social and financial effects. It is not only these other researchers, (Luo,2014) elucidated on this direction, Money laundering is the convention of criminal incomes into assets that cannot be tracked to the underlying crime, where the process of concealing sources of money is referred to ‘laundering. Generally, Money laundering is a process of concealing illegitimate income from the origin and passing it through a complex sequence of banking transfers or commercial transactions to be able to appear legitimate income. As we have seen from various authors the definitions are different but it goes to the same meanings and terms.

When we come to our country, Money laundering is strongly prohibited by the government and the government stated on its proclamation number (Federal Negarit Gazeta ,2009) on this method Money laundering and the financing of terrorism it is not only threatens security but also compromises the stability, transparency, soundness, and efficiency of the financial system. In the case of this, Money Laundering is one of the cumbersome risks in the financial system and financial institutions faced different risk that needs critical thinking before completing a transaction.

According to (Basel Committee on Banking Supervision ,2011), some of risks may include the following such as reputation risk, operational risk, concentration risk, and legal risk. Therefore, the governments and financial regulators require financial institutions. Financial institutions must comply with various controlling mechanisms such as Anti Money Laundering (AML), Counter Financing of Terrorism (CFT), and Known Your Customer (KYC) regulations in customer onboarding processes. According to AML and KYC regulations, financial institutions must apply a risk assessment to their new customers. The purpose of these control mechanism are to enable countries and business companies to detect potential risks of their customers. In the case of this, Financial Intelligence Unit (FIU) have been established in our country before some years ago to protect money laundering process.

As per the explanation of (Zeng-an & Li-fang, 2006), Anti Money Laundering is significantly important to both national financial stability and international security. Thus, in addition to anti money laundering detection, Suspicious money launder transaction activity can be identified during the on-boarding or ongoing due diligence of a client, during the transaction monitoring process or it may be raised by any employee of an obliged entity but it has a secret life cycle.

As many researchers described that, Money laundering operated in different life cycles for instance, (Manjunath, 2015), Money Laundering operated in three stages they are Placement, Layering and Integration.

Placement: - is the movement of cash from its source and occasionally, the source easily disguised or misrepresented. This followed by placing it into circulation through financial

institutions, casinos, shops, and other businesses, both local and abroad. The process of placement carried out through many processes.

Layering: - The aim of this stage is to make it more difficult to detect the true origin of the unlawful money and uncover a laundering activity.

Integration: - This is the movement of previously laundered money into the economy mainly through the banking system and thus such monies appear to be normal. Therefore, to eliminate this life cycle anti money laundering procedures and laws are required.

Anti-Money Laundering (AML): - can be a set of procedures, laws, and regulations that are designed to stop money laundering stages or life cycles because money launderers are dynamic to hide their criminal activity or illegal source of money. To stop this criminal activity AML is mandatory.

Nowadays, many researchers have been doing in the areas of anti-money laundering and many financial institutions give serious attention and assigning enormous amounts of budget to control money laundering. During this time, laws and regulations are mandatory to prevent and suppress money laundering activities. For instance, validating customer identification validation before banking business, checking suspicious foreign exchange cash transactions, tracking large cash flows, and blacklisting accounts of suspected money laundering this and the rest of tasks are handled via KYC (Known Your Customer) analyst in the banking industry.

Known Your Customer (KYC): - is the process of verifying the identity of customers, either before or during the time that they start doing business with the organization. In banking industry, KYC is a position for an employee like other bank staff but his/her duties are differs from others because they spent a lot of time by investigating customer's transaction information manually whether the transactions are illicit or licit and providing report to the responsible body. Thus, to improve the investigation process, further studies and modern techniques are required.

As per (Luo, 2014), “existing anti-money laundering methods replay on human intervention, and applying modern techniques remains at an initiating phase”. Therefore,

to improve the detection method modern techniques are needed because the nature of money laundering is dynamic, in the case of this it needs different algorithms and techniques that can be able to detect suspicious financial transactions. Beside this (Zezhong, 2004) described that, detecting suspicious financial transactions is an essential precondition and key aspect of anti-money laundering”. Now a day, anti-money laundering is performed in different governmental and non-governmental banking sectors via KYC analysts especially in Ethiopia the KYC analyst tries to identify, verify, and monitor his customer’s financial transactions manually to ensure, whether the transaction is suspected or not from money laundering and the bank send the hard copy report to Financial Intelligence Unit. Financial Intelligence Unit takes the report that comes from different banking industries and feeds into the database system and they try to make analysis. One of the great challenges for FIU is getting fulfilled and up-to-date report form banking industries due to various reasons such as some banks are forgetting the report, reports are not submitted to FIU on the predefined time and date, others are sending unfulfilled data and the given data is not properly fed into the FIU database system. Finally, FIU generates a suspected report and send it to the Federal Court and the Federal Court will begin an investigation by taking the generated report. In the case of this and the rest of things, have their own side effect on the process of suspicious money laundering transactions detection.

Now a days, using imbalance dataset is a challenge for all area of study specially when the study has conducted with fraud detection. As per (Ertekin, 2007) stated that, various real-world classification tasks, such as medical diagnosis, text categorization and fraud detection suffer by class imbalance problem. Therefore, money laundry is a kind of fraud that can be operated by vairious criminal activities and to detect this kind of fraudulent transactions first the imbalanced dataset must be balance to improve the classification task.

Therefore, the aim of this study is easily detecting suspicious money laundry transactions using imbalanced data in combination with other data resampling and data regularization techniques to improve the performance accuracy.

1.2 Statement of the Problem

Today financial transactions are largely depending upon the advancement of information technology. Due to this reason, the job a lot easier, but it came with its own challenge because the bad guys can easily facilitate its own criminal activity using this technology. In short, the money made by various criminal activities in various parts of the world is injected into a nation's economy to disguise it or give it a lawful appearance. This system is known as Money Laundering. One of the difficulty tasks in anti-money laundering process is detecting suspicious money laundry transaction, because money laundering process touches various life cycle such as placement, layering and integration to disguise suspicious transaction activities. It is not only that, due to the secrecy of financial data, gathering financial data strictly prohibited by the financial institutions this and the rest of things are a great challenge for many researchers to apply different techniques in real situation. But now a days, instead of using real financial data, many researchers are using synthetic data to solve the absence of real financial data. On the other hand, selecting inappropriate synthetic dataset has its own side effect on the specified result. For the past few years, many researchers have been doing various techniques to enhance the detection process of suspicious money laundry transactions but, still now the problem is not completely solved due to various gaps such as, using imbalanced dataset, selecting inappropriate model, overfitting, underfitting and using inappropriate dataset. Therefore, to tackle this problem and to enhance the detection process, further studies are required and the aim of this thesis work is to fulfil the above-mentioned gaps.

1.3 Objectives

1.3.1 General Objective

The general objective of this thesis is detecting suspicious money laundering transactions using deep learning techniques in combination with data resampling and regularization techniques.

1.3.2 Specific objectives

To attain the general objective, different specific objectives have been addressed, these specific objectives among others are to: -

- Understand imbalance dataset and its controlling mechanism.
- Study various types of data resampling techniques to balance the imbalance dataset.
- Understand and propose the model to improve suspicious transaction detection.
- Test and validate the performance accuracy.

1.4 Research Questions

Detecting suspicious money laundering transaction is one of the cumbersome tasks in this are specially when the training dataset is imbalanced. Thus, to easily detect and increase the performance accuracy this paper is needed. At the end of this research process, the following questions must be answered;

- Which technique is better to balance the imbalanced data?
- Can the proposed system easily show the detection process ?
- How to increase the performance accuracy of suspicious transaction detection?

1.5 Methodologies

In this section, several techniques have been applied to realize suspicious money laundering detection. Therefore, we used a deep learning technique to succeed the aim of this thesis work.

1.6 Model

To realize this thesis work, we used a reference model known as the Cross-Industry Standard Process for Data Mining (CRISP-DM). this model contains the following phases such as Business Understanding, Data Understanding, Data preparation, Modeling,

Evaluation and Deployment. According to (Paula, 2016), this model is “well known, flexible and execution of sequence is not rigid.

1.7 Data Sources

Identifying and using relevant data sources is one of the vital phases to implement deep learning model. Therefore, to improve the performance of the detection process we used an Elliptical financial dataset prepared by (Weber, 2018), due to the lack of real financial dataset and this dataset is prepared for money laundering detection based on real dataset samples. This dataset, maps Bitcoin transactions to real entities belonging to licit categories (exchanges, wallet providers, miners, licit services, etc.) versus illicit ones (scams, malware, terrorist organizations, ransomware, Ponzi schemes, etc.).

1.8 Procedure

This study processed chronologically. This is very essential to ensure that the research process is comprehensive and all areas of study are covered. We used a well-designed action plan to guide the research process. In case of this, we can ensure a comprehensive research is achieved on time and the stipulated resources.

1.9 Implementation environment

Deep learning is a subpart of machine learning and it permits computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. As (Lecun, 2015) description, Deep learning can be applied on various domains such as fraud detection, speech recognition, visual object recognition, object detection, sequential data such as text and speech recognitions, and many other domains. To apply deep learning, we used Python because it is a popular and powerful interpreted language which used for both research and development. There are enormous number of libraries in combination with the proposed tools that enable us to develop the proposed model.

1.10 Scope of the study

The scope of this study is only detecting suspicious money laundry transactions and increasing the performance of the detection accuracy using the proposed model in concatenate with various techniques. The proposed algorithm has been applied on an

Elliptical dataset due to the secrecy of real data from financial intelligence unit and this dataset is created for money laundering detection and the data is freely available on kaggle.com.

1.11 Significance of the study

Now a day, suspicious money laundering detection is a very difficult task due to the dynamic nature of money laundering, especially for developing countries like Ethiopia, because, detection process performed manually via bank staff so-called KYC analyst. This way of detection is not effective and efficient to trap suspicious money laundry transactions. Therefore, many researchers have been doing different research using different algorithm such as machine learning, datamining, and other related techniques but still now, the gap is not entirely solved. Therefore, Various types of gaps such as data misclassification, overfitting, underfitting, and performance accuracy problems require deep study in the detection process and researchers are revealed this gap to the next researcher. Thus, this paper clearly stated what does it mean Convolutional Neural Network in deep learning and it showed us how CNN can detect suspicious money laundering transactions. In addition to this, various types of data resampling techniques have been clearly identified to make it balance the imbalance dataset such as oversampling, under sampling, hybrid data resampling techniques and other related to increasing the performance of the accuracy. Finally, anyone who read this paper can easily discover how to regularize data using various types of data regularizes techniques with its advantages. It is not only this, various types of data resampling techniques have been applied in python code to curb the specified gaps and any one can easily understand how to increase the performance accuracy of suspicious money laundering transactions detection using the above-mentioned techniques and any one can compare the result of this paper from other techniques or proposed models to evaluate the performance of this model.

1.12 Organization of the study

We organize the remainder of the paper as follows. Chapter 2 presents literature review and related works. Chapter 3, present the proposed model, Chapter 4 present details the experimental setup and introduces the relevant money laundering detection methods. In Chapter 5 we present our results. Finally, the main conclusions are discussed in Chapter 6.

Chapter 2 Literature Review

2.1 Related Work

Since money laundry has become a huge problem to the financial market, many financial institutions have allocated huge amount of money and gathered teams of the human expert to develop money laundry detection systems. Many researchers have been actively working on this to mitigate the problems such as class imbalance problem, overfitting, performance accuracy and so on. In this chapter, we tried to review some of the past related works that has been done via other researchers.

As we knew, various researchers have been conducted to detect money laundering transactions using various fields for instance machine learning and datamining have been applied in combination with various techniques to give a solution for detection issues such as data selection, algorithm design, and hyperparameter optimization that can affect the detection ability of the proposed solutions. As we stated on section 1.3.1, we used a deep learning technique, because it has the ability to make accurate predictions in the realm of fraud detection and it is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks and as per the explanation of (Maghrebi, 2016), it relies on sets of algorithms that attempt to model high-level abstractions in data by using model architectures with multiple processing layers, composed of a sequence of scalar products and non-linear transformations called activation functions. In addition to this, it is current state-of-the-art to identify suspected money laundering financial transactions and it delivers a lower number of false positives and it increase the performance accuracy the proposed solution instead of traditional one. In the case of this, different researchers used various types of techniques as we have seen underneath.

(Paula, 2016), they used unsupervised deep learning autoencoder technique to detect anomaly situations and to classify Brazilian exporters. In this study, authors introduced CRISPDPM and the data enter in CRISP-DM cycle, and anomalies are detected based on

model learning and they applied unsupervised technique and they did comparison with PCA this is one of the strengths of this paper. On the other way this paper is different from us in terms of, proposed algorithm, data regularization techniques and development tools. for instance, we used various types of data resampling techniques and L2 data regularization to curb overfitting problem but when we came to them, they didn't used regularization techniques as they stated on the paper.

Other researchers, (Weber, 2018), they proposed a Graph Convolutional Network (GCN) with other machine learning techniques such as Variations of Logistic Regression (LR), Random Forest (RF), and Multilayer Perceptron's (MLP) to combat Money laundering problem and all techniques are applied on the elliptical dataset. One of the strengths of this paper is, authors can provide an elliptical dataset to give a solution for the lack of financial dataset but this dataset is large and highly imbalanced and prone to overfitting. According to this, the author didn't consider any solution that can solve class imbalance and overfitting. Therefore, our work is different from them via proposed algorithm, techniques to solved the imbalanced dataset issue like overfitting and developing tools.

As (Jamshidi, 2019) paper, they proposed Adaptive Neuro-Fuzzy with various techniques to analyze online banking accounts and to combat money laundering. In addition to this the proposed method can replace a conventional method to detect the risk of money laundering in suspicious banking transaction. For implementation they used MATLAB software and this work is different from ours through proposed techniques such as data regularization, and implementation tools.

It is not only deep learning, but other techniques have been applied to combat money laundering. For instance data mining techniques are used by Suresh (2016), under this technique they used Hash based Association approach to identify the traversal path of the Laundered and to identifying agent and integrator they used graph theoretical approach. Therefore, our work is different from this one in various ways such as algorithm, techniques, and datasets.

According to (Soltani, 2016), they have proposed a structural similarity framework that applies case reduction methods to progressively reduce the input data set to a significantly

smaller size then, the framework scans the reduced data to find pairs of transactions with common attributes and behaviors that are potentially involved in ML activities for this purpose they used a clustering method to detect potential ML groups. It is not only this but they try to incorporate other techniques like matching-transaction detection and balance scoring filter in order to improve the efficiency of the framework.

Like other fields, machine learning techniques have been applied in order to detect money laundering transactions, as per the description of (Gyamfi & Abdulai, 2019), they have proposed a supervised learning method so-called Support vector machine with spark (SVM-S) to build models that can identify normal and abnormal customer behavior and to evaluate the validity of new transactions. In addition to this study, (Tai & Kan, 2019), they used a two-phase intelligent method based on machine learning and data analysis techniques to identify suspicious money laundering accounts. The first phase is identifying every suspicious money laundering account while the second phase further retrieves highly suspicious ones. Our work is different from this in terms of algorithms, datasets, data resampling and data regularization techniques.

Although, there have been multiple research studies carried out on the different aspects of money laundry detection such as tracking money launderer behavior, improving money laundering detection accuracy this and the rest of things have been covered via different researchers but very little research studies carried out to tackle the class imbalance problem in the fraud detection task. Therefore, our studies aim to perform suspicious transactions analysis on the money laundry detection task that mainly focuses on various techniques such as data resampling and data regularization to deal with the class imbalance problem.

Authors	Title	Techniques	Main Findings	Strength	Weakness
(Paula ,2016)	Deep Learning Anomaly Detection as Support Fraud Investigation in Brazilian Exports and Anti-Money Laundering	unsupervised deep learning Autoencoder	In this study, (CRISPDm) has been introduced and Data in this methodology enters CRISP-DM cycle, and anomalies are detected based on model learning	attributes that proved sufficient to characterize fraudulent exports based on the experience of the author	-They didn't use any algorithm to control over fitting problem. -Using unsupervised techniques is the evaluation of the results against the business objectives to be achieved
(Weber, 2018)	Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics	Random Forest (RF), Multilayer Perceptron's (MLP), and Graph Convolutional Networks (GCN)	A large, labelled transaction data set to the AML community	Various type of algorithms has been done on the dataset before transfer to AML community	The dataset is highly imbalance and I never seen any technique to balance the imbalance data and the accuracy is 96.1
(Jamshidi, 2019)	A Novel Multi objective Approach for Detecting Money Laundering with a Neuro-Fuzzy Technique	Adaptive Neuro-Fuzzy Inference System (ANFIS)	They can create a novel intelligent method for estimating the risk of money laundering	This approach can be used in banking systems as an online technique to analyze the data of customers' accounts	The accuracy of the result is not clearly stated. The future task is not clearly stated.
(Suresh, 2016)	A Hybrid Approach for Detecting Suspicious Accounts in Money Laundering Using Data Mining Techniques	Hash based Association Mining approach And Graph Theoretic Approach	Identify the traversal path of the Laundered Money And Identifying agent and integrator	They proposed a good technique because it can easily detect complex dependencies between transaction and the feature work is clearly stated.	They didn't clearly state what kind of technique applied during data processing.

