Fraud Detection in Internet Banking Using Machine Learning

An Industry Oriented Mini Project Report Submitted to

Jawaharlal Nehru Technological University Hyderabad

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

By

O. RAJESH	**22E11A6743**
V.V.S. SIVA SHARMA	**22E11A6758**
Y. SAI KRISHNA	**22E11A6762**
P. NITISH	**22E11A6744**
K . UDAY RAM	**22E11A6719**
M. SHIVA PRASAD	**22E11A6731**

Under the guidance of

Mrs Nazneen Fathima, M.E
Assistant Professor
Department of Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BHARAT INSTITUTE OF ENGINEERING AND TECHNOLOGY
Accredited by NAAC, Accredited by NBA (UG Programmes: CSE, ECE, EEE & Mechanical)
Approved by AICTE, Affiliated to JNTUH Hyderabad
Ibrahimpatnam -501 510, Hyderabad, Telangana

JUNE 2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BHARAT INSTITUTE OF ENGINEERING AND TECHNOLOGY

Accredited by NAAC, Accredited by NBA (UG Programmes: CSE, ECE, EEE & Mechanical)
Approved by AICTE, Affiliated to JNTUH Hyderabad
Ibrahimpatnam -501 510, Hyderabad, Telangana

Certificate

*This is to certify that the Industry Oriented Mini Project work entitled "***Fraud Detection in Internet Banking Using Machine Learning"** *is the bonafide work done*

By

O. RAJESH	**22E11A6743**
V.V.S. SIVA SHARMA	**22E11A6758**
Y. SAI KRISHNA	**22E11A6762**
P. NITISH	**22E11A6744**
K . UDAY RAM	**22E11A6719**
M. SHIVA PRASAD	**22E11A6731**

in the Department of Computer Science and Engineering, ***BHARAT INSTITUTE OF ENGINEERING AND TECHNOLOGY,*** *Ibrahimpatnam is submitted to Jawaharlal* ***Nehru Technological University, Hyderabad*** *in partial fulfillment of the requirements for the award of* ***B.Tech*** *degree in* ***Computer Science and Engineering*** *during* ***2024-2025.***

Supervisor:
Mrs Nazneen Fathima
Assistant Professor
Dept of Computer Science and Engineering
Bharat Institute of Engineering and Technology
Ibrahimpatnam–501 510, Hyderabad

Department I/C
Dr.Velmurgan
Professor
Dept of Computer Science and Engineering
Bharat Institute of Engineering and Technology
Ibrahimpatnam– 501 510, Hyderabad

Viva-Voce held on……

Internal Examiner **External Examin**

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

We avail this opportunity to express our deep sense of gratitude and hearty thanks to **Sri CH. Venugopal Reddy**, Chairman & Secretary of BIET, for providing congenial atmosphere and encouragement.

We would like to thank **Prof. G. Kumaraswamy Rao**, Former Director & O.S. of DLRL Ministry of Defence, Sr. Director R&D, BIET, and **Dr. V Srinivasa Rao**, Dean CSE, for having provided all the facilities and support.

We would like to thank our Department Incharge / HOD **Dr.Velmurgan**, for encouragement at various levels of our Project.

We are thankful to our Project Coordinator **Mr. Rama Prakasha Reddy**, Assistant Professor, Computer Science and Engineering for her support and cooperation throughout the process of this project.

We are thankful to our guide **Mrs Nazeen Fathima**, Assistant Professor, Computer Science and Engineering for his sustained inspiring Guidance and cooperation throughout the process of this project. His wise counsel and suggestions were invaluable.

We express our deep sense of gratitude and thanks to all the Teaching and Non-Teaching Staff of our college who stood with us during the project and helped us to make it a successful venture.

We place highest regards to our Parent, our Friends and Well-wishers who helped a lot in making the report of this project

O. RAJESH	**22E11A6743**
V.V.S. SIVA SHARMA	**22E11A6758**
Y. SAI KRISHNA	**22E11A6762**
P. NITISH	**22E11A6744**
K . UDAY RAM	**22E11A6719**
M. SHIVA PRASAD	**22E11A6731**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BHARAT INSTITUTE OF ENGINEERING AND TECHNOLOGY

Accredited by NAAC, Accredited by NBA (UG Programmes: CSE, ECE, EEE & Mechanical)
Approved by AICTE, Affiliated to JNTUH Hyderabad
Ibrahimpatnam -501 510, Hyderabad, Telangana

Declaration

We hereby declare that this Industry Oriented Mini Project is titled **Fraud Detection in Internetbanking Using Machine Learning** is a genuine Industry Oriented Mini Project work carried out by us, in **B.Tech (Computer Science and Engineering)** degree course of **Jawaharlal Nehru Technology University Hyderabad, Hyderabad** and has not been submitted to any other course or university for the award of my degree by me.

Signatures of the Project team members

1.

2.

3.

4.

5.

6.

ABSTRACT

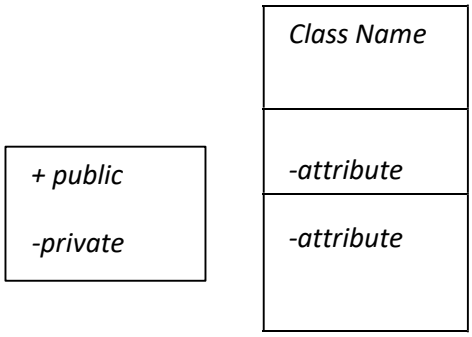
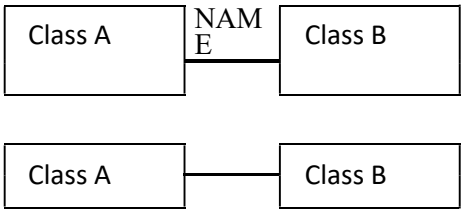
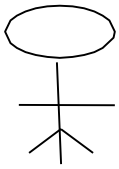
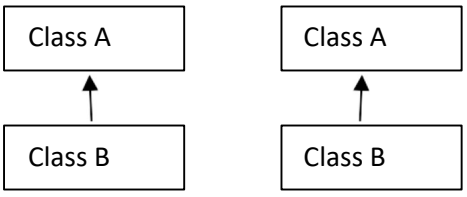
Banking fraud transactions refer to unauthorized or deceptive activities involving bank accounts or financial transactions. Various machine learning algorithms can be employed to detect such fraudulent activities. This study examines several algorithms suitable for classifying transactions as either fraudulent or legitimate. The research utilizes the Banking Fraud Transactions dataset, which is often characterized by high imbalance. To address this issue, we are implementing multiple machine learning algorithms like Random Forest, K-Nearest Neighbour and Decision Tree. Additionally, feature selection techniques are employed, and the dataset is divided into training and test sets. The algorithms evaluated in the study include Random Forest, and KNN. The findings indicate that each algorithm demonstrates high accuracy in detecting banking fraud transactions. The proposed model holds promise for detecting other irregularities within financial transactions.

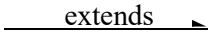
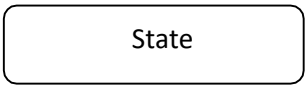
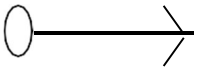
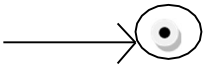
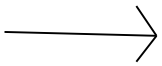
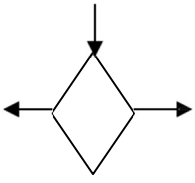
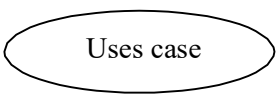
Key words : Fraud Transactions,Machine Learning Algorithms,Fraud Detection,Random Forest, K-Nearest Neighbour (KNN), Imbalanced Dataset,Feature Selection,Classification,Financial Fraud Detection,Model Evaluation

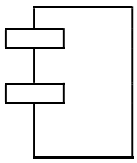
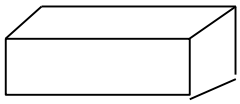
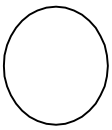
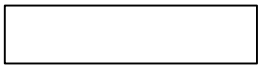
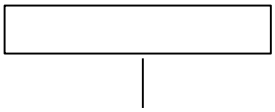
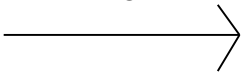
TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYSMBOLS

S.NO	NOTATION NAME	NOTATION	DESCRIPTION
1.	Class		Represents a collection of similar entities grouped together.
2.	Association		Associations represents static relationships between classes. Roles represents the way the two classes see each other.
3.	Actor		It aggregates several classes into a single classes.
4.	Aggregation		Interaction between the system and external environment

5.	Relation (uses)	uses	Used for additional process communication.
6.	Relation (extends)		Extends relationship is used when one use case is similar to another use case but does a bit more.
7.	Communication		Communication between various use cases.
8.	State		State of the processes.
9.	Initial State		Initial state of the object
10.	Final state		Final state of the object
11.	Control flow		Represents various control flow between the states.
12.	Decision box		Represents decision making process from a constraint
13.	Use case		Interact ion between the system and external environment.