Authors	Title	Techniques	Main Findings	Strength	Weakness
(Soltani, 2016)	A New Algorithm for Money Laundering Detection Based on Structural Similarity	A new algorithm based on structural similarity And clustering method	They proposed a framework that applies case reduction methods to progressively reduce the input data set to a significantly smaller size.	The performance accuracy of the new algorithm and feature work have been properly stated by the authors.	They generate a synthetic data set without a real transaction and the result may be biased. Accuracy result is 80 and it needs further improvement
(Gyamfi & Abdulai, 2019)	Bank Fraud Detection Using Support Vector Machine	They used a supervised learning method, Support Vector Machines with Spark (SVM-S)	They provide fraud detection architecture.	The author tries to compare the proposed algorithm with the bank propagation algorithms	The precision is 80% to mean this it needs further analysis
(Tai & Kan, 2019)	Identifying Money Laundering Accounts	two-phase intelligent method based on machine learning and data analysis techniques	McLayS method is to facilitate the identification of money laundering accounts from the transactions recorded in the bank sites.	Anti-money laundering specialists manual work has been stopped due to McLayS	The authors used 18 features to trap suspicious money laundering accounts but the characteristics of the suspicious account is different from the predefined feature it can't identify. Accuracy result is 87.04

Table-1: Summary of Related work

2.2 Preliminaries

Before we going to the detail part, it is better to be familiar with some of the vital deep learning theories. This content will help the reader to understand the proposed detection model and deep learning. More specifically, we focused on the supervised learning, in which the model trained based on labeled data.

2.3 Deep Learning

There is nonexistence of unified definition for deep learning in various written documents and literatures. In the case of this, many researcher defined in several ways as we have seen from various published documents, As (Deng & Yu, 2013) stated that, A class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification. On the other side, (Alom, 2019) explained that, Deep learning based representation learning involves a hierarchy of features or concepts, where the high-level concepts can be defined from the low-level ones and low-level concepts can be defined from high-level ones.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. There are various models in deep learning and these models are trained by using a large set of labeled data and neural network architectures that contain many layers. These models are a part of machine learning techniques which involves Artificial Neural Networks such as Convolutional neural networks, Deep Belief Network, Auto-encoders, Recurrent Neural Network, and Restricted Boltzmann Machine are all various methods.

Using this model deep learning can able to solve almost all kinds of problems in different application domains. In order to this, deep learning can be defined on the following way. Deep learning is a part of machine learning and Artificial Intelligence that imitates the working nature of the human brain for processing data and creating patterns for decision making.

2.4 Deep Learning Approach

Deep learning methods or approach can be categorized as follows: Supervised, semi-supervised, or partially supervised, and unsupervised.

Supervised type of approach used a labeled data, in this algorithm, samples of data both illicit and licit records are predetermined to train the models. Therefore, this method is used to identify patterns that have been occurred previously. The other type of approach is, unsupervised learning that used unlabeled data and the algorithm will find the pattern by itself. According to (Niu, 2019), they used both types of approach for comparison study of credit card fraud detection and the experimental results show that supervised models perform slightly better than unsupervised models in this study.

The final approach is Semi-supervised, this approach used the combination of both supervised and unsupervised learning. This approach used by (Niu, 2019), for bank fraud detection in wire transaction and it is effective in finding abnormal users with various fraud pattern. This technique is some data is labeled while other data is not. But typically, most data will be unlabeled. Semi-supervised learning can be used for classification, regression, and prediction.

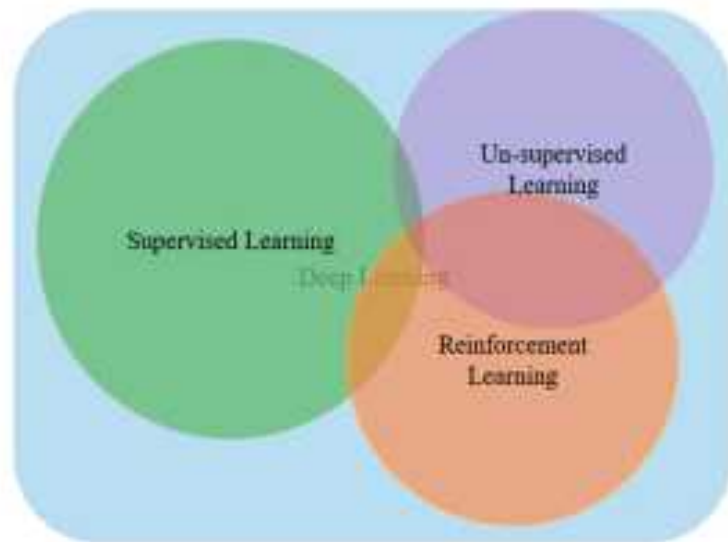


Figure-1: Types of Learning: (Alom, 2019)

To realize this study, we used a supervised or labeled dataset in deep learning techniques to detect suspicious money laundering transactions because the elliptical data set is already having a labeled dataset as per (Weber, 2018) and this approach has various merits as we stated on the above section 2.4

2.5 CNN

CNN is one of the most popular deep neural networks, and the name comes from mathematical linear operation between matrixes called convolution as stated by (Albawi & Mohammed, 2017) and it is designed for processing structured arrays of data such as Image Classification and Segmentation, Object Detection, Video Processing, Natural Language Processing, and Speech Recognition as clearly stated by (Khan, 2020). The architecture of a convolutional neural network is a multi-layered feed-forward neural network, made by stacking many hidden layers on top of each other in sequence. This sequential design that allows convolutional neural networks to learn hierarchical features and the connectivity pattern between neurons started by the organization of the visual cortex as (Lindsay, 2020) paper.

CNN has the following four layers the first layer is convolution, the second is the ReLU layer, the third layer is pooling and the fourth layer is Fully connected. Convolutional and Fully Connected layers have parameters but non-linearity and pooling layers don't have parameters.

In this study, we used a Convolutional Neural Network (CNN) to detect suspicious money laundering and it is one of the popular deep learning techniques which used for various fields such as Image recognition and OCR, Object detection for self-driving cars, Face recognition on social media, Image analysis in healthcare and so on. In the case of this, different researcher proposed this technique for classification and anomaly behavior detection. According to (Fu, 2016) they used CNN model for credit card fraud detection. On the other hand (Lv, 2019), they used CNN model to detect fraudulent bank account and it is not only this but other researchers applied for different tasks to detect fraud and as per their conclusion this technique has a good capability for classification and detection of suspicious transactions.

2.6 Convolutional Neural Network Layers

The CNN model contains a sequence of layers, and each layer of a Convolutional Neural Network transforms one volume of activations to another through a differentiable function.

This model composed of the following layers such as Convolution layer, Rectified Linear Unit (ReLU), Pooling Layer and Fully connected layer as we have stated on the below section as (Indolia, 2018) paper.

The main building block of CNN is the convolutional layer, this layer is the first layer to extract features from the given data and the fundamental mathematical operation to merge sets of information have been done in this layer. Convolutional networks are merely neural networks that use convolution in place of general matrix multiplication and contain parameters like a filter, kernel, and so on.

The other types of layer is Rectified Linear Unit (ReLU). As (Murugan, 2017) stated that this is an activation function defines the output of a neuron based on given a set of inputs. This layer transform function only activates a node if the input is above a certain quantity while the input is below zero, the output is zero, but when the input rises above a certain threshold it has a linear relationship with the dependent variable. In this layer, every negative value is removed and replaced by zero's this is done to avoid the values from summing up to zero.

pooling layer is another vital component of a convolutional neural network. As (Murugan, 2017) clarified that, it is a form of nonlinear subsampling which used to implement reduction .This layer reduce the number of parameters, memory footprint, and amount of computation in the network by taking the highest value and hence control over-fitting. In the case of this, we used max pooling to to make it downsampling.

Finally, after the above layers, high-level reasoning in the neural network has been done by fully connected layers. (C. L. Zhang, 2018) show us, The fully connected (FC) layer is one of the most fundamental modules in CNN. It is widely used in traditional CNN models. However, it is known that FC may cause overfitting

Neurons in a fully connected layer have connections to all activations in the previous layer, as seen in regular (non-convolutional) artificial neural networks. We can thus compute their activations thus refine transformation, with matrix multiplication followed by a bias offset (vector addition of a learned or fixed bias term).

2.7 Classification

The classification problem in deep learning can be defined as the task of detecting the class label of a given data point. For instance, suspicious money laundering transaction detection can be identified as a classification problem. In this case, the goal is to detect whether the given transaction is illicit or licit. As (Sokolova & Lapalme, 2009), classification is categorized into four types they are: -

Binary classification:- refers to those classification tasks that have two class labels.

Multi-class classification:- refers to those classification tasks that have more than two class labels.

Multi-label classification:- refers to those classification tasks that have two or more class labels, where one or more class labels may be predicted for each sample.

Imbalanced classification: refers to classification tasks where the number of examples in each class is unequally distributed.

Typically, imbalanced classification tasks are binary classification tasks where most samples in the training dataset belong to the normal class and a minority of samples belong to the abnormal class. This thesis work, deals with the binary classification problem because (Wang, 2016) a binary-class data set, we call the data set imbalanced if the minority class is under represented compared to the majority class.

2.8 Class imbalance problem

Most of the time, suspicious transactions detection possesses unbalanced class distribution the problem occurs when the number of the illicit class labels is very low as compared to the licit class label. As(Ertekin, 2007), Various real-world classification tasks, such as medical diagnosis, text categorization, and fraud detection suffer from this phenomenon.

Thus, Resampling approaches address this concern by modifying the class distribution to balance the minority and majority class distribute as (Bennin, 2019).

2.9 Handling class imbalance problem

There are various approaches to handle class imbalance problem for instance resampling approach, ensemble-based approach, and cost-sensitive learning approach can be mentioned. This study only deals with resampling approach to give a solution for unbalanced class distribution. Therefore, some data preprocessing task must be performed before providing data as an input to the model. In the case of class imbalance problem, such data preprocessing is performed at data level approach, which is called data resampling approach.

As (Wang, 2016) stated that, standard classifiers (classifiers don't consider data imbalance) tend to misclassify the minority samples into majority samples when the data is imbalanced, which results in quite poor classification performance and to curb this problem, we used the data re-sampling technique, which can improve the classification performance.

Re-sampling is a widely adopted technique for dealing with imbalanced datasets, and it is often very easy to implement, fast to run, and an excellent starting point. Re-sampling changes the dataset into a more balanced one by adding instances to the minority class or deleting ones from the majority class, that way we build better deep learning models. The following approaches have been done to balance the imbalance dataset.

Random Under Sampling technique: - is one of a resampling technique which used to randomly remove instances from the majority class to balance the dataset before applying classification technique as stated by (Kaur & Gosain, 2018). It is best to implement when the size of the dataset is huge and reducing the majority samples can greatly boost the runtime and reduce the storage troubles.

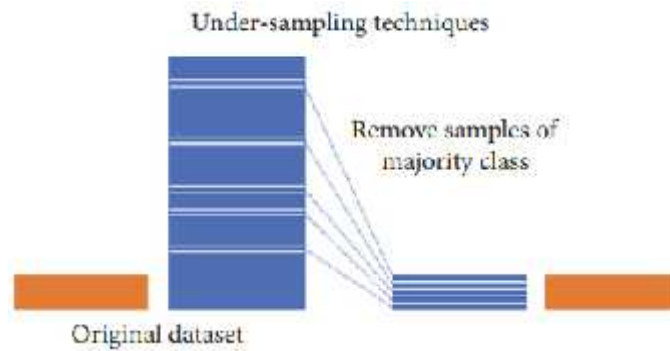


Figure-3: Under Sampling Technique (Shakya, 2018)

One of the disadvantages of using this technique is losing some important information. Therefore, to solve this problem we used another type of resampling techniques.

Oversampling technique:- is another type of technique which used to balance the data distribution by applying the opposite or reverse of under-sampling methods. In random oversampling method, existing minority samples are replicated in order to increase the size of a minority class (Shelke, 2017). One of the main advantages of using oversampling technique is that, it is better to keep information.

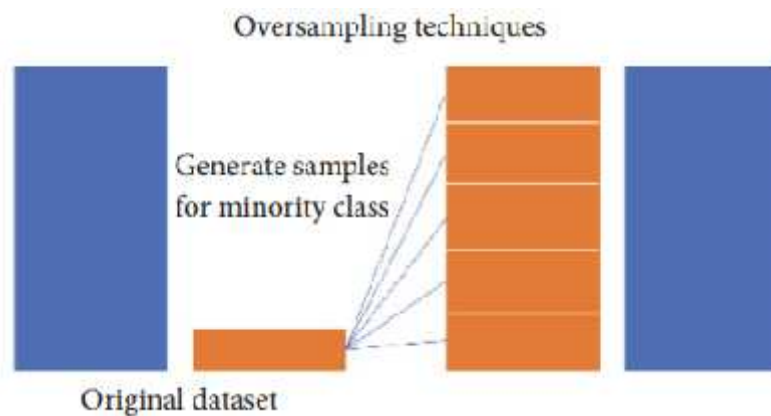


Figure-4: Over sampling Techniques (Shakya, 2018)

Hybrid Technique:- imbalanced data has great potential to cause misclassification because the majority data is more dominant on minority data and the results is misleading to incorrect accuracy. Thus, to control this problem it needs another method so called

hybrid. As (Junsomboon & Phienthrakul, 2017) stated that, this technique is combination of SMOTE and ENN techniques or SMOTE and Tomek Links. This technique is the combination of oversampling and under sampling technique that can solving imbalance data cases and can improve the detection performance of accuracy. In this hybrid technique the following techniques have been performed to balance the imbalanced data and to enhance the accuracy of the prediction.

SMOTEENN: Synthetic minority over-sampling technique and Edited nearest neighbor are two classic resampling algorithms. SMOTE synthesizes samples in the minority class by linear interpolation to increase the number of samples in the minority class, while ENN reduces the number of samples in the majority class by removing noise samples from the majority class (Xu, 2020). ENN removes any example whose class label differs from the class of at least two of its three nearest neighbors. The ENN method removes the instances of the majority class whose prediction made by KNN method is different from the majority class. ENN tends to remove more examples than the Tomek links does, so it is expected that it will provide a more in-depth data cleaning.

SMOTE Tomek: is a hybrid sampling technique. The main idea of SMOTE + Tomek Links is to increase minority class instances by using SMOTE algorithm to create new synthetic instances and decrease majority class instances by using Tomek Links algorithm to eliminate some of majority class instances (Sanguanmak & Hanskunatai, 2016). Both SMOTE and Tomek Link removal approach can be applied for balancing the class distribution. In this process, the original training dataset is first oversampled using SMOTE, and then Tomek Link removal is applied to the resulting dataset producing a balanced dataset.

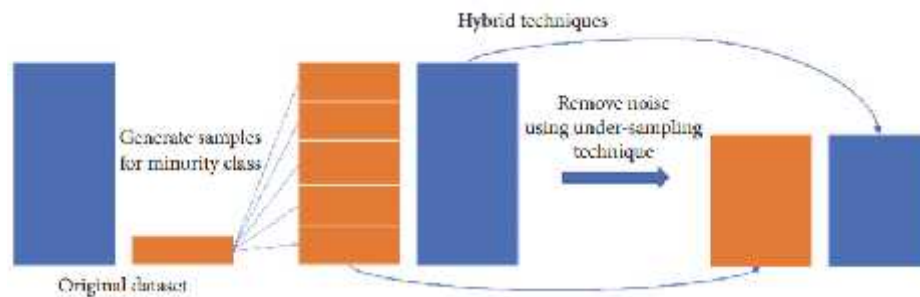


Figure-5: Hybrid Techniques (Shakya, 2018)

Random Under sampling :- used for randomly eliminating most data samples from the dataset to make balance the imbalance dataset. This technique is popular when we are using a huge amount of training datasets. By removing the frequency of majority samples, it improves the runtime and reduces the storage troubles. As (Ganganwar, 2012), the main drawback of under-sampling is that potentially useful information contained in these ignored examples is neglected. There many ways attempt to improve upon the performance of random sampling, such as Tomek links, Condensed Nearest Neighbor Rule and One-sided selection etc.

Random Oversampling:- used to randomly replicate the minority samples to balance the imbalanced dataset. Unlike random under sampling, this approach does not lead to information loss. However, there's a high possibility of overfitting the data since it replicates the minority samples. As per (Ganganwar, 2012) description, since this method replicates existing examples in the minority class, overfitting is more likely to occur while excessive under sampling can affect the loss of some important information contained in the dataset therefore other techniques are needed to curb this problem.

Synthetic Minority Oversampling Technique (SMOTE) is a well-known preprocessing approach for handling imbalanced datasets, where the minority class is oversampled by producing synthetic examples in feature vector rather than data space(Hussein, 2019).

SMOTE faced to two major problems: noise and borderline examples. In the case of this it needs other techniques to eliminate the problem. According to this we apply two types of techniques to give a solution for this problem such as SMOTEENN and SMOTE Tomek. Depending upon the amount of oversampling required, the nearest neighbors of minority samples are randomly selected (Shakya, 2018).

Tomek links are pairs of very close instances that belong to different classes. They're samples near the borderline between classes. By removing sampling of the majority class of each pair, we can increase the space between the two classes and move toward balancing the dataset by deleting those points.

NearMiss under sampling algorithm is a family of undersampling strategies which choose the samples of majority class based on their distance to the minority class (Pang,

2021). Basically, the nearmiss algorithm will include the following points such as: 1st, calculating the distances between all instances of the majority class and the instances of the minority class. 2nd, k instances of the majority class that have the smallest distances to those in the minority class are selected to be retained. 3rd finally if there are n instances in the minority class, NearMiss will result in $k \times n$ instances of the majority class.

Under Sampling with edited nearest neighbor:- the ENN technique is proposed by (Gyamfi & Abdulai, 2019), to remove instances from the majority class whose class label is different from the class label of at least two of its three nearest neighbors. To mean this, removing samples which do not agree with their neighborhood. The main purpose is, increasing the classification accuracy of minority instances rather than majority by removing both the noisy examples as borderline examples and by providing a smoother decision surface.

Under Sampling with Neighbor Hood Cleaning:- the Neighbor Cleaning Rule (NCL) projected by (Laurikkala, 2014), it deals with the majority and minority samples separately when sampling the datasets and it improv the ENN method of the two-classes problems in the following way:

First, the chosen instance is removed when the instance is a member of the majority class and the classification given by its three nearest neighbors is the opposite of the class of the chosen instance.

Second, the nearest neighbors that belong to the majority class are removed when an instance belongs to the minority class and it's misclassified by its three nearest neighbors.

ADASYN Over Sampling Algorithm:- can self-decide the number of artificial data samples that are required to be produced for minority class. They also investigated that ADASYN not only provided a balanced data distribution but also forced the learning algorithm to focus on complex samples in the dataset (Amin, 2016).

2.10 Evaluation Metrics

Evaluation metric plays a critical role in achieving the optimal classifier during the classification training (Hossin & Sulaiman, 2015). Accuracy is one of the metrics for

evaluating classification models and this technique is used by various researcher to measure the accuracy of the imbalanced data classification. According to (Burnaev, 2017), they used accuracy to measure imbalanced data classification. On the other side, using accuracy has its own side effect when the dataset is imbalanced. As per (Kuncheva, 2019), handling imbalanced data sets is difficult because traditional classifiers are designed to maximize a global measure of accuracy, which often results in the minority class being ignored. In addition to this, (Cervantes, 2013) clarified that, the classification accuracy is poor because the learning systems require balance data and the algorithms do not consider the classes distribution. Even so, classifier training algorithms work better when the classes are balanced. In the case of this, minority class are not ignored in this work and to balance the imbalance data, we applied various techniques as we stated on section 2.9.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

TP = True Positives

TN = True Negatives

FP = False Positives

FN = False Negatives

Chapter 3 Research Methodology

3.1 Proposed Model

CNN is constructed from multiple layers and in this work, we used the following layers such as:-

Convolution layer: - is the first layer in CNN and we used the following parameter to extract features from the given input. The first convolution filter size =32, kernel size = 2 and the second convolution filter size=28 and kernel size=3 we did this by try and error and finally we can get a better extracted feature.

Batch normalization: - we used this layer to normalize the output of the previous layers and It applying a transformation that maintains the mean activation close to 0 and the activation standard deviation close to 1.

Max-polling layer: - we used this layer to reducing the size of feature maps and the pool size that we used is 2 to determine dimensions of the pooling regions and to solve overfitting problem.

Regularization technique: - we applied two types of regularization technique such as L2 regularization and Dropout to avoid overfitting problem. For L2 regularization we used default value and the value is $L2=0.01$ to apply penalties on layer parameters or layer activity during optimization. For dropout we used 0.2 (20%) of neurons are randomly neglected during training time and this regularization technique give us a good result. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass.

Flatten layer we used this layer to convert the data into a 1-dimensional array for inputting it to the next layer.

Activation Function we used two types of activation function the first is ReLU and the second is Sigmoid. We used ReLU to eliminate vanishing gradient problem and allowing the models to learn faster and perform better. On the other hand, we used Sigmoid

activation function in the dense layer to convert a real value into one that can be interpreted as a probability.

Epochs: we used 10 epochs for undersampling methods and 15 epochs for oversampling methods because the content of the sampled dataset is different during undersampling and oversampling. During under sampling technique, much of data are removed from the majority class and when we used epochs morethan 10 the model is prone to overfitting and during oversampling technique much of data are added in to the minority class by replicating the existing data or producing the new synthetic one to balance the imbalanced class and the content of sampldata are increased because the minority samples are increased. Thus, when we used more than 15 epoch the model faces overfitting problem. According to this, both epochs provide the best result and the epoch is determine using trial and error in python program.

For optimization purpose we used Adam optimizer due to combination of the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems. The default learning rate is 0.001 but I got the best result using 0.0001.

Generally, the convolution neural network has achieved state-of-the-art performance in many computer vision applications e.g., classification, recognition, detection, etc. However, the global optimization of CNN training is still have a problem. In the case of this, fast classification and training play a key role in the development of the CNN. According to this, we used the following flow chart to train the CNN model with a little bit modification because, (Fu, 2016) clarified that, the training part is offline and the prediction part is online but when we come to our work training and classification have been done through the given offline dataset.

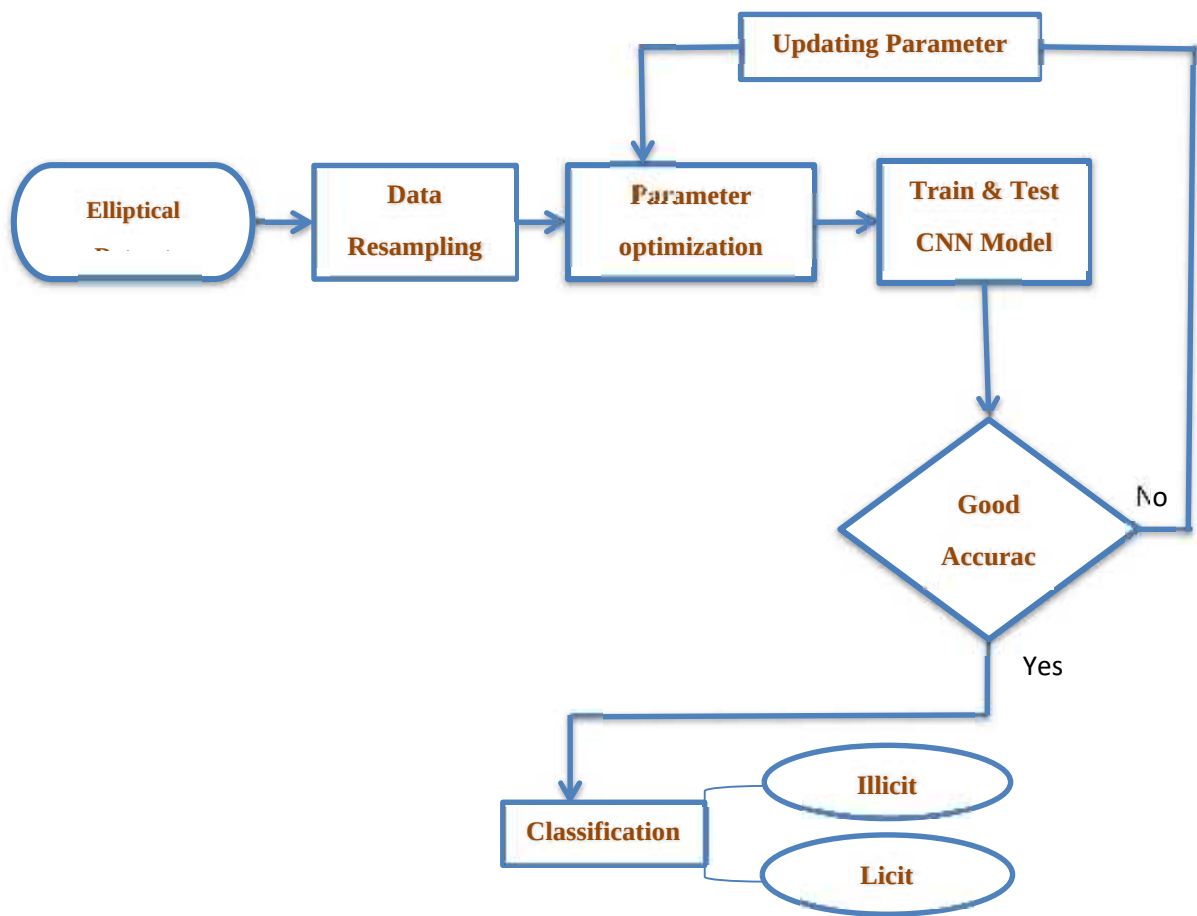


Figure- 8: The illustration of suspicious Money laundering detection system (Fu, 2016)

3.2 Dataset Description

Datasets are an important part of the field of deep learning but gathering real data is one of the cumbersome tasks in most research areas, especially when the data has related to financial transactions because disclosing customer data is strictly forbidden in any financial industries due to the secrecy of customer information. In the case of this, we will use an Elliptical synthetic data set that has been prepared based on real data (Weber, 2018). This dataset was released at Kaggle in 2019, a community of data scientists and machine learners. This dataset was prepared based on Bitcoin transactions with the combination of two categories licit categories (exchanges, wallet providers, miners, licit services, etc.) versus illicit ones (scams, malware, terrorist organizations, ransomware, Ponzi schemes,

etc.). In this dataset, there are 203,769 transaction with 166 features associated with each transaction. Due to intellectual property issues, the authors couldn't provide an exact description of all the features in the dataset. There is a time step associated with each node, representing a measure of the time when a transaction was broadcasted to the Bitcoin network. The time steps, running from 1 to 49, are evenly spaced with an interval of about two weeks. Each time step contains a single connected component of transactions that appeared on the blockchain within less than three hours between each other; no edges are connecting the different time steps.

The first 94 features represent local information about the transaction – including the time step described above, the number of inputs/outputs, transaction fee, output volume and aggregated figures such as average BTC received (spent) by the inputs/outputs and the average number of incoming (outgoing) transactions associated with the inputs/outputs. The remaining 72 features are aggregated features, obtained using transaction information one-hop backward/forward from the center node - giving the maximum, minimum, standard deviation, and correlation coefficients of the neighbor transactions for the same information data (number of inputs/outputs, transaction fee, etc.).

In this dataset, the dataset provider prepared two percent (4,545) of the dataset are labeled class1 (illicit). Twenty-one percent (42,019) of the data are labelled class2 (licit). The remaining transactions are not labeled concerning licit versus illicit but have other features.

Class imbalanced datasets occur in many real-world applications where the class distributions of data are highly imbalanced. As we have seen from the below graph, the class distributions are imbalanced. In the case of this (Somasundaram, 2016) stated that, the classifier gets insufficiently trained and hence provide inaccurate predictions.

The illicit class is highly dispersed, as we have seen from the below graph and to give a solution for this problem, we applied various technique. On the other hand, all class data distributions it looks like the below graph. The below graph showed us, the number of class in combination with number of transactions.

Class 0 Unknown: 157205
Class 1 Illicit: 4545
Class 2 Licit: 42019

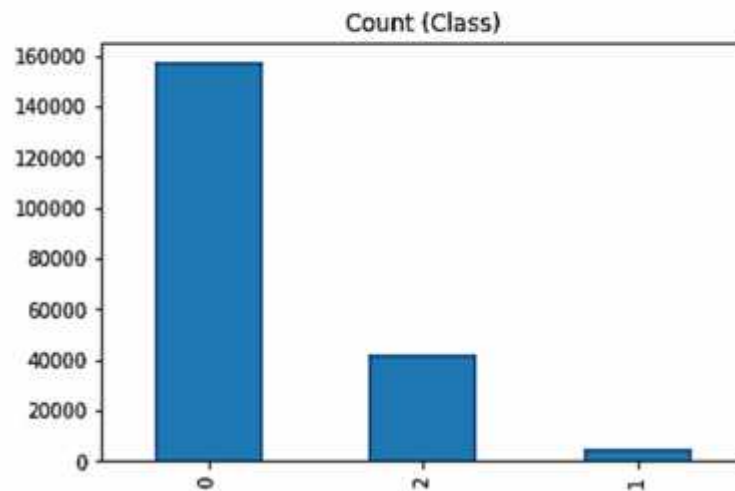


Figure-9: Unknown, Licit verses Illicit graph with values

As we have seen from the above figure the number of unknown transactions is 157205, illicit transactions are 4545 and the Licit transactions are 42019, to mean this the class distributions are high imbalanced. Therefore, data resampling is needed to give a solution to this problem data resampling approach is needed to balance the imbalanced data.

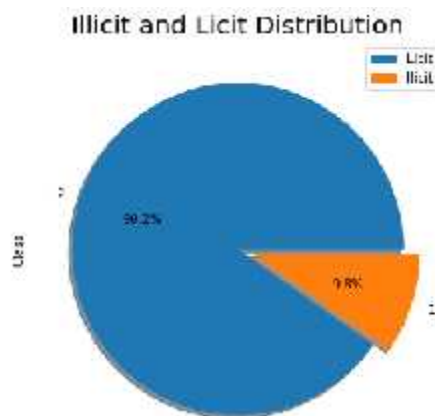


Figure-10: Licit verses Illicit graph

Variable Description in the Dataset

Variable	Type	Description
txId	Int64	Transaction Amounts
time_step	Int64	Time elapsed between each transaction and the first transaction
....		
....		
....		
....		
Class	Int64	The response variable (1=Illicit and 2=Licit)

Table-2: Variable Description

Before going into the implementation part, data analysis is better to visualize the data. According to this, we are using correlation analysis to see the mutual relationship between the variables. Correlation analysis is a statistical technique used to evaluate the strength of the relationships between two quantitative variables. A high correlation means that two or more variables have a strong relationship with each other and it has a more significant impact during the training phase. The correlation matrix of this data set looks like in the below figure.

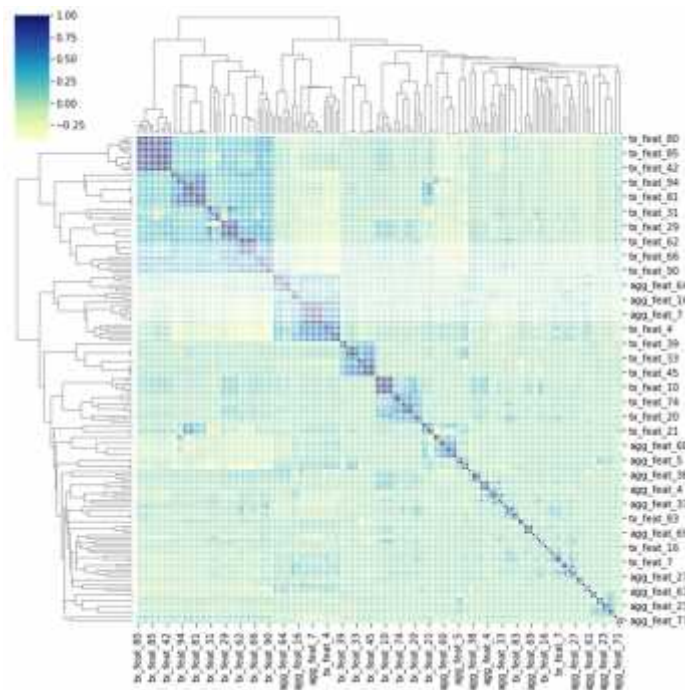


Figure-11: Data Correlation

In correlated data, the change in the magnitude of 1 variable is associated with a change in the magnitude of another variable, either in the same (positive correlation) or in the opposite (negative correlation) direction. Both correlation coefficients are scaled such that they range from -1 to $+1$, where 0 indicates that there is no linear or monotonic association and as we have seen the above picture the blue line show us there is a positive correlation between the data because the correlation coefficient is $+1$.