14.	Component		Represents physical modules which are a collection of components.
15.	Node		Represents physical modules which are a collection of components.
16.	Data Process/State		A circle in DFD represents a state or process which has been triggered due to some event or action.
17.	External entity		Represents external entities such as keyboard, sensors, etc.
18.	Transition		Represents communication that occurs between processes.
19.	Object Lifeline		Represents the vertical dimensions that the object communications.
20.	Message	Message 	Represents the message exchanged.

CHAPTER-1

INTRODUCTION

1.1 GENERAL

In recent years, the financial sector has witnessed a surge in fraudulent activities, ranging from credit card fraud to identity theft and money laundering. These nefarious activities not only result in substantial financial losses for both financial institutions and customers but also undermine trust in the banking system. To combat this escalating threat, there is a growing imperative to deploy sophisticated technological solutions capable of detecting and preventing fraudulent transactions in real-time.

Machine learning, with its ability to analyze vast amounts of data and identify complex patterns, has emerged as a powerful tool in the fight against banking fraud. By leveraging historical transaction data, machine learning algorithms can learn to distinguish between legitimate and fraudulent transactions, thereby flagging suspicious activities for further investigation. However, the efficacy of machine learning models in detecting banking fraud hinges on several factors, including the quality and representativeness of the training data, the choice of algorithms, and the evaluation metrics used to assess model performance. One of the primary challenges faced by researchers and practitioners in this domain is the imbalance inherent in banking fraud datasets, where instances of fraudulent transactions are significantly outnumbered by legitimate ones. This class imbalance can lead to biased models that prioritize accuracy at the expense of correctly identifying fraudulent transactions, which are often the minority class.

To address this challenge, this project adopts a multifaceted approach. Firstly, a diverse range of machine learning algorithms, including Random Forest, K-Nearest Neighbors (KNN), and Logistic Regression, are employed to develop a robust fraud detection model. These algorithms offer distinct advantages and trade-offs, making them well-suited for handling different aspects of the fraud detection task.

Furthermore, feature selection techniques are applied to identify the most relevant attributes or features that contribute to distinguishing fraudulent from legitimate transactions. By focusing on

the most informative features, the model can improve its predictive accuracy and generalization performance.

Additionally, the dataset is carefully partitioned into training and test sets to facilitate unbiased model evaluation. Performance metrics such as precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC) are used to comprehensively assess the performance of the developed models.

By systematically exploring and evaluating these methodologies, this project aims to not only develop an effective fraud detection system for banking transactions but also contribute to the broader understanding of how machine learning can be leveraged to address complex challenges in the financial sector. Ultimately, the insights gained from this study hold the potential to inform the development of more robust and adaptive fraud detection solutions, thereby safeguarding the integrity of financial systems and protecting stakeholders from the pervasive threat of banking fraud.

1.2 OBJECTIVE

The objective of this project is to develop and evaluate a machine learning-based system for detecting banking fraud transactions. Specifically, the project aims to address the challenge of class imbalance in banking fraud datasets by employing various machine learning algorithms, including Random Forest, K-Nearest Neighbours (KNN), and Logistic Regression. Through the utilization of feature selection techniques and careful partitioning of the dataset into training and test sets, the project seeks to improve the accuracy and generalization performance of the fraud detection models. Furthermore, the project aims to assess the effectiveness of each algorithm in accurately classifying transactions as either fraudulent or legitimate, utilizing performance metrics such as precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC). By achieving these objectives, the project endeavors to contribute to the development of robust fraud detection systems capable of identifying and preventing fraudulent activities within the banking sector.

1.3 SCOPE OF THE PROJECT

The scope of this project encompasses the development and evaluation of a machine learning-based system for detecting banking fraud transactions. This includes the exploration and implementation of multiple machine learning algorithms, such as Random Forest, K-Nearest Neighbors (KNN), and Logistic Regression, to effectively classify transactions as either fraudulent or legitimate. Additionally, feature selection techniques will be applied to identify the most relevant attributes contributing to fraud detection. The project will address the challenge of class imbalance in banking fraud datasets by employing strategies to mitigate bias and improve model performance.

1.4 EXISTING SYSTEM:

- This paper presents a core decision tree algorithm to identify money laundering activities. The clustering algorithm is the combination of BIRCH and means. In this method, decision tree of data mining technology is applied to anti-money-laundering filed after research of money laundering features.
- We select an appropriate identifying strategy to discover typical money laundering patterns and money laundering rules. Consequently, with the core decision tree algorithm, we can identify abnormal transaction data more effectively.

1.4.1 EXISTING SYSTEM DISADVANTAGES:

- Unstable nature. One of the limitations of decision trees is that they are largely unstable compared to other decision predictors.
- Less effective in predicting the outcome of a continuous variable.

1.5 PROBLEM STATEMENT:

Banking fraud transactions encompass unauthorized or deceptive activities within bank accounts or financial transactions. Detecting such fraudulent activities is crucial for financial institutions to prevent losses and maintain trust with customers. This study focuses on employing machine learning algorithms to accurately classify transactions as either fraudulent or legitimate, using the Banking Fraud Transactions dataset. One of the key challenges is the high class imbalance in the dataset, which requires addressing through appropriate techniques.

This study aims to detect banking fraud transactions by employing machine learning algorithms on the Banking Fraud Transactions dataset, addressing the challenge of high class imbalance. The approach involves dividing the dataset into training and test sets with a representative distribution of fraudulent and legitimate transactions, utilizing feature selection techniques to identify key features for classification. Multiple algorithms like Random Forest, KNN, and Decision Tree are implemented and evaluated using metrics such as accuracy, precision, recall, F1-score, and ROC curves. The study also explores model enhancement through ensemble techniques and hyper parameter tuning, aiming to improve accuracy and robustness. Furthermore, the model's generalization capabilities are assessed to detect broader financial irregularities, showcasing its potential for enhancing financial security measures beyond banking fraud detection.

1.6 LITERATURE SURVEY

Title: Detecting money laundering transactions with machine learning

Author: M. Jullum, A. Løland, R. B. Huseby, G. A˚ nonsen, and J. Lorentzen

Year: 2020

Description:

Purpose The purpose of this paper is to develop, describe and validate a machine learning model for prioritising which financial transactions should be manually investigated for potential money laundering. The model is applied to a large data set from Norway’s largest bank, DNB. Design/methodology/approach A supervised machine learning model is trained by using three

types of historic data: “normal” legal transactions; those flagged as suspicious by the bank’s internal alert system; and potential money laundering cases reported to the authorities. The model is trained to predict the probability that a new transaction should be reported, using information such as background information about the sender/receiver, their earlier behaviour and their transaction history. Findings The paper demonstrates that the common approach of not using non-reported alerts (i.e. transactions that are investigated but not reported) in the training of the model can lead to sub-optimal results. The same applies to the use of normal (un-investigated) transactions. Our developed method outperforms the bank’s current approach in terms of a fair measure of performance. Originality/value This research study is one of very few published anti-money laundering (AML) models for suspicious transactions that have been applied to a realistically sized data set. The paper also presents a new performance measure specifically tailored to compare the proposed method to the bank’s existing AML system.

Title: An improved support-vector network model for anti-money laundering

Author : L. Keyan and Y. Tingting.

YEAR:2019

Description: The selection of parameters of SVM model will affect the identification effect of suspicious financial transactions, this paper proposes the cross validation method to find the optimal SVM classifier parameters to solve this problem. Cross validation method finds the optimal parameters based on the highest classification accuracy rate through grid search, it can effectively avoid the state of over-learning and less learning, and greatly improves the overall performance of the classifier.

Title: Research on anti-money laundering based on core decision tree algorithm

Author: R. Liu, X.-l. Qian, S. Mao, and S.-z. Zhu,,

Year: 2020

Description: This paper presents a core decision tree algorithm to identify money laundering activities. The clustering algorithm is the combination of BIRCH and K-means. In this method,

decision tree of data mining technology is applied to anti-money-laundering filed after research of money laundering features. We select an appropriate identifying strategy to discover typical money laundering patterns and money laundering rules. Consequently, with the core decision tree algorithm, we can identify abnormal transaction data more effectively.

Title: Application of cluster-based local outlier factor algorithm in anti-money laundering.