3.3 Data splitting

Data splitting for training and testing is one of the crucial part to evaluating the proposed models. (Gholamy, 2018), Empirical studies show that the best results are obtained if we use 20-30% of the data for testing, and the remaining 70-80% of the data for training. In the case of this, many researcher splitting the dataset randomly (van Rosendael, 2018), for training set (80%), for testing set (20%) and they can get greater accuracy. It is not only this, (Hemdan, 2020) used 80-20% for experiments and evaluation of the COVIDX-Net. In the case of this, we split the entire dataset into 80-20% for training and testing.

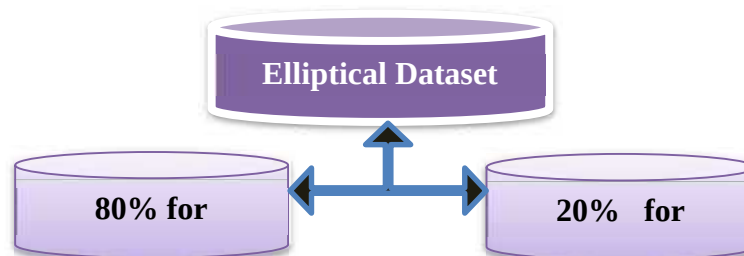


Figure-12: Data Splitting

3.4 Regularization techniques

One of the main problems in CNN is overfitting because (Li, 2019) small amounts of training data is available, overfitting can become a critical issue or CNNs may memorize specific patterns of the training data, leading to poor generalization at test time. To curb this problem, we used different techniques such as: L1, L2 and Dropout regularization.

L1 and L2 regularization techniques are the most important regularization techniques which makes slight modifications to the learning algorithm such that the model generalizes better and controlling overfitting problem. These update the general cost function by adding another term known as the regularization term.

Cost function = Loss (say, binary cross entropy) + Regularization term

Due to the addition of this regularization term, the values of weight matrices decrease because it assumes that a neural network with smaller weight matrices leads to simpler models. Therefore, it will also reduce overfitting to quite an extent. However, this regularization term differs in L1 and L2. As (Phaisangittisagul, 2016) stated that, overfitting still remains one of the major challenges to overcome in machine learning research community. Several approaches have been proposed to mitigate this problem. A widely used method is to add a regularization term to an error function so that the total error function is minimized in the following form.

$$\mathcal{L}_T(\mathbf{W}) = \mathcal{L}_D(\mathbf{W}) + \lambda \mathcal{L}_W(\mathbf{W})$$

where $\mathbf{W}$ is neural network parameters. $\mathcal{L}_T(\mathbf{W})$ is a total loss function and λ is a regularization parameter that controls the tradeoff between $\mathcal{L}_D(\mathbf{W})$ and $\mathcal{L}_W(\mathbf{W})$. $\mathcal{L}_D(\mathbf{W})$ is a sum-of-squares error function between the target output, $y(i)$ and the network output, $h(x(i); \mathbf{W})$

$$\text{Cost function} = \text{Loss} + \frac{\lambda}{2m} * \sum \|w\|^2$$

L2 Regularization

Here, λ is the regularization parameter. It is the hyperparameter whose value is optimized for better results. According to (Ying, 2019), L2 regularization is also known as

weight decay this approach makes the networks prefers to learn features with small weights. To mean this, it forces the weights to decay towards zero (but not exactly zero).

$$Cost\ function = Loss + \frac{\lambda}{2m} * \sum ||w||$$

L1 Regularization

In this regularization technique, to minimize the cost function, we need to set the weights of some features to be zero (Ying, 2019). Unlike L2, the weights may be reduced to zero here. Therefore, it is very useful when we are trying to compress our model. Otherwise, we usually prefer L2 over it.

According to (Gu , 2006), Droopout is a recently proposed regularizer to fight against over-fitting. It is a regularization method that stochastically sets to zero the activations of hidden units for each training case at training time. The goal of dropout is, reducing unnecessary feature dependencies in the network, allowing it to be simpler and improves its generalization abilities (reduces overfitting). In simple terms, Dropout randomly selects some nodes and removes them along with all their incoming and outgoing connections as shown below.

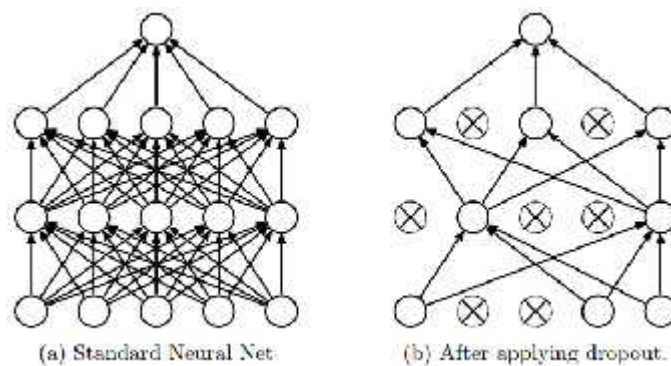


Figure-13: Dropout Regularization(Ikasari, 2016)

3.6 Training and testing phase

After re-sampling the dataset for the proposed model, we set a value for Dropout respectively, and then we passed the re-sampled training set to each model as the training data. Thus, the models learned different patterns in the re-sampled training data. Then, we used the test set, which we separated before while splitting the whole dataset, to test the performance of the model.

3.7 Performance evaluation of the proposed model

The last step of the detection process is the performance evaluation of the model. In this thesis, we evaluated the performance of the models using different resampling and regularization techniques. Finally, the result of each technique described on next chapter.

CHaPTETR 4 Experment Result

4.1 Without Resampling and Regularization

Before we applied the selected resampling and regularization technique, we feed the imbalanced data to the proposed model to see the difference between the model with and without regularization and resampling techniques. To train this model, we used 10 epochs. when the number of epochs is more than ten it prone to overfitting and the third epochs have better result than other epochs and can generate 36, 833 total parameters and all parameters are properly trained. Therefore, the untrained parameters of this model are zero. We used 37,251 samples for training purpose and 9,313 samples are for validation purpose and to compile this model, we used (Kingma & Ba, 2015) Adam, it is an adaptive learning rate optimization algorithm that's has designed specifically for training deep neural networks and we used 0.0001 for learning rate and we can get a better result instead of the default, for loss function we used binary Cross-entropy to measures the performance of a classification model whose output is a probability value between 0 and 1. For metrics purpose, we used accuracy as we stated on section 2.10. We used Verbose = 1 to check over fitting, to display progress bar and one line per epoch.

Without Resampling and Regularization						
Training Sample	Validation sample	Proposed Technique	Total Epoch (10)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
37,251	9,313	Only CNN model	Best epoch is 3	0.1257	0.9615 (96.15)	0.9160

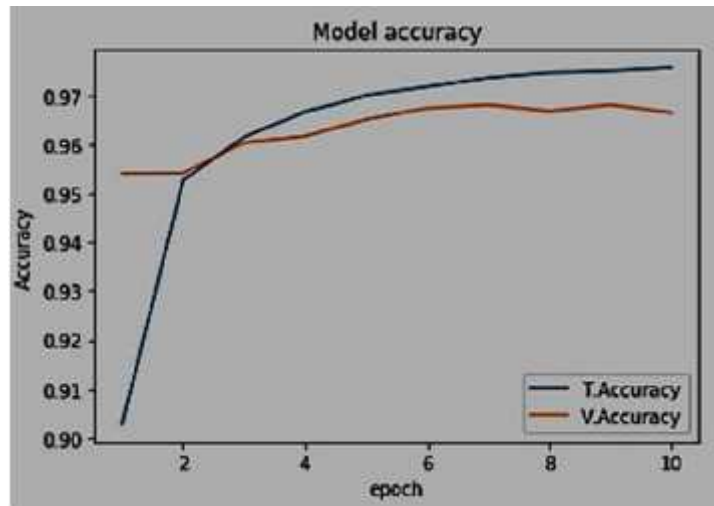
Table-3: Accuracy Result Without Resampling and Regularization

The below image showed us, the training accuracy is more closest to the validation accuracy rather than others. To mean this, the overfitting and underfitting problem in this epochs are very less but the accuracy need further improvment.

```
Train on 37251 samples, validate on 9313 samples
Epoch 1/10
37251/37251 [=====] - 6s 163ms/step - loss: 0.2556 - accuracy: 0.9030 - val_loss: 0.1920 - v
al_accuracy: 0.9540
Epoch 2/10
37251/37251 [=====] - 6s 190ms/step - loss: 0.1915 - accuracy: 0.9528 - val_loss: 0.1758 - v
al_accuracy: 0.9542
Epoch 3/10
37251/37251 [=====] - 6s 146ms/step - loss: 0.1254 - accuracy: 0.9615 - val_loss: 0.1614 - v
al_accuracy: 0.9624
```

Figure-14: Best Training Accuracy and Validation Accuracy at epoch 3.

The below graph showed us, the training accuracy progress is very good when we compare with the validation accuracy and in epochs 3 overfitting and underfitting problems can be minimized.



During training time there is a model loss as we have seen from the below depicted graph.

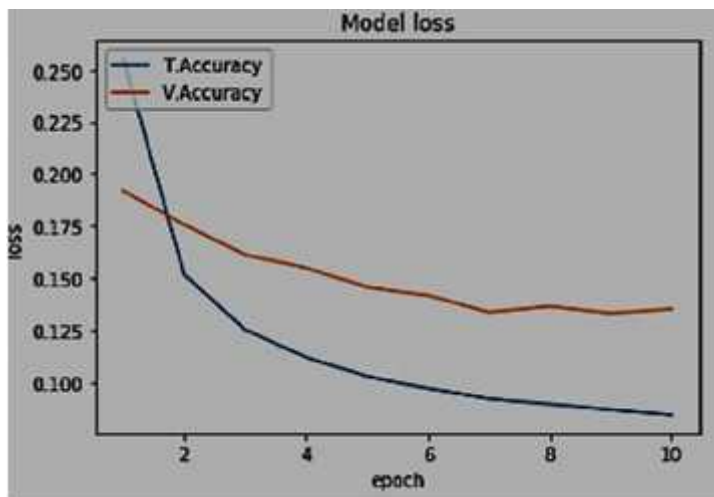


Figure-15: Model Accuracy and Loss Without Resampling and Regularization

4.2 Under sampling

As we described before, the imbalance data should be a balance to control overfitting in the case of this, various types of techniques have been applied in under-sampling techniques, for instance, Random under-sampling, Near Miss Under-sampling, Tomek Links under-sampling, Under-sampling with Edited Nearest Neighbor Rule, Under-sampling with Neighborhood Cleaning Rule. Under-sampling methods delete instances from the over majority class in different ways. The most obvious way is to do delete instances randomly to make it balance the imbalanced as we have seen from the below figure.

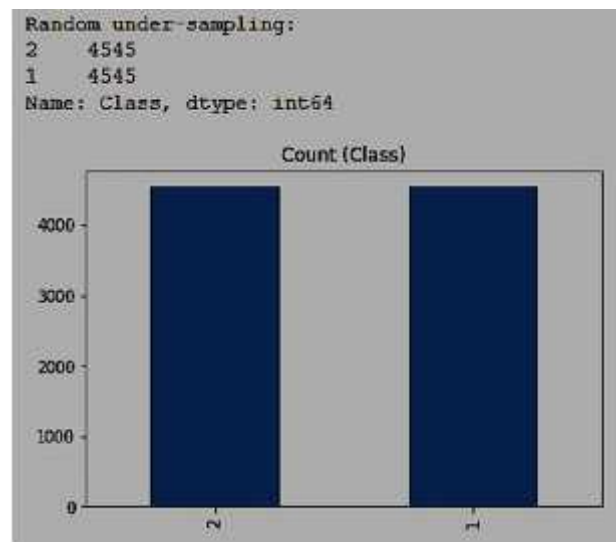


Figure-16: Random Under Sampling

This balanced dataset is given to the proposed model and various types of regularization techniques are incorporated to curb overfitting and under fitting problems such as L2 regularization and dropout. During the training period of time we used 10 epochs and epoch 10 have a better result in terms of overfitting and underfitting management but the detection performance accuracy is very small (90.94) in the case of this, another techniques are needed. Using this epochs we can get 6,881 parameters and all parameters are trained. We used 8,144 samples for training purpose and 9,313 for validation one. Thus, the rest techniques are the same as to the previous one for instance learning rate, loss function, performance metrics and verbosity.

As we have seen from the below result epoch 10 training accuracy is more closest to validation accuracy from all epochs. To mean this, overfitting and underfitting hav been rectified a littlebit.

Random Under Sampling						
Training Sample	Validati on sample	Proposed Technique	Total Epoch (10)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
8,144	9,313	Random sampling Under	Best Epoch is 10	0.3276	0.9116 (91.16)	0.9033

Table-4: Random Under Sampling Result

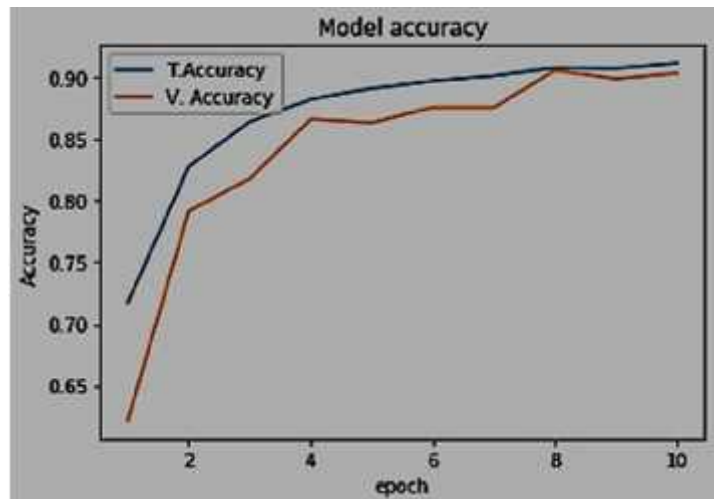
```

8144/8144 [=====] - 1s 170us/step - loss: 0.3463 - accuracy: 0.9079 - val_loss: 0.3739 - val
accuracy: 0.9036
Epoch 9/10
8144/8144 [=====] - 1s 165us/step - loss: 0.3253 - accuracy: 0.9074 - val_loss: 0.3856 - val
accuracy: 0.8986
Epoch 10/10
8144/8144 [=====] - 1s 168us/step - loss: 0.3077 - accuracy: 0.9116 - val_loss: 0.3669 - val
accuracy: 0.9033

```

Figure-17: Best Training Accuracy and Validation Accuracy at epoch 10.

The below graph showed us the training and validation accuracy can goes parallely with a little bit difference. Therefore, to improve the performance of the accuracy and we need a similar techniques that can curb this problem.



As we have seen from the below graph, the loss of accuracy is less than the loss of validation.

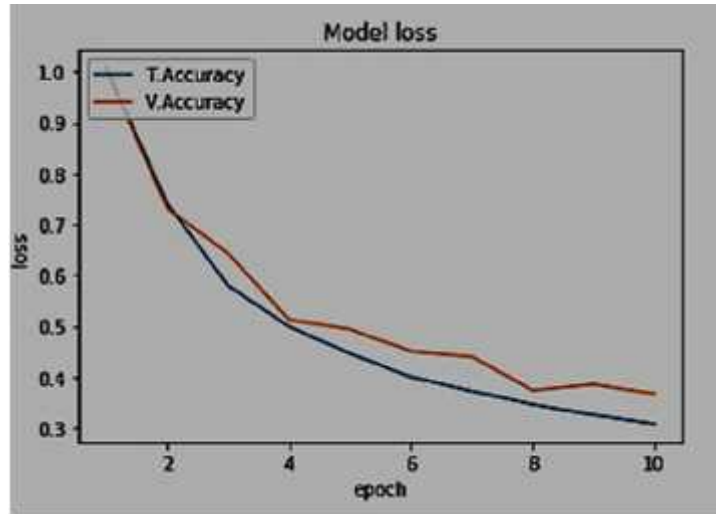


Figure-18: Model Accuracy and Loss for Random under Sampling

4.1.1 Near Miss Under-sampling

Near Miss under-sampling is a kind of under-sampling technique which used to control overfitting and underfitting by solve the issue of potential information loss based on the nearest neighbor algorithm. This data is give to the proposed model and the model have been trained by 7, 535 trainable parameters. In addition to this, for training purpose we used 37,251 samples and for validation purpose we used 9,313 samples. According to the generated result epoch 6 has 0.9701 training accuracy and 0.97 for validation accuracy to mean this, controlling overfitting and underfitting problem can be managed properly on theis state. On the other hand various techniques have been applied just like the previous one for instance learning rate, loss function, performance metrics and verbosity.

Near Miss Under-Sampling						
Training Sample	Validation Sample	Proposed Technique	Total Epoch (10)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
37251	9,313	Near Miss Under sampling	The best epoch is 6	0.1127	09701 (97.01)	0.1420

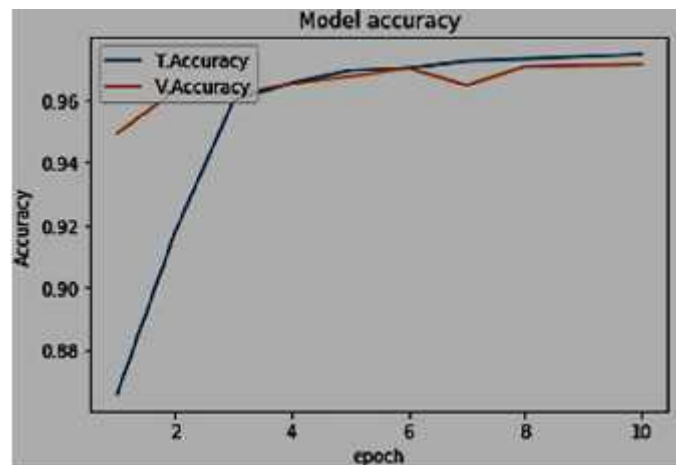
Table-5: Near Miss Under Sampling Result

As we have seen from the below result, epoch 6 has a better result than other epochs because the training accuracy and the validation accuracy is almost the same.

```
Epoch 4/10
37251/37251 [-----] - 18s 482us/step - loss: 0.1378 - accuracy: 0.9654 - val_loss: 0.1608 -
val accuracy: 0.9651
Epoch 5/10
37251/37251 [-----] - 16s 432us/step - loss: 0.1217 - accuracy: 0.9693 - val_loss: 0.1486 -
val accuracy: 0.9675
Epoch 6/10
37251/37251 [-----] - 16s 429us/step - loss: 0.1127 - accuracy: 0.9701 - val_loss: 0.1426 -
val accuracy: 0.9700 0.97 - ETA: 0s - loss:
```

Figure-19: Best Training Accuracy and Validation Accuracy at epoch 6.

The below graph showed us the training accuracy and the validation accuracy almost the same but the it needs another technique that can improve the performance of accuracy.



The below graph showed us, the loss of the Training accuracy and validation accuracy of the proposed model loss.

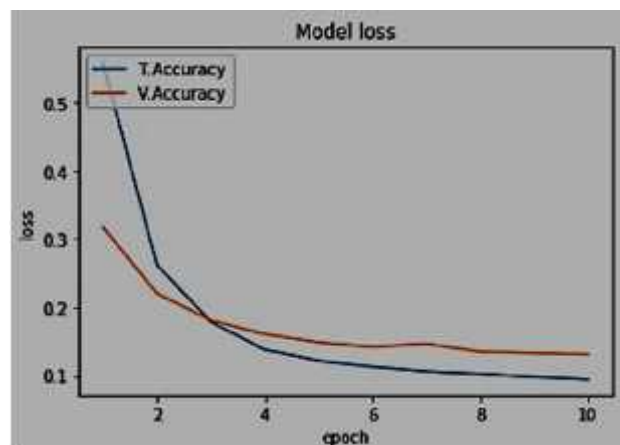


Figure-20: Model Accuracy and Loss for Near Miss under Sampling

4.1.2 Tomek links Under sampling

Tomek link under sampling removal is another type of data resampling techniques and the data processed by this technique is given to the proposed model. This model contains 41,025 parameters and all parameters are trained without any remainder. The epochs that we used for this training is 10 and 37,080 samples of dtas are for training purpose and 9,313 samples are for validation one. To control overfitting we used dropout (0.2) and L2 regularization. Epoch 2 has the best result in terms of overfitting and underfitting from the rest of nine epochs and the performance accuracy is 0.9634 to mean this it needs another technique that can increase the performance of the model in combination with controlling over and underfitting problem.

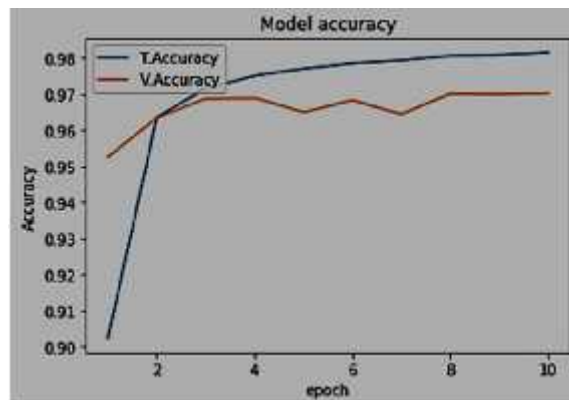
Tomek Links Under-Sampling Result						
Training samples	Validation samples	Proposed Technique	Epoch	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
37,080	9,313	Tomek Link Under sampling	10	0.1749	0.9634 (96.34)	0.9634

Table-6: Tomek Links Under-Sampling Result

Training accuracy and validation accuracy are the same on epoch 2 this indicates that overfitting and underfitting problems are completely solved on this state.

```
Train on 37080 samples, validate on 9313 samples
Epoch 1/10
37080/37080 [=====] - 17s 464us/step - loss: 0.6335 - accuracy: 0.9025 - val_loss: 0.3120 -
val_accuracy: 0.9525
Epoch 2/10
37080/37080 [=====] - 15s 405us/step - loss: 0.1978 - accuracy: 0.9634 - val_loss: 0.1936 -
val_accuracy: 0.9634
Epoch 3/10
```

Figure-21: Best Training Accuracy and Validation Accuracy at epoch 2.



The below graph showd us, the loss of the Training accuracy and validation accuracy of the proposed model loss.

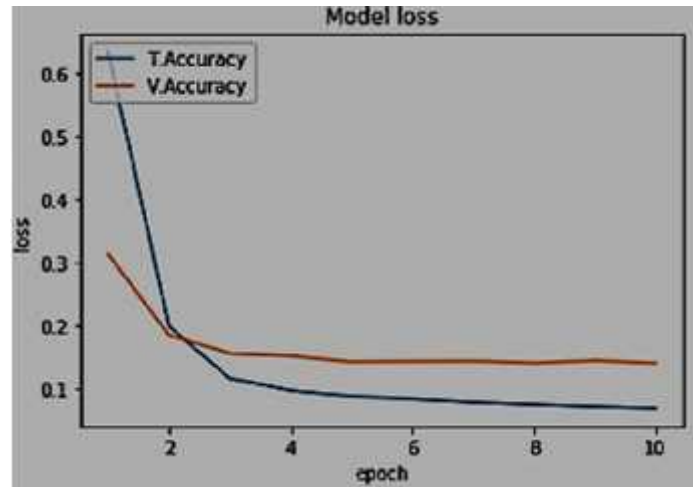


Figure-22: Model Accuracy and Loss for Tomek Links Under-Sampling.

4.1.3 Under-sampling with Edited Nearest Neighbor Rule

This technique is classified in under resampling technique which used to mitigate over and under fitting problem like the above technique. In this technique we used majority sampling strategy to train the data. The proposed model contains 35,521 trainable parameters within 10 epochs and the best epoch is epoch 2 because the difference between training accuracy and validation accuracy is very narrower. we used 35,489 samples to train the data and 9,313 samples are used for validation purpose. In addition to this, regularization techniques have been applied.

Under-sampling with Edited Nearest Neighbor Rule						
Training samples	Validation samples	Proposed Technique	Total Epoch (10)	T.Accuracy validation loss	Model accuracy	Validation Accuracy
35,489	9,313	Edited Nearest Neighbor Rule	10	0.1749	0.9557 (95.57)	0.9576

Table-7: Under-Sampling with Edited Nearest Neighbor Rule Result

Training accuracy is closest to validation accuracy within epoch 2 from other epochs as we have seen from the below image.

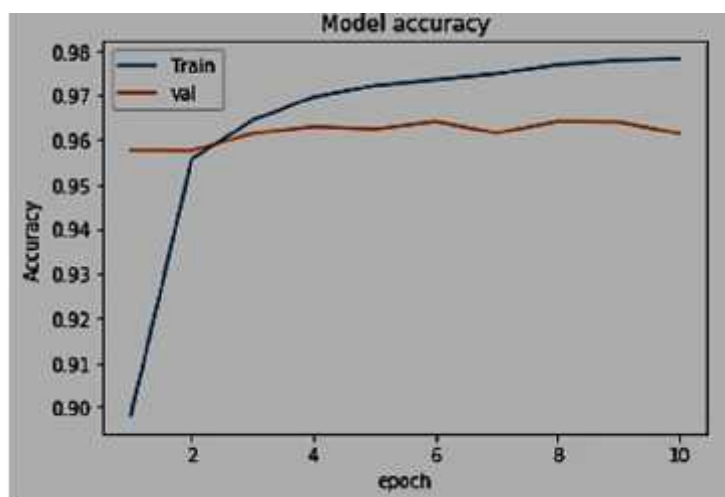
```

Train on 35489 samples, validate on 9313 samples
Epoch 1/10
35489/35489 [=====] - 6s 161ms/step - loss: 0.4959 - accuracy: 0.8982 - val_loss: 0.2979 - v
al accuracy: 0.9577
Epoch 2/10
35489/35489 [=====] - 5s 151ms/step - loss: 0.2009 - accuracy: 0.9557 - val_loss: 0.1809 - v
al accuracy: 0.9579
Epoch 3/10
35489/35489 [=====] - 6s 158ms/step - loss: 0.1301 - accuracy: 0.9644 - val_loss: 0.1588 - v

```

Figure-23: Best Training Accuracy and Validation Accuracy at epoch 2.

The model accuracy graph showed us, the accuracy and the validation accuracy.



The below graph illustrate the loss of model accuracy and model validation.

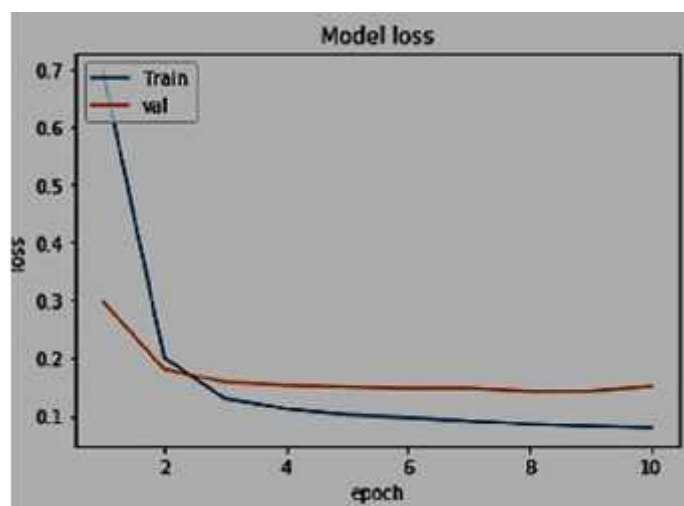


Figure-24: Model Accuracy and Loss for Edited Nearest Neighbor

4.1.4 Under-sampling with Neighbor hood Cleaning Rule

We used this technique to resample the majority class of undersampling and to train this model we used 24,065 parameters and all parameters are trained with 10 epochs and epoch 3 has a better result interms of controlling overfitting and under fitting because the difference between the two accuracies are very small when we compare to the other one but the highest accuracy is generated on epoch 10 and the result is 0.9796. for training purpose we used 35,701 and for validation purpose we used 9,313 samples and various types of regularization techniques have been applied for instance dropout 0.4 and for kernel regularizer we used L1 regularization technique.

Under-sampling with Neighborhood Cleaning Rule						
Training samples	Validati on samples	Proposed Technique	Total Epoch (10)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
35,701	9,313	Neighborhood Cleaning Rule	10	0.1707	0.9666 (96.66)	0.9635

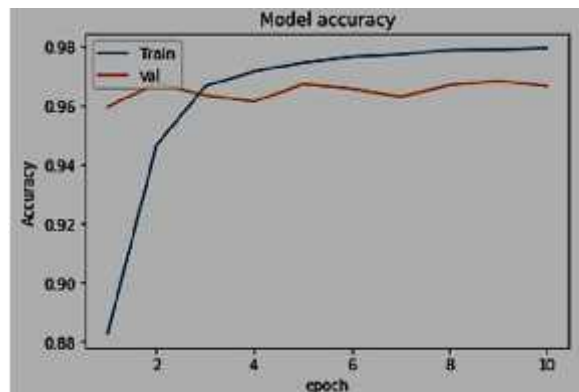
Table-8: Under-sampling with Neighborhood Cleaning Rule

As we have seen from the below graph, epoch 3 is better than from the rest of epochs.

```
Train on 35701 samples, validate on 9313 samples
Epoch 1/10
35701/35701 [=====] - 7s 196us/step - loss: 4.0027 - accuracy: 0.8828 - val_loss: 1.3065 - v
al_accuracy: 0.9596
Epoch 2/10
35701/35701 [=====] - 5s 134us/step - loss: 0.4621 - accuracy: 0.9466 - val_loss: 0.2032 - v
al_accuracy: 0.9676
Epoch 3/10
35701/35701 [=====] - 5s 137us/step - loss: 0.1707 - accuracy: 0.9666 - val_loss: 0.1857 - v
al_accuracy: 0.9635
```

Figure-25: Best Training Accuracy and Validation Accuracy at epoch 3.

The below graph showed us, the model accuracy of the proposed model.



The model loss during the training period is depicted under the below graph.

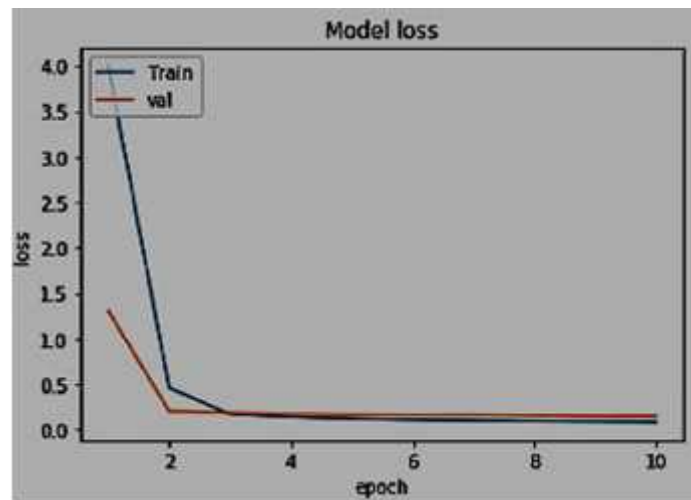


Figure-26: Model Accuracy and Loss for Neighbor hood Cleaning Rule

4.2 Over sampling

Over sampling is another way of resampling techniques under this technique various types of resampling techniques are included such as Random over sampling, SMOTE over sampling, ADASYN oversampling, hybrid oversampling (the combination of under sampling and oversampling) have been applied as we have seen from the below section.

4.2.1 Random Over Sampling

This is one of the resampling techniques that can over-sampling the dataset by duplicating some of the original samples of the minority class. To train the model we used 24,065 parameters and all parameters are properly trained without any reminder within 10 epochs. Epoch 10 has a better performance when we compare to the rest of epochs to combat over and under fitting problem in colabration with regularizer techniques forinstance we used kernel regularizer L1 instade of L2 because of best performance. In addition to this we used dropout 0.2 to reduce overfitting. For training purpose we used 66,358 training sample and for validation purpose is 9,313.