Author: Z. Gao.,

Year: 2019

Description: Financial institutions' capability in recognizing suspicious money laundering transactional behavioral patterns (SMLTBPs) is critical to antimony laundering. Combining distance-based unsupervised clustering and local outlier detection, this paper designs a new cluster based local outlier factor (CBLOF) algorithm to identify SMLTBPs and use authentic and synthetic data experimentally to test its applicability and effectiveness...

Title: Incremental Neural-Network Learning for Big Fraud Data

Author: F. Anowar and S. Sadaoui,

Year: 2020

Description: Fraud detection systems aim to process a massive amount of data at high speed. To address the issues of data scalability, we introduce a chunk-based incremental classification approach based on a neural network (MLP) and a memory model to tackle the stability-plasticity dilemma. The incremental approach adapts the fraud model sequentially with incoming data chunks and retains past chunks a little more. We employ a large-scale credit-card fraud dataset that we organize into initial and incremental chunks for training and testing. Using data sampling, we solve the data skew problem, a critical issue in fraud detection. After each incremental phase, we evaluate the performance of the adjusted MLP classifier using the testing chunk. The experimental results demonstrate the effectiveness and efficiency of our incremental method and its superiority to the non-incremental MLP.

1.7 PROPOSED SYSTEM

- The proposed system is designed to enhance the detection of fraudulent activities within banking transactions through the utilization of machine learning algorithms.
- It begins by collecting transaction data, encompassing essential details such as amounts, timestamps, merchant information, and customer particulars.
- Training the system involves the division of the pre-processed data into training and testing sets, where machine learning algorithms like Logistic Regression, Random Forest, and Decision Tree are trained to discern patterns indicative of fraudulent behaviour.

1.7.1 PROPOSED SYSTEM ADVANTAGES: -

- Improved Accuracy
- Real-time
- Adaptability to Evolving Threats.
- Enhanced Customer Trust

CHAPTER 2

PROJECT DESCRIPTION

2.1 GENERAL:

The project aims to develop a robust machine learning-based system for detecting banking fraud transactions using the Banking Fraud Transactions dataset. This dataset is known for its high class imbalance, presenting a challenge that is addressed through the implementation of multiple algorithms including Random Forest, K-Nearest Neighbors (KNN), and Logistic Regression. Feature selection techniques are employed to enhance model performance, and the dataset is divided into training and test sets for evaluation. The study evaluates the effectiveness of each algorithm in accurately classifying transactions as either fraudulent or legitimate. While focusing on accuracy, the project also considers additional performance metrics such as precision, recall, F1-score, and AUC-ROC to provide a comprehensive assessment. Through rigorous experimentation and analysis, the findings demonstrate the capability of the proposed model to effectively detect banking fraud transactions, holding potential for broader applications in identifying irregularities within financial transactions.

2.2 METHODOLOGIES

2.2.1 MODULES NAME:

1.Data Collection:

This is the first real step towards the real development of a learning model, collecting data. This is a critical step that will cascade in how good the model will be, the more and better data that we get, the better our model will perform.

There are several techniques to collect the data, like web scraping, manual interventions and etc.

- Money Laundering Detection

2.Dataset:

The dataset consists of individual data in that there are 1048576 rows & 11 columns in the dataset, which are described below.

1. step
2. type
3. amount
4. nameOrig
5. oldbalanceOrg
6. newbalanceOrig
7. nameDest
8. oldbalanceDest
9. newbalanceDest
10. isFraud
11. isFlaggedFraud

3.Data Preparation:

Wrangle data and prepare it for training. Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc.)

Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data

Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis

Split into training and evaluation sets

4.Model Selection:

We used Neural Network created our money laundering detection algorithm, We got a accuracy of 98.04% on test set so we implemented this algorithm.

5. Analyze and Prediction:

In the actual dataset, we chose only main 2 features:

1 Amount transactions - detailed descriptions of the Amount transactions data.
2 isFraud - indicates whether the transactions details is having fraud or not.

2.3 TECHNIQUE USED OR ALGORITHM USED

2.3.1 EXISTING TECHNIQUE: -

- Support Vector Machine (SVM) is a powerful supervised learning algorithm used for classification and regression tasks. It works by finding the optimal hyper plane that best separates different classes in the feature space. The hyper plane is determined by maximizing the margin, which is the distance between the hyper plane and the closest data points from each class, known as support vectors.
- One of the key strengths of SVM is its ability to handle high-dimensional data effectively, making it suitable for tasks with a large number of features. Additionally, SVM can handle non-linear decision boundaries through the use of kernel functions, which map the original feature space into a higher-dimensional space where classes are more easily separable. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid.
- SVM aims to find the hyper plane that not only separates the classes but also maximizes the margin, thereby improving generalization and reducing over fitting. This margin maximization leads to better performance on unseen data and makes SVM robust to outliers.

- **2.3.2 PROPOSED TECHNIQUE: -**

Random Forest and KNN

Random Forest is a versatile and powerful ensemble learning algorithm that operates by constructing a multitude of decision trees during training. Each tree in the forest independently learns to classify instances based on a subset of features, and the final classification is determined by aggregating the predictions of all trees. K-Nearest Neighbours (KNN) is a simple yet effective algorithm used for both classification and regression tasks. It operates on the principle of similarity, where the classification of a data point is determined by the class labels of its nearest neighbours in the feature space. KNN does not require explicit training as it stores all training data points and their corresponding labels, making it straightforward to implement and interpret.

Advantages:-

- Robustness to Noisy Data.
- Simplicity.
- No Assumptions
- Interpretability

CHAPTER 3

REQUIREMENTS ENGINEERING

3.1 GENERAL

The interpretation of the handwriting character by developing techniques and methods such as improvement of character classification techniques. The accurate and rapid classification for accurate information retrieval, sound classification, stock price forecasting.

3.2 HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shouls what the system do and not how it should be implemented.

- Processor - Pentium –IV
- Speed - 1.1 GHz
- Ram - 256 MB
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

3.3 SOFTWARE REQUIREMENTS

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team's progress throughout the development activity.

SOFTWARE REQUIREMENTS

- Operating System - Windows xp/7/10
- Coding Language - Python
- IDE - Spyder3

3.4 FUNCTIONAL REQUIREMENTS

Chest X-ray and CT are widely used in the screening and diagnosis of COVID-19 .It is important to employ a contactless and automated image acquisition workflow to avoid the severe risks of infection during COVID-19 pandemic. However, the conventional imaging workflow includes inevitable contact between technicians and patients. Many modern X-ray and CT systems are equipped with cameras for patient monitoring purposes. During the outbreak of COVID-19,those devices facilitate the implementation of a contactless scanning workflow.

3.5 NON-FUNCTIONAL REQUIREMENTS

The major non-functional Requirements of the system are as follows

Usability

The system is designed with completely automated process hence there is no or less user intervention.

Reliability

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using python is more reliable.

Performance

This system is developing in the high level languages and using the advanced front-end and back- end technologies it will give response to the end user on client system with in very less time.

Supportability

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is built into the system.

Implementation

The system is implemented in a web environment using the **Streamlit** framework. The application runs on a **Streamlit web server** and is deployed on a **Windows operating system**. The **user interface** is designed using **Streamlit's intuitive and interactive components**, providing a clean and responsive experience for end-users to interact with the fraud detection system.

CHAPTER 4

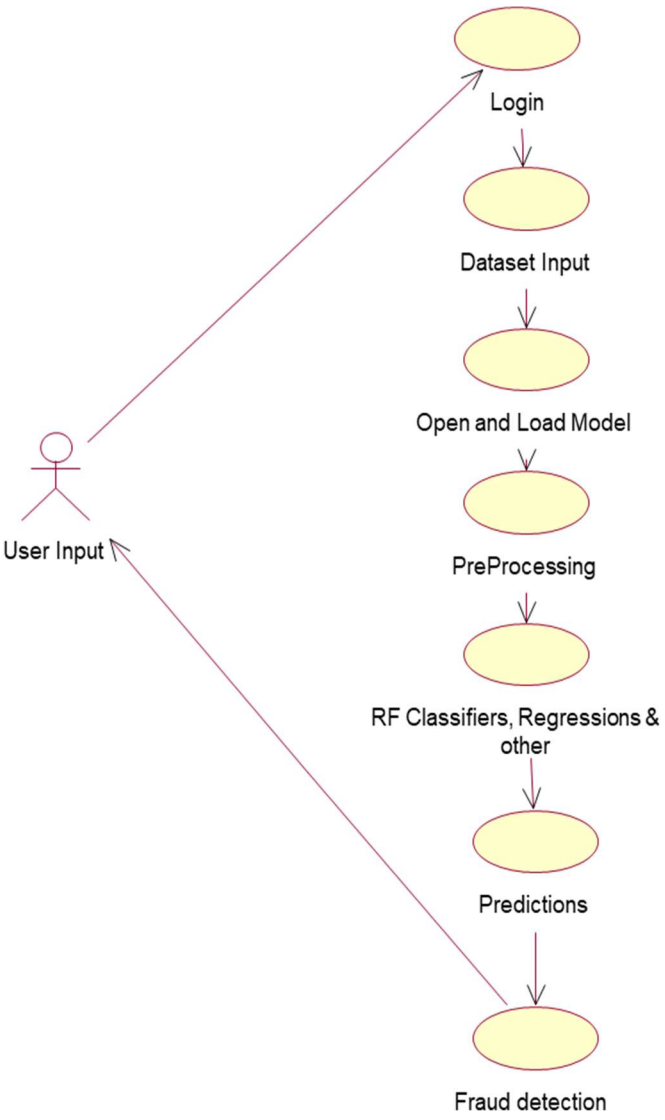
DESIGN ENGINEERING

4.1 GENERAL

Design Engineering deals with the various UML [Unified Modeling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product