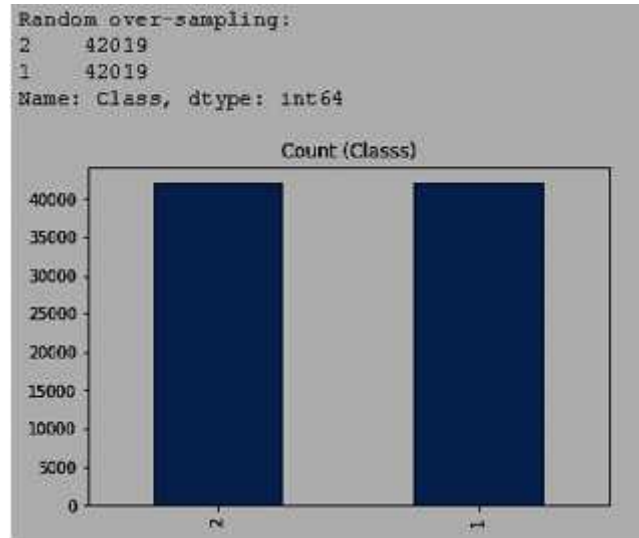


Figure-27: Random Over Sampling

Random Over Sampling						
Training samples	Validation samples	Proposed Technique	Total Epoch (10)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
66,358	9,313	Random Over Sampling	10	0.1188	0.9661 (96.61)	0.9534

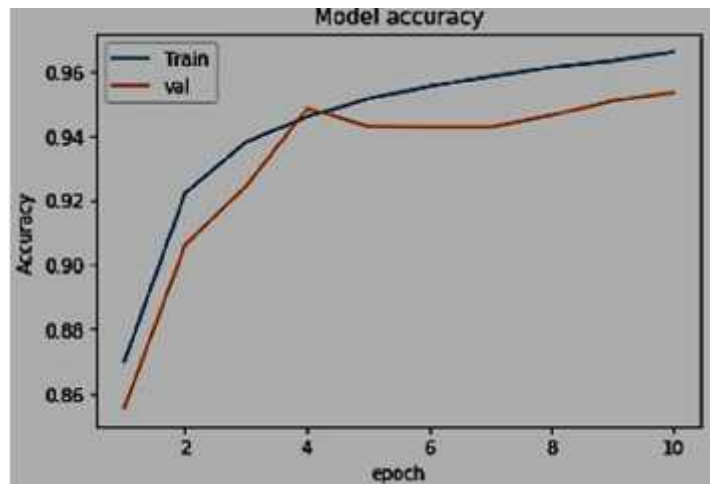
Table-9: Random Over Sampling Result

Epoch 10 is better than other epochs as we have seen from the below depicted graphs because the difference between the validation accuracy and training accuracy relatively better than others.

```
Epoch 9/10
66358/66358 [=====] - 9s 136us/step - loss: 0.1325 - accuracy: 0.9613 - val_loss: 0.1781 - val_accuracy: 0.9463
Epoch 9/10
66358/66358 [=====] - 9s 138us/step - loss: 0.1252 - accuracy: 0.9612 - val_loss: 0.1692 - val_accuracy: 0.9508
Epoch 10/10
66358/66358 [=====] - 9s 138us/step - loss: 0.1188 - accuracy: 0.9661 - val_loss: 0.1642 - val_accuracy: 0.9534
```

Figure-28: Best Training Accuracy and Validation Accuracy at epoch 10

The below model accuracy graph showed us, how the training accuracy and validation accuracy are goes side to side during the training time.



The below graph showed us, the model loss during the training time.

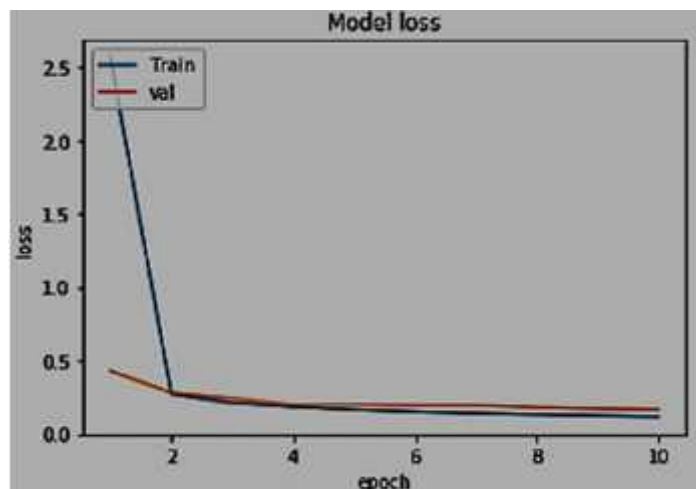


Figure-29: Model Accuracy and Loss for Over Sampling

4.2.2 SMOTE over Sampling

SMOTE is a kind of resampling techniques which might connect inliers and outliers. To applied this techniques we used 24,065 parameters to train the model. In this model various types of regularization techniques are applied for instance kernel regularizer and dropout can be mentioned with in 10 epochs. Epoch 10 has the highest accuracy when we compare to others in terms of controlling over and under fitting problem. For training purpose we used 66,358 samples and for training purpose is 9,313. For regularization we used L1 kernel regularizer and dropout 0.2

SMOTE Over Sampling						
Training samples	Validation samples	Proposed Technique	Total Epoch (10)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
66,358	9,313	SMOTE Over Sampling	10	0.0730	0.9774 (97.74)	0.9627

Table-10: SMOTE Over Sampling Result

As we have seen from the below figure there is a little bit difference between training accuracy and validation accuracy that indicates overfitting.

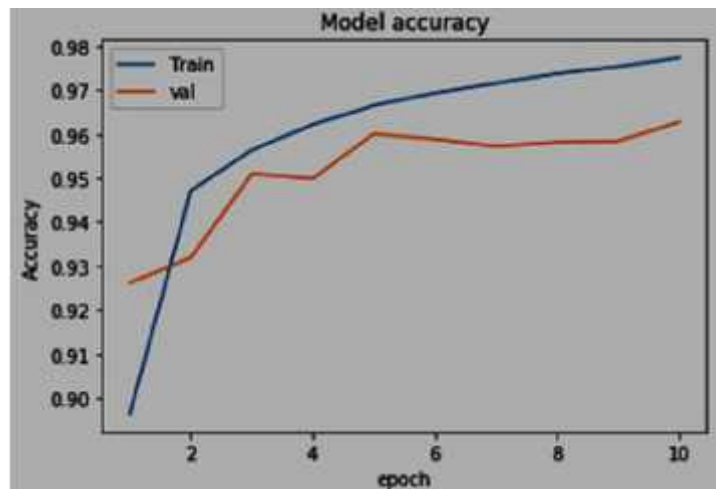
```

Epoch 8/10:
66358/66358 [-----] - 9s 134ms/step - loss: 0.0842 - accuracy: 0.9738 - val_loss: 0.1551 - v
al accuracy: 0.9581
Epoch 9/10:
66358/66358 [-----] - 9s 133ms/step - loss: 0.0781 - accuracy: 0.9753 - val_loss: 0.1547 - v
al accuracy: 0.9583
Epoch 10/10:
66358/66358 [-----] - 9s 134ms/step - loss: 0.0730 - accuracy: 0.9774 - val_loss: 0.1467 - v
al accuracy: 0.9627

```

Figure-30: Best Training Accuracy and Validation Accuracy at epoch 10

The model accuracy graph showed us, how the training accuracy and validation accuracy are plotted on the proposed SMOTE over sampling techniques.



During the training the training phase there is the training model loss as we have seen from the below depicted graph.

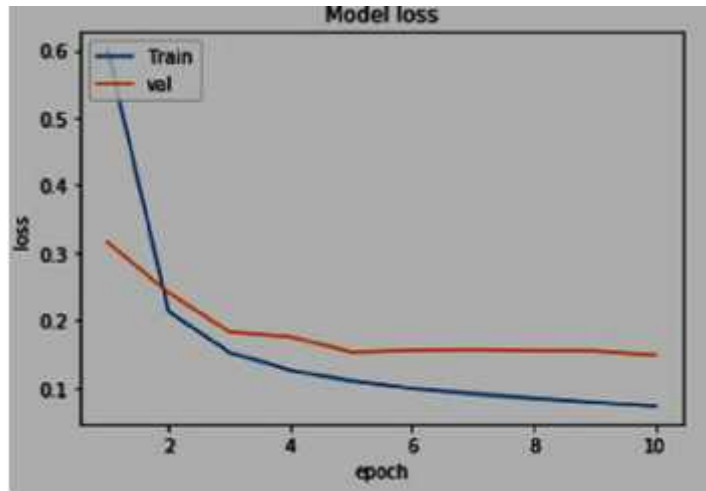


Figure-31: Model Accuracy and Loss for SMOTE Sampling

4.2.3 ADASYN Over Sampling

This resampling technique derived from SMOTE and the main difference is that ADASYN will bias the sample space (that is, the likelihood that any particular point will be chosen for duping) towards points which are located not in homogenous neighborhoods. We used this technique to train the proposed model by feeding 57,025 trainable parameters with out any remainder. In addition to this, we used 15 epochs and epoch 15 has the best performance accuracy instade of others and 66,295 samples of datas are feed to the proposed model and 9,313 samples are for validation and for regularization techniques we used L2 kernel regularizer and dropout.

ADASYN Over Sampling						
Training samples	Validati on samples	Proposed Technique	Total Epoch (15)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
66,295	9,313	ADASYN over Sa mpling	15	0.0478	0.9887 (98.87)	0.9566

Table-11: ADASYN Over Sampling Result

As we have seen from the below image epoch 15 is better than from other epochs in terms of training accuracy with validation accuracy and training loss with validation loss.

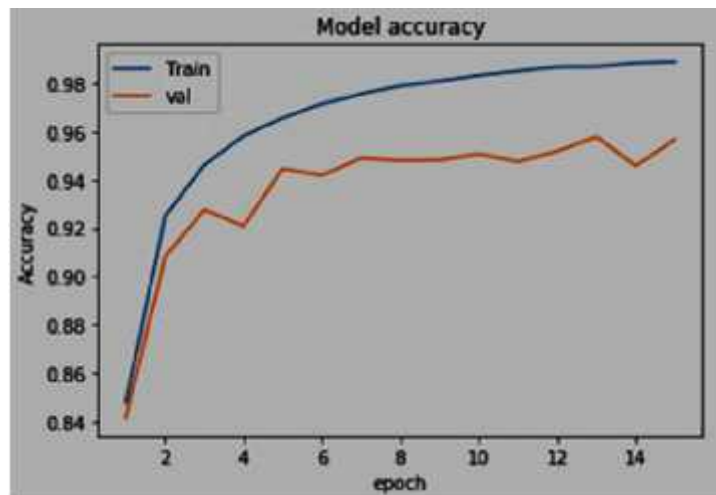
```

Epoch 13/15
66295/66295 [=====] - 34s 509us/step - loss: 0.0543 - accuracy: 0.9868 - val_loss: 0.3111 -
val accuracy: 0.9577
Epoch 14/15
66295/66295 [=====] - 33s 502us/step - loss: 0.0503 - accuracy: 0.9883 - val_loss: 0.3364 -
val accuracy: 0.9457
Epoch 15/15
66295/66295 [=====] - 33s 501us/step - loss: 0.0478 - accuracy: 0.9887 - val_loss: 0.3462 -
val accuracy: 0.9566

```

Figure-32: Best Training Accuracy and Validation Accuracy at epoch 15

The following graph showed us, the model accuracy and validation accuracy of the proposed model within the predefined epoch.



The below graph showed us, the model training loss and validation loss on the predefined epochs.

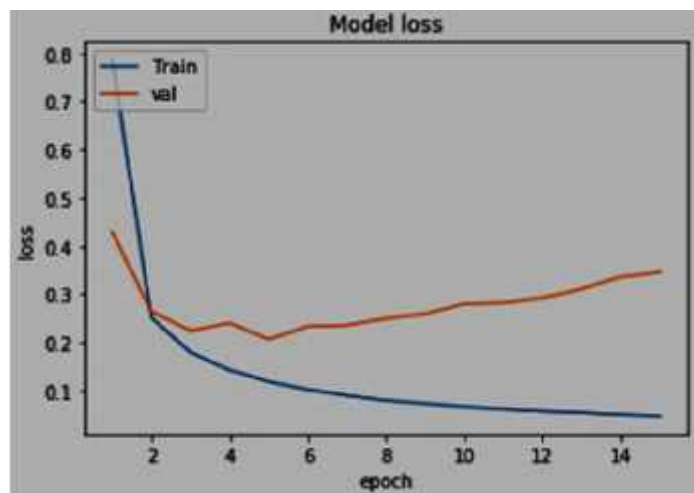


Figure-33: Model Accuracy and Loss for ADASYN Sampling

4.2.4 Hybrid (combination of over sampling and Under sampling techniques)

4.2.4.1 SMOTETomek

SMOTETomek: is the combination of under sampling and over sampling techniques which used to control overfitting. We applied this technique on the proposed model and to train this mode we feed 19,905 trainable parameters and all parameters are properly trained without any remainedr and 57,850 samples of data are applied for traing purpose and for validation purpose we used 13,970 with in 15 epochs and the best epoch is from epoch 1 up to epoch 5 in terms of controlling overfitting and under fitting but the highest performance accuracy is epoch 15. In addition to this we used a regularization techniques so called L2. kernel regularization and dropout.

Hybrid Over Sampling						
Training samples	Validati on samples	Proposed Technique	Total Epoch (15)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
64,281	16,071	SMOTETomek	15	0.0455	0.9871 (98.71)	0.9655

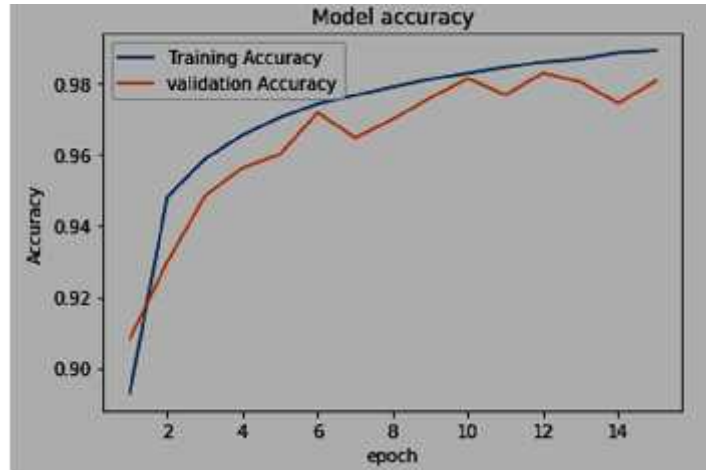
Table-12: SMOTETomek Result

. As we have seen the below image epoch 15 is better than from other epochs in terms of training accuracy.

```
Epoch 13/15  
64281/64281 [=====] - 10s 152us/step - loss: 0.0461 - accuracy: 0.9870 - val_loss: 0.0605 - val_ accuracy: 0.9805  
Epoch 14/15  
64281/64281 [=====] - 9s 147us/step - loss: 0.0406 - accuracy: 0.9888 - val_loss: 0.0748 - val_a accuracy: 0.9746  
Epoch 15/15  
64281/64281 [=====] - 11s 177us/step - loss: 0.0377 - accuracy: 0.9893 - val_loss: 0.0622 - val accuracy: 0.9808
```

Figure-34: Best Training Accuracy and Validation Accuracy at epoch 15

As we have seen the below graph, from epoch 1 up to epoch 6 has a good result to combat overfitting and underfitting problem but the highest performance accuracy is stated on epoch 15.



The below graph showed us, the proposed model of training loss and validation loss on the predefined epochs.

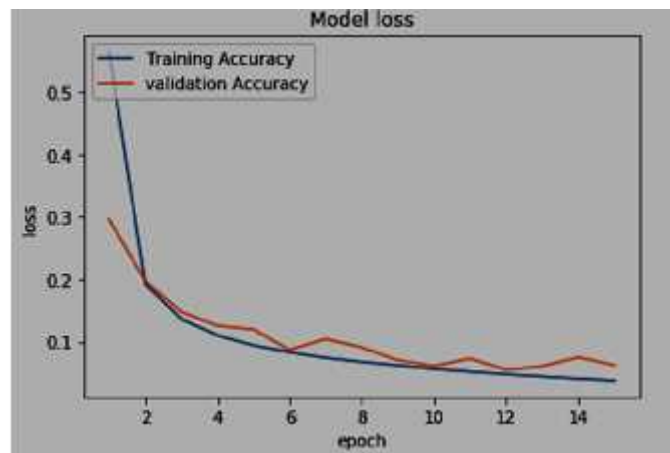


Figure-35: Model Accuracy and Loss for SMOTETomke

4.2.4.2 SMOTEENN

This is another types of hybrid resampling technique which used to control overfitting by using Edited Nearest Neighbors Rule. We applied this technique on the proposed model to combat overfitting in the case of this we used 63,648 samples for traing purpose and 9313 samples for the case of validation. We used 15 epoch and the best epoch to control overfitting is epoch 15. In addition to data resampling technique we used L2 data regularization technique.