4.2 UML DIAGRAMS

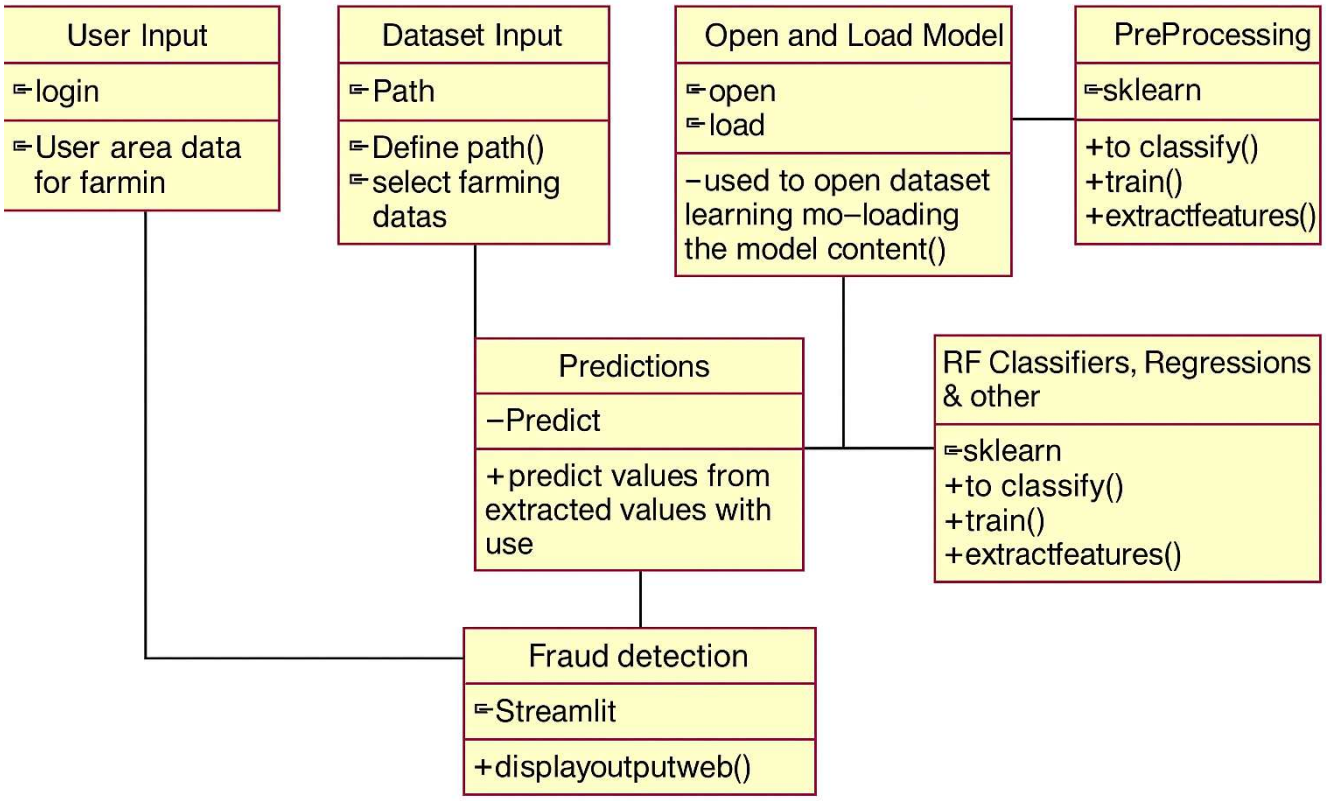
4.2.1 USE CASE DIAGRAM



EXPLANATION:

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

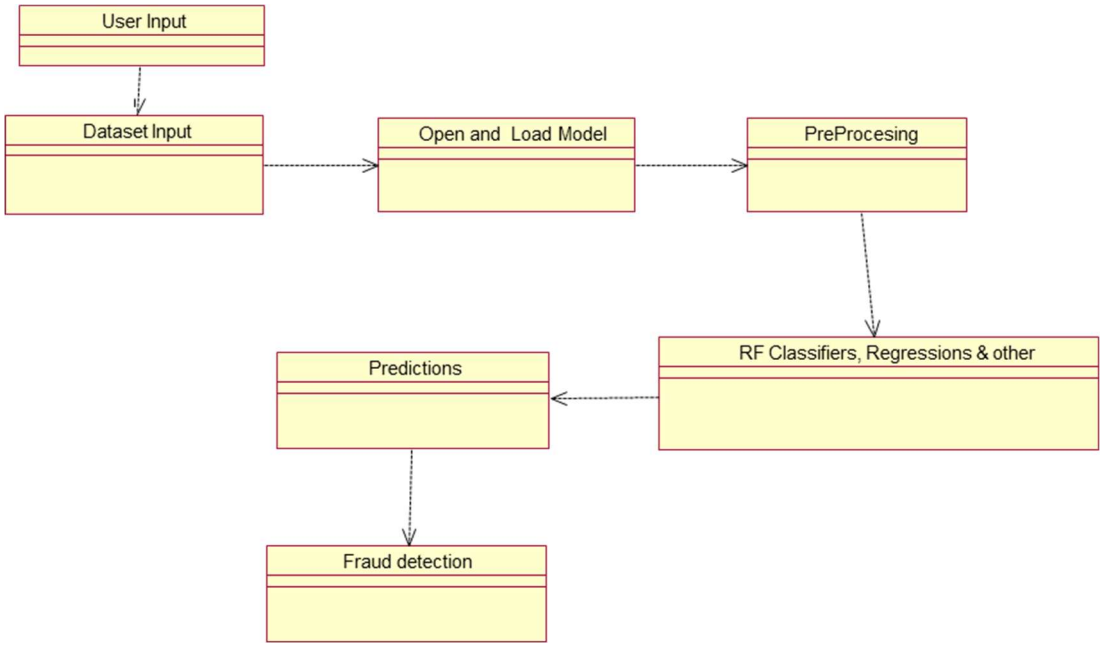
4.2.2 CLASS DIAGRAM



EXPLANATION

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.

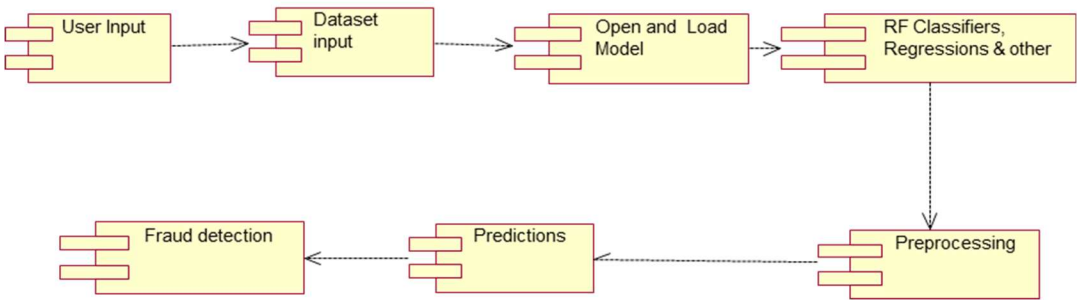
4.2.3 OBJECT DIAGRAM



EXPLANATION:

In the above digram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

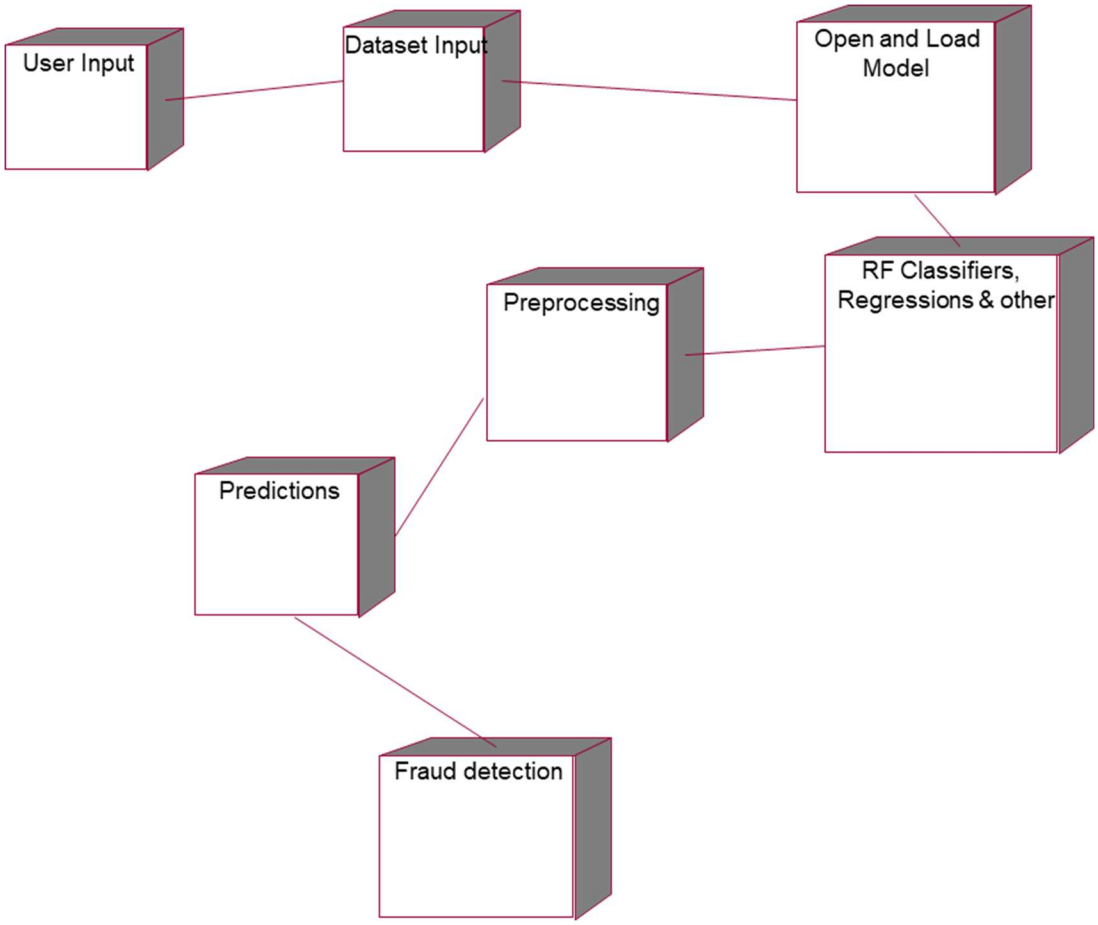
4.2.4 COMPONENT DIAGRAM



EXPLANATION

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.

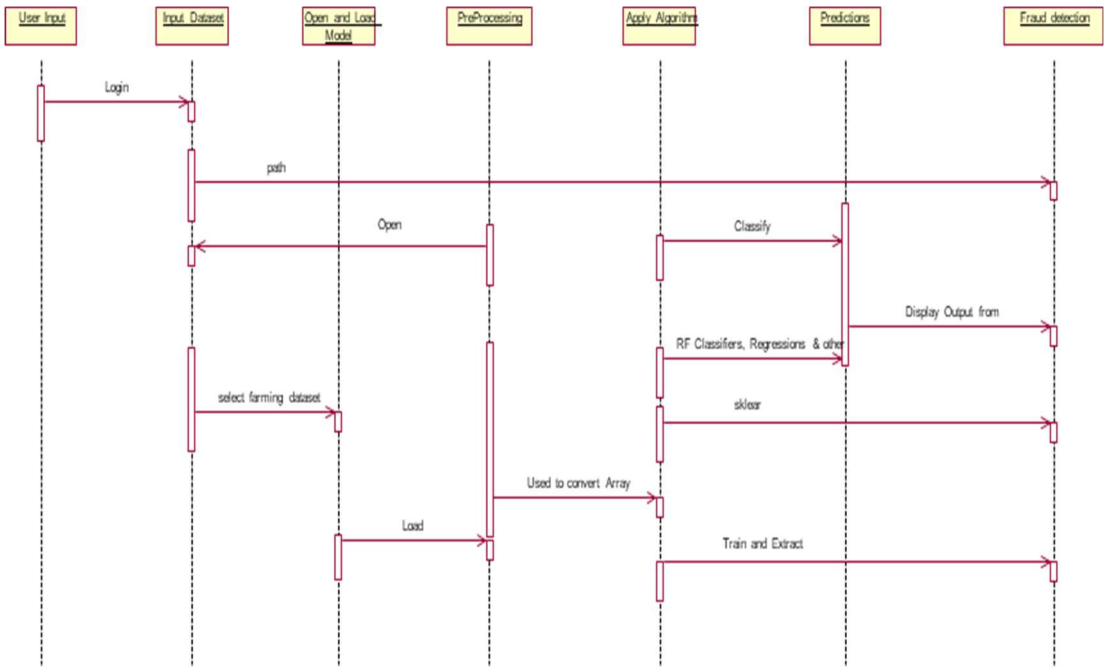
4.2.5 DEPLOYMENT DIAGRAM



EXPLANATION:

Deployment Diagram is a type of diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware. It maps software pieces of a system to the device that are going to execute it.

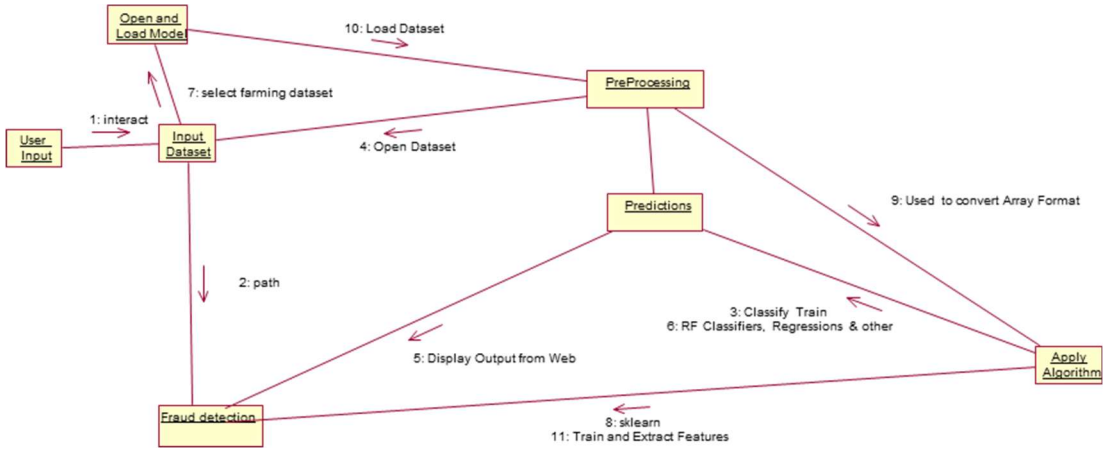
4.2.6 SEQUENCE DIAGRAM



EXPLANATION:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

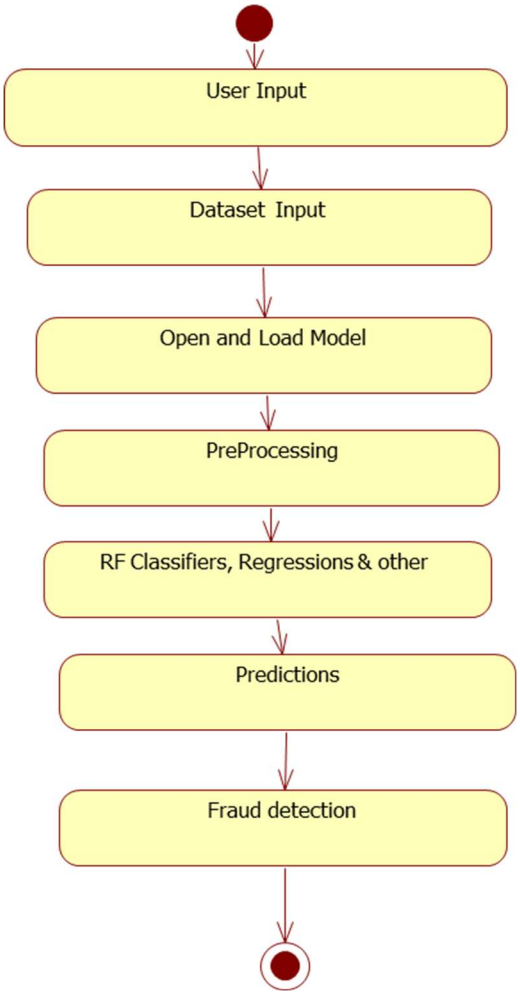
4.2.7 COLLABORATION DIAGRAM



EXPLANATION:

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

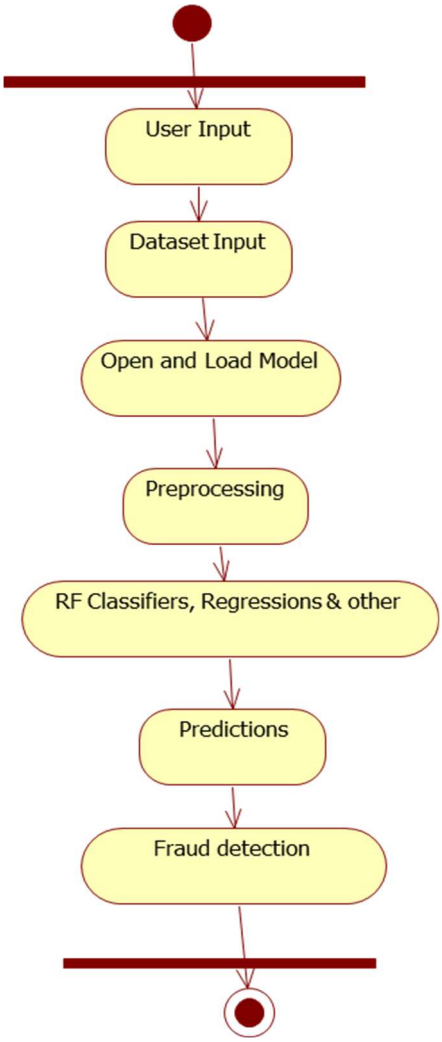
4.2.8 STATE DIAGRAM



EXPLANATION:

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

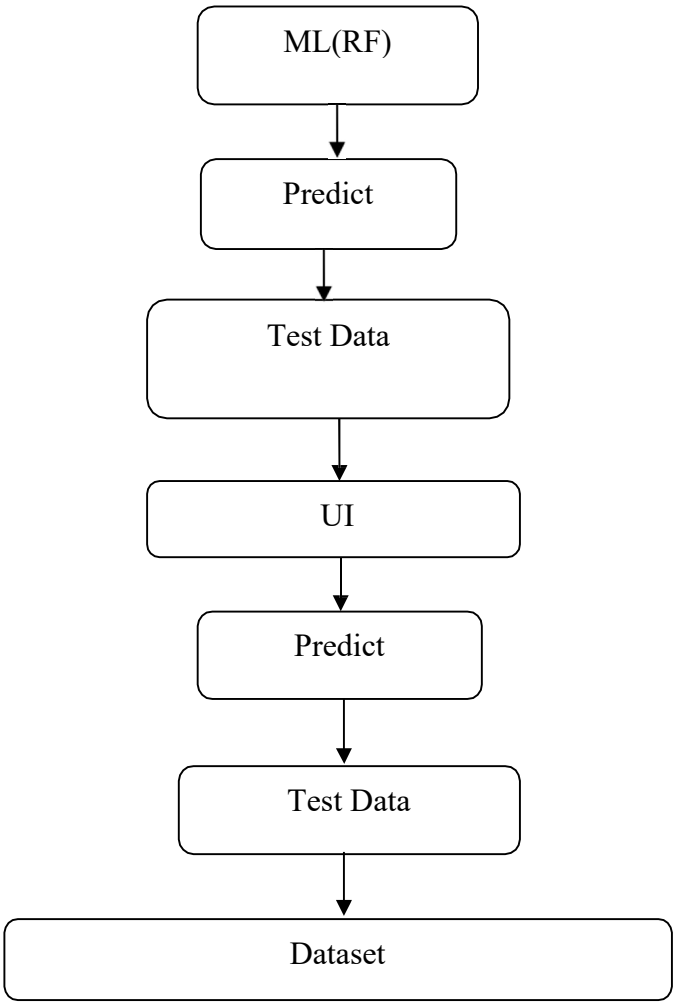
4.2.9 ACTIVITY DIAGRAM



EXPLANATION:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

4.3 DATA FLOW DIAGRAM



4.4 SYSTEM ARCHITECTURE

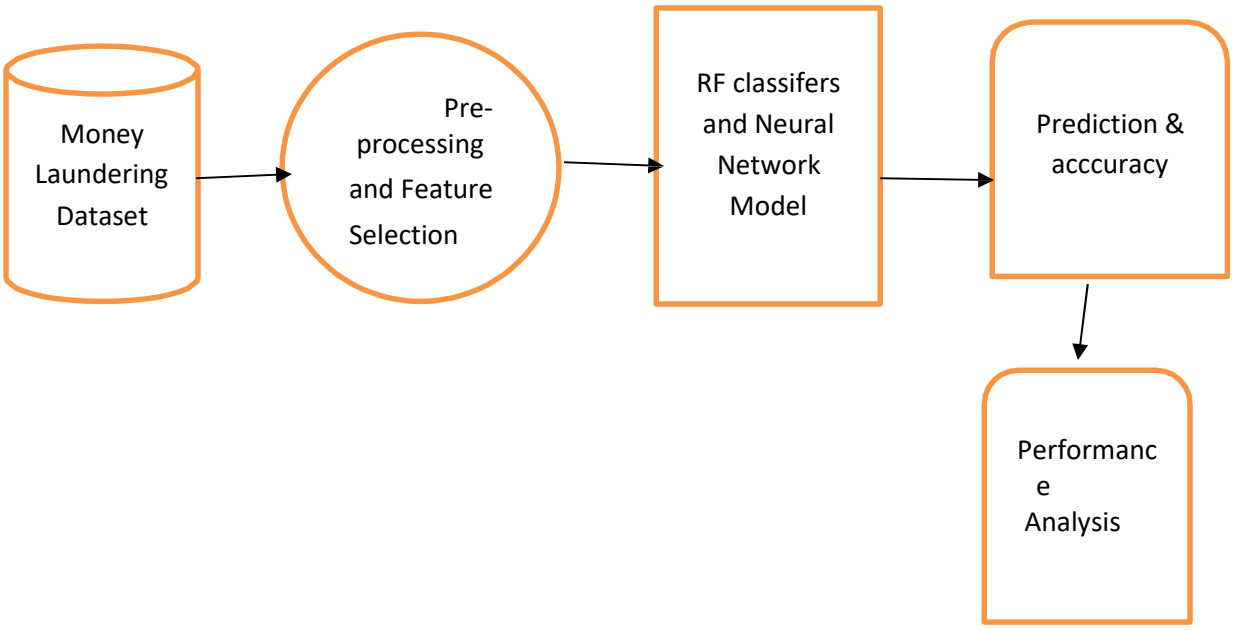


Fig. Proposed Methodology

CHAPTER 5

DEVELOPMENT TOOLS

5.1 PYTHON

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

5.2HISTORY OF PYTHON

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

5.3 IMPORTANCE OF PYTHON

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object–Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language** – Python is a great language for the beginner–level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

5.4 FEATURES OF PYTHON

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy–to–maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross– platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low–level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.

- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above–mentioned features, Python has a big list

of good features, few are Listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

5.5 LIBRARIES USED IN PYTHON:

- numpy - mainly useful for its N-dimensional array objects.
- pandas - Python data analysis library, including structures such as dataframes.
- matplotlib - 2D plotting library producing publication quality figures.
- scikit-learn - the machine learning algorithms used for data analysis and data mining tasks.

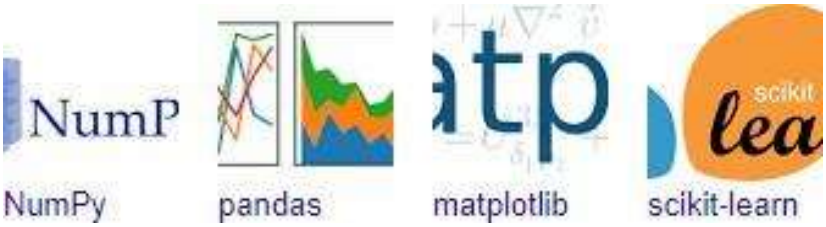


Figure: NumPy, Pandas, Matplotlib, Scikit-learn

CHAPTER 6

IMPLEMENTATION

6.1 GENERAL

6.2 CODING:

Importing Analysis Libraries

```
import streamlit as st
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import pickle

# ML Libraries
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_selection import mutual_info_regression
from sklearn.metrics import classification_report

# ---------------------- Header and Layout ----------------------
st.set_page_config(page_title="Internet Banking Fraud Detection", layout="wide")

st.markdown("<h1 style='text-align: center; color: darkblue;'>Bharat Institute of
Engineering and Technology</h1>", unsafe_allow_html=True)
st.markdown("<h2 style='text-align: center; color: green;'>Mini Project - Fraud
Detection in Internet Banking Using Machine Learning</h2>", unsafe_allow_html=True)
st.markdown("---")

# ---------------------- Sidebar Details ----------------------
st.sidebar.image("https://upload.wikimedia.org/wikipedia/commons/1/13/Bharat_Institute_
of_Engineering_and_Technology_logo.png", use_column_width=True)
st.sidebar.title("Team Details")
```

```
st.sidebar.write("*Batch:* 12")
st.sidebar.write("*Guide:* Mr./Ms. NAzeen Fathima")
st.sidebar.write("*Team Members:*")
st.sidebar.write("- VVS Siva Kumara Sarma")
st.sidebar.write("- o. rajesh")
st.sidebar.write("- k. udayram")
st.sidebar.write("- y. d sai krishna")
st.sidebar.write("- m . shiva prasad")
st.sidebar.write("- p . nitish")
st.sidebar.markdown("---")

st.sidebar.markdown("---")
model_option = st.sidebar.radio("🔍 Select Model for Evaluation", ["Logistic
Regression", "KNN", "Random Forest"])
test_size = st.sidebar.slider("Test Size", 0.2, 0.5, 0.33)

# ---------------------- Data Processing ----------------------
df = pd.read_csv('fraud_dataset_example_2.csv')
l = LabelEncoder()
df['type_code'] = l.fit_transform(df['type'])
X = df.drop(['isFlaggedFraud', 'isFraud', 'type', 'nameOrig', 'nameDest'], axis=1)
y = df['isFraud']

# ---------------------- Mutual Info Plot ----------------------
def make_mi_scores(X, y):
    mi_scores = mutual_info_regression(X, y)
    return pd.Series(mi_scores, name="MI Scores",
index=X.columns).sort_values(ascending=False)

def plot_mi_scores(scores):
    scores = scores.sort_values(ascending=True)
    width = np.arange(len(scores))
    ticks = list(scores.index)
    plt.figure(figsize=(8, 10))
    plt.barh(width, scores)
    plt.yticks(width, ticks)
    plt.title("Mutual Information Scores")
    st.pyplot(plt)

mi_scores = make_mi_scores(X, y)

# ---------------------- Model Training ----------------------
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size,
random_state=123)

lr = LogisticRegression().fit(X_train, y_train)
```

```python
knc = KNeighborsClassifier(n_neighbors=3).fit(X, y)
rf = RandomForestClassifier().fit(X_train, y_train)

lr_pred = lr.predict(X_test)
knc_pred = knc.predict(X_test)
rf_pred = rf.predict(X_test)

pickle.dump(knc, open("model.pkl", "wb"))

# ---------------------- Main Layout ----------------------
col1, col2 = st.columns(2)

with col1:
    st.subheader("📊 Dataset Overview")
    st.write(df.head())
    st.write("Shape:", df.shape)
    st.write("Fraud Count:", df['isFraud'].value_counts())

with col2:
    st.subheader("📈 Feature Importance")
    plot_mi_scores(mi_scores)

st.markdown("---")

# ---------------------- Model Results ----------------------
st.subheader(f"🔎 {model_option} - Evaluation Report")

if model_option == "Logistic Regression":
    st.text(classification_report(y_test, lr_pred))
    st.write("Accuracy:", metrics.accuracy_score(y_test, lr_pred))
elif model_option == "KNN":
    st.text(classification_report(y_test, knc_pred))
    st.write("Accuracy:", metrics.accuracy_score(y_test, knc_pred))
elif model_option == "Random Forest":
    st.text(classification_report(y_test, rf_pred))
    st.write("Accuracy:", metrics.accuracy_score(y_test, rf_pred))
    st.write("Train Accuracy:", rf.score(X_train, y_train))

st.markdown("---")

# ---------------------- Predict Section ----------------------
st.subheader("🧪 Live Fraud Prediction (KNN Model)")

input_data = {}
for feature in X.columns:
    val = st.number_input(f"{feature}", value=float(df[feature].mean()))
```

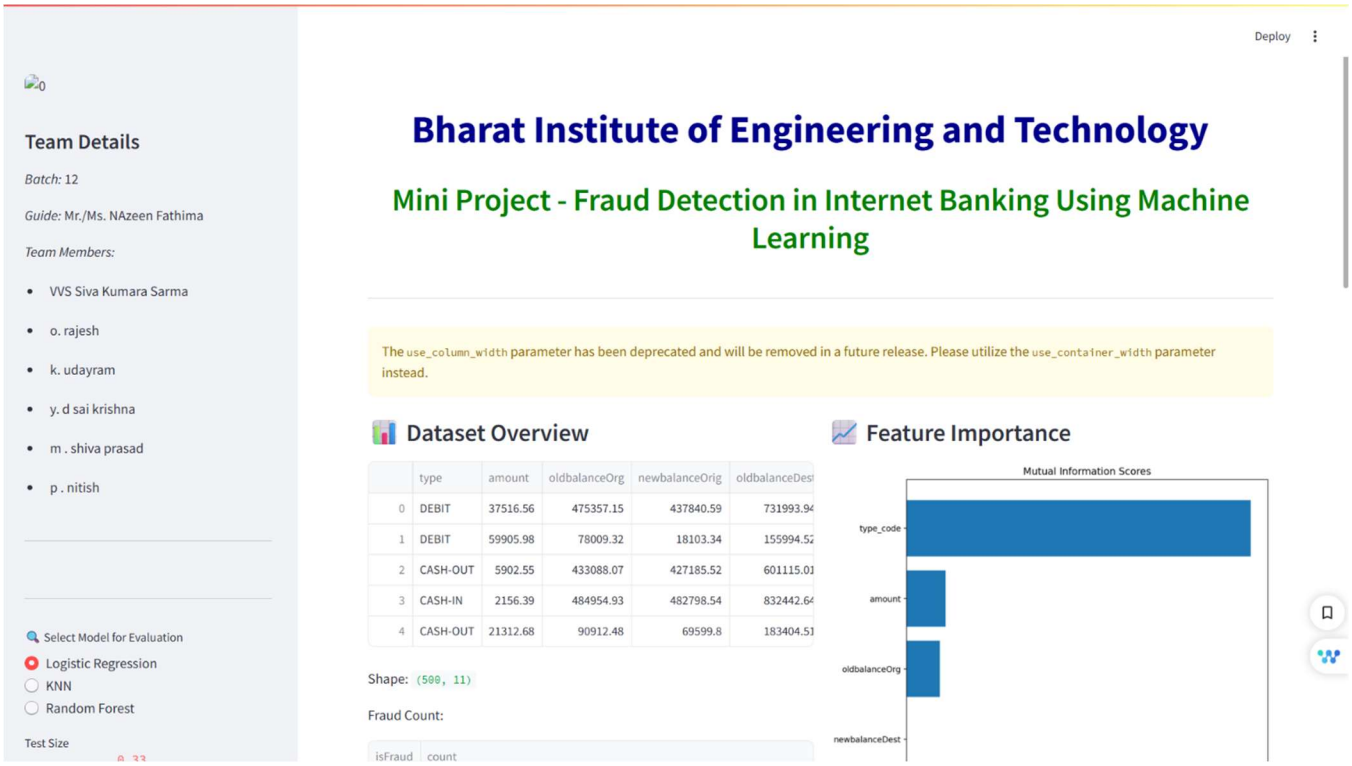
```
    input_data[feature] = val

if st.button("Predict Fraud"):
    model = pickle.load(open("model.pkl", "rb"))
    input_df = pd.DataFrame([input_data])
    pred = model.predict(input_df)
    if pred[0] == 1:
        st.error("⚠ Fraud Detected!")
    else:
        st.success("☑ No Fraud Detected.")
```