Hybrid Over Sampling						
Training samples	Validation samples	Proposed Technique	Total Epoch (15)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
64,281	16,071	SMOTEENN	15	0.0455	0.9893 (98.93)	0.9808

Table-13: SMOTEENN Result

As we have seen the below image epoch 15 is better than from the rest of epochs in terms of training accuracy result

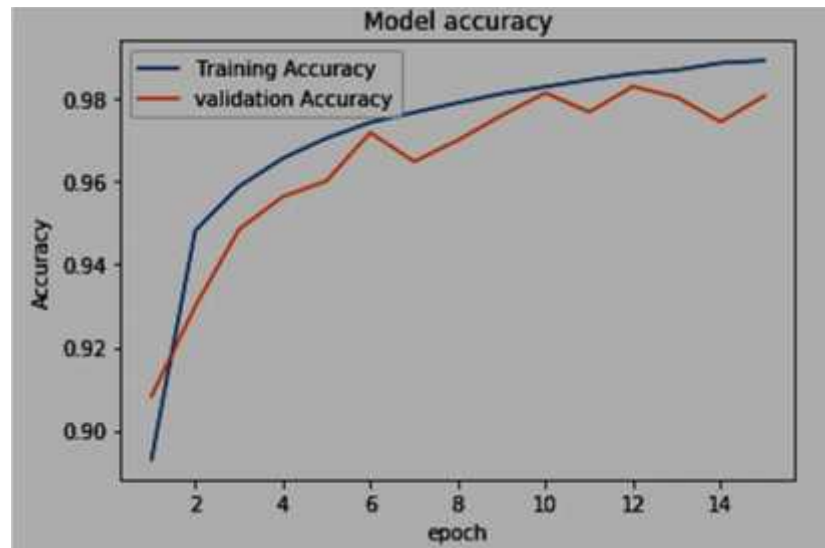
```

Epoch 13/15
64281/64281 [=====] - 10s 152ms/step - loss: 0.0441 - accuracy: 0.9870 - val_loss: 0.0605
- val accuracy: 0.9805
Epoch 14/15
64281/64281 [=====] - 9s 147ms/step - loss: 0.0406 - accuracy: 0.9886 - val_loss: 0.0748 -
val accuracy: 0.9746
Epoch 15/15
64281/64281 [=====] - 11s 177ms/step - loss: 0.0377 - accuracy: 0.9893 - val_loss: 0.0620
- val accuracy: 0.9808

```

Figure-36: Best Training Accuracy and Validation Accuracy at epoch 15

As we have seen from the below graph, this technique is better than other techniques in terms of controlling overfitting problem and accuracy performance result.



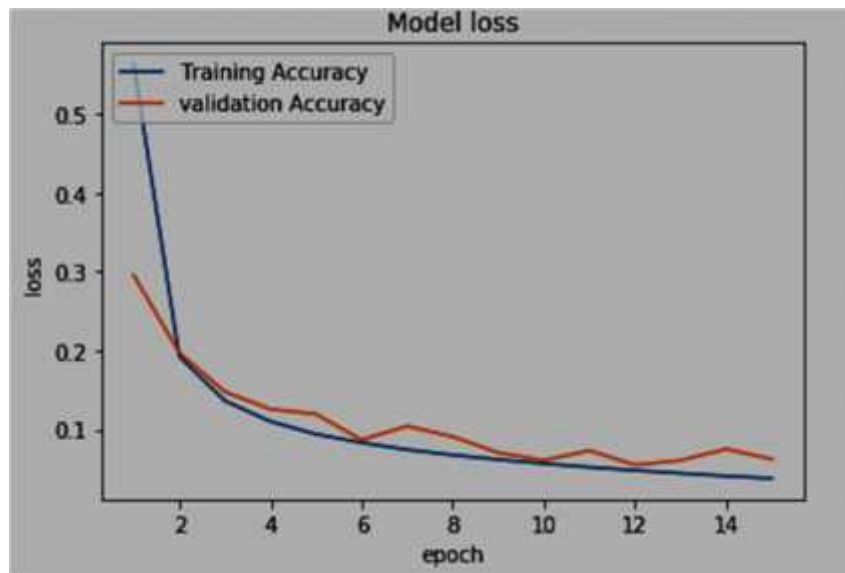


Figure-37: Model Accuracy and Loss for SMOTEENN

All techniques result on the proposed model						
Training Sample	Validation sample	Proposed Technique	Total Epoch (10/15)	T.Accuracy Validation loss	Model accuracy	Validation Accuracy
37,251	9,313	Only CNN model	Best epoch is 3	0.1257	0.9615 (96.15)	0.9160
8,144	9,313	Random Under sampling	Best Epoch is 10	0.3276	0.9116 (91.16)	0.9033
37,251	9,313	Near Miss Under sampling	The best epoch is 6	0.1127	0.9701 (97.01)	0.1420
37,080	9,313	Tomek Link Under sampling	The best epoch is 10	0.1749	0.9634 (96.34)	0.9634
35,489	9,313	Edited Nearest Neighbor Rule	The best epoch is 10	0.1749	0.9557 (95.57)	0.9576
35,701	9,313	Neighborhood Cleaning Rule	The best epoch is 10	0.1707	0.9666 (96.66)	0.9635
66,358	9,313	Random Over Sampling	The best epoch is 10	0.1188	0.9661 (96.61)	0.9534
66,358	9,313	SMOTE Over Sampling	The best epoch is 10	0.0730	0.9774 (97.74)	0.9627
66,295	9,313	ADASYN over Sampling	The best epoch is 15	0.0478	0.9887 (98.87)	0.9566
57,850	9,313	SMOTTOMEK	15	0.0455	0.9871 (98.71)	0.9655
64,281	16,071	SMOTEENN	15	0.0455	0.9893 (98.93)	0.9808

Table-14: Summerized Results

As we have seen from the above table, various types of techniques are applied to detect illicit transactions and controlling overfitting and underfitting problem that has been occurred due to imbalanced dataset. The techniques have been classified and applied in to two parts, the first part contain *Random under sampling*, *Nearmiss undersampling*, *Tomek Link Under sampling*, *Edited Nearest Neighbor Rule*, and *Neighborhood Cleaning Rule*. Therefore, Nearmiss under sampling technique perform a great result than other techniques in terms of performance accuracy, controlling overfitting and underfitting problem.

The second part, contain *Random Over Sampling*, *SMOTE Over Sampling*, *ADASYN over Sampling* and *Hybrid over Sampling* are included and the best result performed by Hybrid over sampling because theis techniques used the combination of the two class, to mean this it use under sampled and over sampled data.

Finally in this study, the Nearmis under sampling technique has better to control overfitting and underfitting but it has a low performance when we compare to the hybrid one. On the other way a hybrid approach has a good performance accuracy in terms of increasing performance accuracy and controlling over and underfitting problem.

Chapter 5 Conclusion and Future Works

5.1 Conclusion

Money laundry is a serious world wide criminal activity and can be easily operated in three stages according to this it is very difficult to easily detect the laundred transaction specially when the dataset is imbalanced. The main contribution of this paper is, increasing the performance accuracy of the detection level by compairing various data resampling technicque such as undersampling, oversampling and hybrid techniques. In addition to this, various regularization technique has been done to enhance the learning capability of the model and which resampling and regularization technique can increase the performance accuracy of the detection level is clearly stated.

To facilitate this study, we used an Elliptical dataset that has been provided by the weber, which contains Bitcoin transactions data. This dataset is the world's largest labelled transaction data set publicly available in 2019 from any cryptocurrency. This dataset contains 203,769 transactions from this amount 4545 are illicit and 42019 are Licit. The dataset is highly imbalanced when we compare both class.

When we feed an input data of a highly unbalanced class distribution to the model, the model tends to be biased towards the majority samples. As a result, it tends to misrepresent an illicit transaction as a licit transaction. To tackle this problem, we implemented regularization techniques such as *dropout*, *L1 and L2 kernel regularizer* and in data- level approach which includes various resampling techniques namely, *random under sampling*, *Tomek links removal*, *random oversampling*, *Synthetic Minority Over-sampling Technique (SMOTE)* and a *hybrid resampling*. Then, we analyzed all techniques using CNN model. The comparison results revealed that the Nearmiss under sampling technique is better interms of controlling overfitting and underfitting but has less performance accuracy (97.01) on the other hand *Hybrid approach (SMOTEENN)* has a greate performance accuracy (98.93) because this technique used the combination of oversampling and undersampling techniques and can easily monitoring overfitting problem.

5.2 Recommendation

To increase the capability of illicit transaction detection, class balancing is mandatory. Therefore, anyone can start from here to completely eliminate overfitting problem using various algorithm and technique.

For implementation purpose we used a synthetic dataset that is developed based on the real scenario due to the lack of real data from financial intelligence unit and any one can start from here by using our country real transaction data.

Sometimes there is result unstability in python code output. Thus, using other tools may increase the detection accuracy result and eliminate overfitting.

REFERENCE

- Albawi, S., & Mohammed, T. A. (2017). *Understanding of a Convolutional Neural Network*.
- Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Hasan, M., Van Essen, B. C., Awwal, A. A. S., & Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics (Switzerland)*, 8(3). <https://doi.org/10.3390/electronics8030292>
- Amin, A., Anwar, S., Adnan, A., Nawaz, M., Howard, N., Qadir, J., Hawalah, A., & Hussain, A. (2016). Comparing Oversampling Techniques to Handle the Class Imbalance Problem: A Customer Churn Prediction Case Study. *IEEE Access*, 4(M1), 7940–7957. <https://doi.org/10.1109/ACCESS.2016.2619719>
- Basel Committee on Banking Supervision. (2011). Principles for the Sound Management of Operational Risk. *Bank for International Settlements, June 2011*, 1–19. <https://www.bis.org/publ/bcbs195.pdf>
- Bennin, K. E., Keung, J. W., & Monden, A. (2019). On the relative value of data resampling approaches for software defect prediction. In *Empirical Software Engineering* (Vol. 24, Issue 2). Empirical Software Engineering. <https://doi.org/10.1007/s10664-018-9633-6>
- Burnaev, E., Erofeev, P., & Papanov, A. (2017). Influence of resampling on accuracy of imbalanced classification. *ArXiv*. <https://doi.org/10.1117/12.2228523>
- Cervantes, J., Li, X., & Yu, W. (2013). Using genetic algorithm to improve classification accuracy on imbalanced data. *Proceedings - 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013*, 2659–2664. <https://doi.org/10.1109/SMC.2013.7>
- Deng, L., & Yu, D. (2013). Deep learning: Methods and applications. *Foundations and Trends in Signal Processing*, 7(3–4), 197–387. <https://doi.org/10.1561/20000000039>
- Ertekin, S., Huang, J., Bottou, L., & Lee Giles, C. (2007). Learning on the border: active learning in imbalanced data classification. *International Conference on Information and Knowledge Management, Proceedings*, 127–136. <https://doi.org/10.1145/1321440.1321461>
- Federal Negarit Gazeta, 2009. (2009). *Federal Negarit. Gazeta No.*(Money Laundering and the financing of terrorism), page 5090.
- Fu, K., Cheng, D., Tu, Y., & B, L. Z. (2016). *Neural Information Processing - 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16-21, 2016, Proceedings, Part III*. 483–490. <https://doi.org/10.1007/978-3-319-46675-0>
- Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, 2(4), 42–47. http://www.ijetae.com/files/Volume2Issue4/IJETAE_0412_07.pdf
- Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation. *Departmental Technical Reports* (CS), 1–6. https://scholarworks.utep.edu/cgi/viewcontent.cgi?article=2202&context=cs_techrep
- Gyamfi, N. K., & Abdulai, J. D. (2019). Bank Fraud Detection Using Support Vector

- Machine. 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON 2018, 37–41. <https://doi.org/10.1109/IEMCON.2018.8614994>
- Hemdan, E. E. D., Shouman, M. A., & Karar, M. E. (2020). COVIDX-Net: A Framework of Deep Learning Classifiers to Diagnose COVID-19 in X-Ray Images. *ArXiv*.
- Hossin, M., & Sulaiman, M. . (2015). A REVIEW ON EVALUATION METRICS FOR DATA CLASSIFICATION EVALUATIONS. 5(2), 1–11.
- Hussein, A. S., Li, T., Yohannese, C. W., & Bashir, K. (2019). A-SMOTE: A new preprocessing approach for highly imbalanced datasets by improving SMOTE. *International Journal of Computational Intelligence Systems*, 12(2), 1412–1422. <https://doi.org/10.2991/ijcis.d.191114.002>
- Ikasari, I. H., Ayumi, V., Fanany, M. I., Mulyono, S., & Government, D. C. (2016). Multiple Regularizations Deep Learning for Paddy Growth Stages Classification Multiple Regularizations Deep Learning for Paddy Growth. October.
- Indolia, S., Goswami, A. K., Mishra, S. P., & Asopa, P. (2018). Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach. *Procedia Computer Science*, 132, 679–688. <https://doi.org/10.1016/j.procs.2018.05.069>
- Jamshidi, M. B., Gorjankhanzad, M., Lalbakhsh, A., & Roshani, S. (2019). A novel multiobjective approach for detecting money laundering with a neuro-fuzzy technique. *Proceedings of the 2019 IEEE 16th International Conference on Networking, Sensing and Control, ICNSC 2019*, 454–458. <https://doi.org/10.1109/ICNSC.2019.8743234>
- Jin, Y., & Qu, Z. (2019). Research on Anti-Money Laundering Hierarchical Model. *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS, 2018-Novem*, 406–411. <https://doi.org/10.1109/ICSESS.2018.8663895>
- Junsomboon, N., & Phienthrakul, T. (2017). Combining over-sampling and under-sampling techniques for imbalance dataset. *ACM International Conference Proceeding Series, Part F1283(1)*, 243–247. <https://doi.org/10.1145/3055635.3056643>
- Kaur, P., & Gosain, A. (2018). Comparing the behavior of oversampling and undersampling approach of class imbalance learning by combining class imbalance problem with noise. *Advances in Intelligent Systems and Computing*, 653(January), 23–30. https://doi.org/10.1007/978-981-10-6602-3_3
- Khan, A., Sohail, A., Zahoor, U., & Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8), 5455–5516. <https://doi.org/10.1007/s10462-020-09825-6>
- Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–15.
- Kuncheva, L. I., Arnaiz-González, Á., Díez-Pastor, J. F., & Gunn, I. A. D. (2019). Instance selection improves geometric mean accuracy: a study on imbalanced data classification. *Progress in Artificial Intelligence*, 8(2), 215–228. <https://doi.org/10.1007/s13748-019-00172-4>
- Le Khac, N. A., Markos, S., & Kechadi, M. T. (2010). A data mining-based solution for detecting suspicious money laundering cases in an investment bank. *2nd International*

- Conference on Advances in Databases, Knowledge, and Data Applications, DBKDA 2010*, 235–240. <https://doi.org/10.1109/DBKDA.2010.27>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning*. <https://doi.org/10.1038/nature14539>
- Li, Z., Kamnitsas, K., & Glocker, B. (2019). Overfitting of Neural Nets Under Class Imbalance: Analysis and Improvements for Segmentation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11766 LNCS, 402–410. https://doi.org/10.1007/978-3-030-32248-9_45
- Lindsay, G. W. (2020). Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future. *ArXiv*. https://doi.org/10.1162/jocn_a_01544
- Lorenz, J. (2020). *Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity*.
- Luo, X. (2014). *mnad in e e V by e th rsio is n fil O e is nly mnad in e e V by e th rsio is n fil O e*. 8(2), 157–166.
- Lv, F., Wang, W., Wei, Y., Sun, Y., Huang, J., & Wang, B. (2019). Detecting Fraudulent Bank Account Based on Convolutional Neural Network with Heterogeneous Data. *Mathematical Problems in Engineering*, 2019(1). <https://doi.org/10.1155/2019/3759607>
- Maghrebi, H., Portigliatti, T., & Prouff, E. (2016). Breaking cryptographic implementations using deep learning techniques. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10076 LNCS, 3–26. https://doi.org/10.1007/978-3-319-49445-6_1
- Manjunath, K. V. (2015). *Data Mining Techniques for Anti Money Laundering*. 2(8), 819–823.
- Murugan, P. (2017). Feed Forward and Backward Run in Deep Convolution Neural Network. *ArXiv*, 1–20.
- Niu, X., Wang, L., & Yang, X. (2019). A comparison study of credit card fraud detection: Supervised versus unsupervised. *ArXiv*.
- Pang, Y., Wang, Z., Jhong, J.-H., & Lee, T.-Y. (2021). Identifying anti-coronavirus peptides by incorporating different negative datasets and imbalanced learning strategies. *Briefings in Bioinformatics*, 00(April 2020), 1–11. <https://doi.org/10.1093/bib/bbaa423>
- Paper, C., & Laurikkala, J. (2014). *Improving Identification of Difficult Small Classes by Improving Identification of Difficult Small Classes by Balancing Class Distribution*. August, 3–7. <https://doi.org/10.1007/3-540-48229-6>
- Paula, E. L., Ladeira, M., Carvalho, R. N., & Marzag, T. (2016). *Deep Learning Anomaly Detection as Support Fraud Investigation in Brazilian Exports and Anti-Money Laundering*. 954–960. <https://doi.org/10.1109/ICMLA.2016.73>
- Phaisangittisagul, E. (2016). *An Analysis of the Regularization between L 2 and Dropout in Single Hidden Layer Neural Network*. <https://doi.org/10.1109/ISMS.2016.14>
- Sanguanmak, Y., & Hanskunatai, A. (2016). DBSM: The combination of DBSCAN and SMOTE for imbalanced data classification. *2016 13th International Joint Conference on Computer Science and Software Engineering, JCSSE 2016*, 1–5. <https://doi.org/10.1109/JCSSE.2016.7748928>
- Shakya, R. (2018). *Application of Machine Learning Techniques in Credit Card Fraud*

- Detection. December, 1–67.*
- Shelke, M. S., Deshmukh, P. R., & Shandilya, P. V. K. (2017). A Review on Imbalanced Data Handling Using Undersampling and Oversampling Technique. *International Journal of Recent Trends in Engineering and Research*, 3(4), 444–449. <https://doi.org/10.23883/ijrter.2017.3168.0uwxm>
- Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 45(4), 427–437. <https://doi.org/10.1016/j.ipm.2009.03.002>
- Soltani, R., Nguyen, U. T., Yang, Y., Faghani, M., Yagoub, A., & An, A. (2016). A New Algorithm for Money Laundering Detection Based on Structural Similarity. October. <https://doi.org/10.1109/UEMCON.2016.7777919>
- Somasundaram, A. (2016). *Data Imbalance : Effects and Solutions for Classification of Large and Highly Imbalanced Data*. November 2017.
- Suresh, C. (2016). A Hybrid Approach for Detecting Suspicious Accounts in Money Laundering Using Data Mining Techniques. May, 37–43. <https://doi.org/10.5815/ijitcs.2016.05.04>
- Tai, C. H., & Kan, T. J. (2019). Identifying Money Laundering Accounts. *Proceedings of 2019 International Conference on System Science and Engineering, ICSSE 2019*, 379–382. <https://doi.org/10.1109/ICSSE.2019.8823264>
- van Rosendael, A. R., Maliakal, G., Kolli, K. K., Beecy, A., Al'Aref, S. J., Dwivedi, A., Singh, G., Panday, M., Kumar, A., Ma, X., Achenbach, S., Al-Mallah, M. H., Andreini, D., Bax, J. J., Berman, D. S., Budoff, M. J., Cademartiri, F., Callister, T. Q., Chang, H. J., ... Min, J. K. (2018). Maximization of the usage of coronary CTA derived plaque information using a machine learning based algorithm to improve risk stratification; insights from the CONFIRM registry. *Journal of Cardiovascular Computed Tomography*, 12(3), 204–209. <https://doi.org/10.1016/j.jcct.2018.04.011>
- Wang, S., Liu, W., Wu, J., Cao, L., Meng, Q., & Kennedy, P. J. (2016). Training Deep Neural Networks on Imbalanced Data Sets. 4368–4374.
- Weber, M., Domeniconi, G., Chen, J., Weidele, D. K. I., Bellei, C., Robinson, T., & Leiserson, C. E. (2018). *Anti-Money Laundering in Bitcoin : Experimenting with Graph Convolutional Networks for Financial Forensics*. 10.
- Xu, Z., Shen, D., Nie, T., & Kou, Y. (2020). A hybrid sampling algorithm combining M-SMOTE and ENN based on Random forest for medical imbalanced data. *Journal of Biomedical Informatics*, 107(May), 103465. <https://doi.org/10.1016/j.jbi.2020.103465>
- Ying, X. (2019). An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series*, 1168(2). <https://doi.org/10.1088/1742-6596/1168/2/022022>
- Zeng-an, G. A. O., & Li-fang, W. (2006). Transfer Price-based Money Laundering in International Trade. 1128–1132.
- Zezhong, J., Yang, H., & Chow, C. W. (2004). The determinants and characteristics of voluntary Internet-based disclosures by listed Chinese companies. 23, 191–225. <https://doi.org/10.1016/j.jaccpubpol.2004.04.002>
- Zhang, C. L., Luo, J. H., Wei, X. S., & Wu, J. (2018). In defense of fully connected layers in visual representation transfer. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10736 LNCS, 807–817. <https://doi.org/10.1007/978-3-319-77383->

- Zhang, Q., Zhang, M., Chen, T., Sun, Z., Ma, Y., & Yu, B. (2019). Recent advances in convolutional neural network acceleration. *Neurocomputing*, 323, 37–51. <https://doi.org/10.1016/j.neucom.2018.09.038>
- Zhang, Y., & Trubey, P. (2018). Machine Learning and Sampling Scheme : An Empirical Study of Money Laundering Detection. *Computational Economics*. <https://doi.org/10.1007/s10614-018-9864-z>

ANNEX

Suspicious Money Laundering Detection using Deep Learning

Technique

Lists of Libraries used to develop this study

```
In [1]: from future import print_function, division
import numpy as np
import pandas as pd
import os

for dirname, __, filenames in os.walk(''):
    for filename in filenames:
        print(os.path.join(dirname, filename))

import tensorflow as tf
import networkx as nx
import matplotlib.pyplot as plt
%matplotlib inline
import plotly.offline as py
import plotly.graph_objs as go
py.init_notebook_mode(connected=True)
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_recall_fscore_support
from sklearn.metrics import f1_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.cluster import KMeans
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import StandardScaler
import plotly.figure_factory as plt
from keras.models import Sequential
from tensorflow.keras.layers import Flatten, Dense, Dropout, BatchNormalization
from tensorflow.keras.layers import Conv1D, Conv2D
from tensorflow.keras.layers import MaxPooling1D
from tensorflow.keras.optimizers import Adam
from keras.layers import Dense, Dropout
from keras import regularizers
print(tf.__version__)
```

Reading csv files

[illegible]

Edgelist sample data

```
In [3]: len(edges), len(features), len(classes)
```

```
Out[3]: (234355, 203769, 203769)
```

```
In [4]: display(edges.head(5), features.head(5), classes.head(5))
```

	txId1	txId2
0	230425980	5530458
1	232022460	232438397
2	230460314	230459870
3	230333930	230595899
4	232013274	232029206

Features sample data

	0	1	2	3	4	5	6	7	8	...	157	158	159	160	
0	230425980	1	-0.171439	-0.104896	-1.201369	-0.121970	-0.042075	-0.112002	-0.061534	-0.182097	...	-0.582152	-0.603999	1.451320	1.461369
1	232022460	1	-0.171439	-0.154058	-1.201369	-0.121970	-0.042075	-0.112002	-0.061534	-0.182097	...	0.347382	0.073103	-0.379037	-0.378056
2	230460314	1	0.112137	0.184888	1.201369	0.121970	0.042075	0.112002	0.061534	0.182097	...	0.870882	0.429728	0.379037	0.378056
3	230333930	1	0.163057	1.053790	-0.646376	17.408794	-0.063725	0.752742	12.414555	-0.153515	...	-0.577066	-0.013010	0.241178	0.241400
4	230460314	1	1.011623	0.081127	1.201369	1.153888	0.322276	1.312856	0.061534	0.182097	...	0.511871	0.463422	0.517257	0.515382

5 rows x 157 columns

Classes sample data

	txId	class
0	230425980	unknown
1	5530458	unknown
2	232022460	unknown
3	232438397	2
4	230460314	unknown

Class concatenation

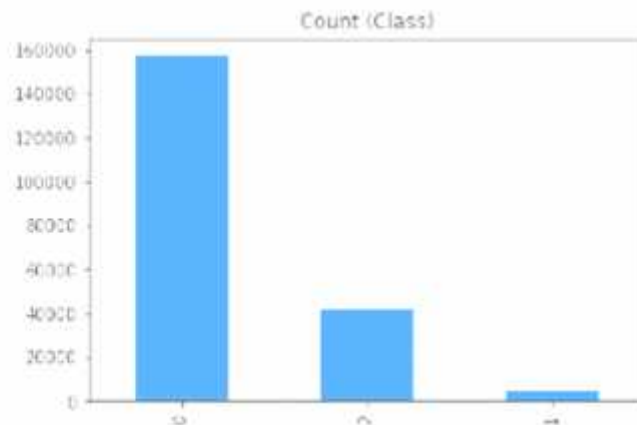
```
In [5]: tx_features = ["tx_feat_{}".format(i) for i in range(2, 168)]
agg_features = ["agg_feat_{}".format(i) for i in range(1, 13)]
features.columns = ["txId", "line_step"] + tx_features + agg_features
features = pd.merge(features, classes, left_on="txId", right_on="txId", how="left")
features["class"] = features["class"].apply(lambda x: "0" if x == "unknown" else x)
```

Class and amount of transactions

```
In [11]: data = features[(features['class']=='1') | (features['class']=='2')]

In [12]: target_count = features['class'].value_counts()
print('Class 0 Unknown:', target_count[0])
print('Class 1 Illicit:', target_count[1])
print('Class 2 Licit:', target_count[2])
target_count.plot(kind='bar', title='Count (Class)')

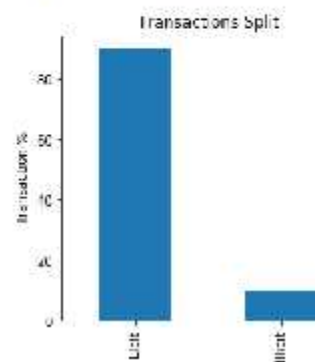
Class 0 Unknown: 157208
Class 1 Illicit: 4248
Class 2 Licit: 4209
```



Transaction split

```
In [14]: plt.figure(figsize=(4, 4))
trans_count = pd.Series([Licit trans pers*100, Illicit trans pers*100], index=['Licit', 'Illicit'])
trans_count.plot(kind='bar', title='Transactions Split')
plt.ylabel('Transaction %')

Out[14]: Text(0, 0.5, 'Transaction %')
```



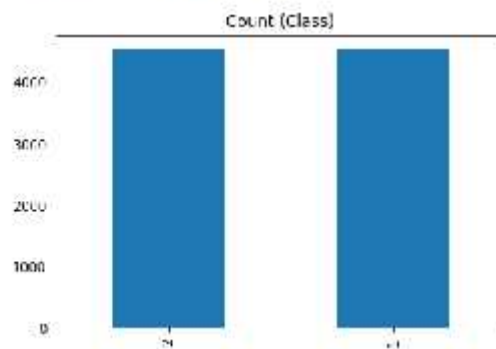
Random under Sampling

```
In [16]: df_class_2_under = df_class_2.sample(count_class_1)
df_test_under = pd.concat([df_class_2_under, df_class_1], axis=0)

print('Random under sampling:')
print(df_test_under.Class.value_counts())

df_test_under.Class.value_counts().plot(kind='bar', title='Count (Class)');

Random under sampling:
2    4545
1    4545
Name: Class, dtype: int64
```



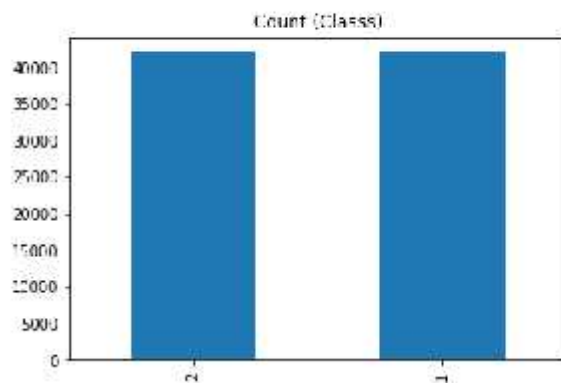
Random Over sampling

```
In [17]: df_class_1_over = df_class_1.sample(count_class_2, replace=True)
df_test_over = pd.concat([df_class_2, df_class_1_over], axis=0)

print('Random over sampling:')
print(df_test_over.Class.value_counts())

df_test_over.Class.value_counts().plot(kind='bar', title='Count (Classes)');

Random over sampling:
2    42019
1    42019
Name: Class, dtype: int64
```



Proposed Model

```
epochs = 100
model = Sequential()
model.add(Conv2D(filters=32, kernel_size=2, activation="relu", input_shape=X_train[0].shape))
model.add(MaxPooling2D(pool_size=2))
model.add(Conv2D(filters=64, kernel_size=2, kernel_regularizer=regularizers.L2(0.01), activation="relu"))
model.add(MaxPooling2D(pool_size=2))
model.add(Flatten())
model.add(Dense(128, activation="relu"))
model.add(Dense(1, activation="sigmoid"))
model.summary()
```

Random Under Sampling

```
In [283]: from imblearn.under_sampling import RandomUnderSampler
under_sampler = RandomUnderSampler()
X_train_under, y_train_under = under_sampler.fit_sample(X_train, y_train)
```

Nearmiss under Sampling

```
In [299]: from imblearn.under_sampling import NearMiss
near = NearMiss(sampling_strategy="not majority")
X_train_near, y_train_near = near.fit_sample(X_train, y_train)
```

Tomek link removal undersampling

```
In [302]: from imblearn.under_sampling import TomekLinks
tomek_links = TomekLinks(sampling_strategy='not majority')
X_train_tl, y_train_tl = tokek_links.fit_sample(X_train, y_train)
```

Undersampling with Edited Nearest Neighbor Rule

```
In [345]: from imblearn.under_sampling import EditedNearestNeighbours
enn = EditedNearestNeighbours(sampling_strategy="majority")
X_train_enn, y_train_enn = enn.fit_sample(X_train, y_train)
```

Undersampling with Neighborhood Cleaning Rule

```
In [25]: from imblearn.under_sampling import NeighbourhoodCleaningRule
ncr = NeighbourhoodCleaningRule(sampling_strategy="majority")
X_train_ncr, y_train_ncr = ncr.fit_sample(X_train, y_train)
```

Random Oversampling

```
In [31]: from imblearn.over_sampling import RandomOverSampler
over_sampler = RandomOverSampler(random_state=0)
X_train_over, y_train_over = over_sampler.fit_sample(X_train, y_train)

In [32]: X_train_over.shape, y_train_over.shape
Out[32]: ((60350, 165), (60350,))
```

SMOTE Oversampling

```
In [80]: from imblearn.over_sampling import SMOTE
smote = SMOTE(sampling_strategy='minority')
X_train_smote, y_train_smote = smote.fit_sample(X_train, y_train)
```

ADASYN Oversampling

```
In [85]: from imblearn.over_sampling import ADASYN
adasyn = ADASYN(sampling_strategy="minority")
X_train_adasyn, y_train_adasyn = adasyn.fit_sample(X_train, y_train)
```

Hybrid(SMOTE and Tomek Link Removal)

```
In [86]: from imblearn.combine import SMOTETomek
smotemek = SMOTETomek(sampling_strategy='auto')
X_train_smt, y_train_smt = smotemek.fit_sample(X_train, y_train)
```

Hybrid(SMOTEENN)

```
[84]: from imblearn.combine import SMOTEENN
SMOTEEN = SMOTEENN()
X_train_smtenn, y_train_smtenn = SMOTEEN.fit_sample(X_train, y_train)
```