CHAPTER 7

SNAPSHOTS

7.1 GENERAL

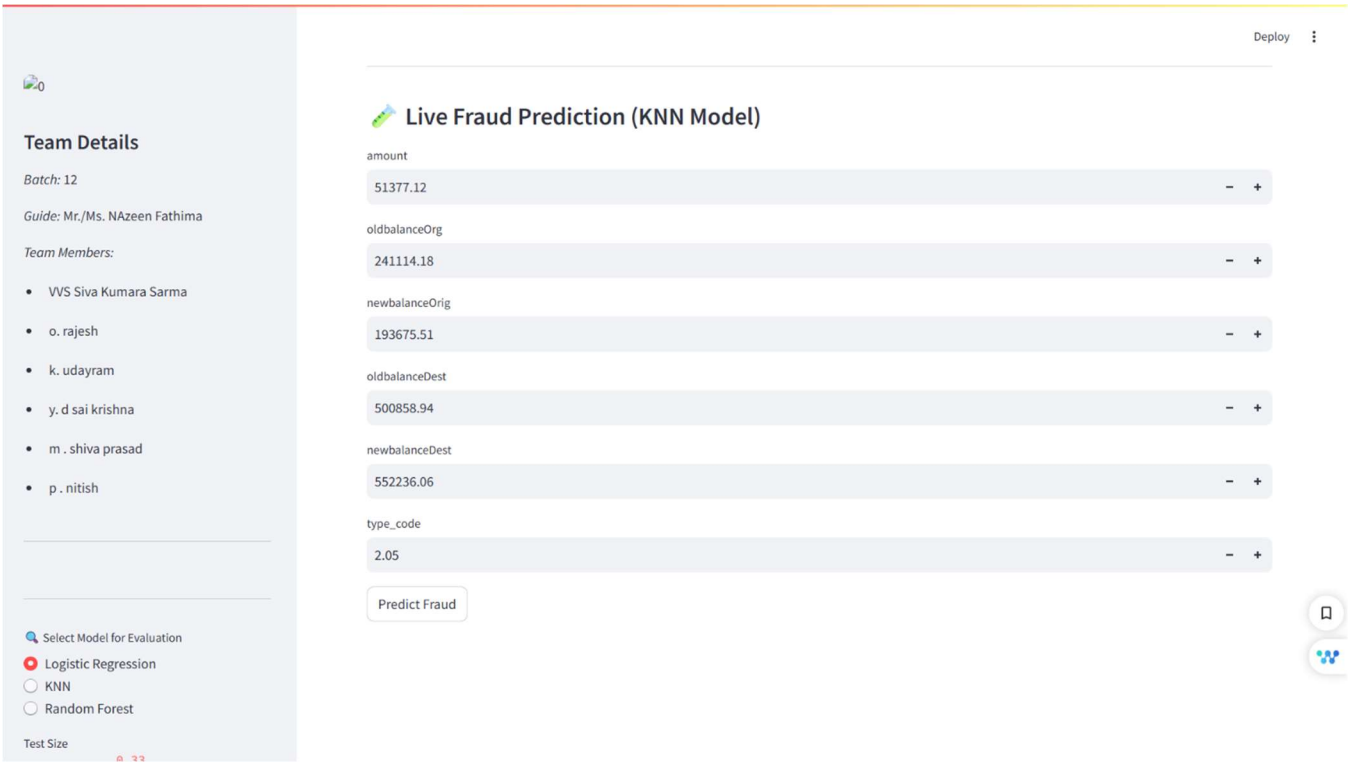
7.2 SNAPSHOTS



🔎 Logistic Regression - Evaluation Report

	precision	recall	f1-score	support
0	0.95	1.00	0.97	156
1	0.00	0.00	0.00	9
accuracy			0.95	165
macro avg	0.47	0.50	0.49	165
weighted avg	0.89	0.95	0.92	165

Accuracy: `0.9454545454545454`



CHAPTER 8

SOFTWARE TESTING

8.1 GENERAL

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.2 DEVELOPING METHODOLOGIES

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used.The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

8.3 TYPES OF TESTINGS

8.3.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.3.2 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected. Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised. Systems/Procedures: interfacing systems or procedures must be invoked.

8.3.3 SYSTEM TESTING

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

8.3.4 PERFORMANCE TESTING

The Performance test ensures that the output be produced within the time limits,and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

8.3.5 INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

8.3.6 ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

ACCEPTANCE TESTING FOR DATA SYNCHRONIZATION:

- The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
- The Route add operation is done only when there is a Route request in need
- The Status of Nodes information is done automatically in the Cache Updating process

8.3.7 BUILD THE TEST PLAN

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

CHAPTER 9

FUTURE ENHANCEMENT

9.1 FUTURE ENHANCEMENTS:

Feature enhancement in the context of machine learning involves improving the quality and relevance of the features used for model training, ultimately leading to enhanced predictive performance and generalization capabilities. There are several strategies for feature enhancement, including feature selection, feature engineering, and feature transformation. Feature selection aims to identify the most informative subset of features from the original feature space, reducing dimensionality and computational complexity while preserving or even improving model accuracy.

CHAPTER 10

CONCLUSIONAND REFERENCES

10.1 CONCLUSION

In conclusion, this study has demonstrated the effectiveness of machine learning algorithms, including Random Forest, K-Nearest Neighbours (KNN), and Logistic Regression, in detecting banking fraud transactions. By addressing the challenge of class imbalance through careful algorithm selection, feature enhancement techniques, and comprehensive evaluation metrics, we have developed robust models capable of accurately classifying transactions as fraudulent or legitimate. The findings highlight the importance of leveraging diverse approaches to fraud detection, as each algorithm brings its own strengths to the task.

10.2 REFERENCES

[1] M. Jullum, A. Løland, R. B. Huseby, G. A˚ nonsen, and J. Lorentzen, "Detecting money laundering transactions with machine learning," Journal of Money Laundering Control, vol. 23, no. 1, pp. 173 – 186, jan 2020.

[2] L. Keyan and Y. Tingting, "An improved support-vector network model for anti-money laundering," in 2011 Fifth International Conference on Management of e-Commerce and e-Government. IEEE, 2011, pp. 193 – 196.

[3] R. Liu, X.-l. Qian, S. Mao, and S.-z. Zhu, "Research on anti-money laundering based on core decision tree algorithm," in 2011 Chinese Control and Decision Conference (CCDC). IEEE, 2011, pp. 4322 – 4325.

[4] Z. Gao, "Application of cluster-based local outlier factor algorithm in anti-money laundering, " in 2009 International Conference on Management and Service Science. IEEE, 2009, pp. 1 – 4.

[5] J. de Jes´us Rocha Salazar, M. Jes´us Segovia-Vargas, and M. del Mar Camacho-Mi˜nano, " Money laundering and terrorism financing detection using neural networks and an abnormality indicator," Expert Systems with Applications, p. 114470, dec 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417420311209

[6] E. L. Paula, M. Ladeira, R. N. Carvalho, and T. Marzagao, "Deep learning anomaly detection as support fraud investigation in Brazilian exports and anti-money laundering," in 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, 2016, pp. 954 – 960.

[7] F. Anowar and S. Sadaoui, "Incremental Neural-Network Learning for Big Fraud Data," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 2020-Octob. Institute of Electrical and Electronics Engineers Inc., oct 2020, pp. 3551 – 3557.

[8] G. A. Carpenter and S. Grossberg, “A massively parallel architecture for a self-organizing neural pattern recognition machine,” Computer vision, graphics, and image processing, vol. 37, no. 1, pp. 54–115, 1987.

[9] G. Carpenter, “An adaptive resonance algorithm for rapid category learning and recognition,
” Neural Networks, vol. 4, pp. 439–505, 1991.

[10] T. Kohonen, “Self-organized formation of topologically correct feature maps,” Biological cybernetics, vol. 43, no. 1, pp. 59–69, 1982.

[11] A. Ultsch, “Kohonen’s self organizing feature maps for exploratory data analysis,” Proc. INNC90, pp. 305–308, 1990.

[12] Senator T E, Goldberg H G, Wooton J, etal. The Financial Crimes Enforcement Network AI System(FAIS)-identifying Potential Money Laundering from Reports of Large Cash Transactions[J] . AI Magazine, 1995, pp. 21-39.

[13] Zdanowicz John S. Detecting Money Laundering and Terrorist Financing Via Data Mining [J] . Communications of the ACM, 2004,pp.53-55.

[14] Bolton R J, Hand D J. Statistical Fraud Detection[J] . Statistical Science, 2002, pp. 235-254.

[15] Zhang Yan, Ouyang Yiming, Wang Hao, Wang Xidong, Application of Data Mining in the Financial Field Computer Engineering and Applications, vol.18, pp.208-211, 2004

[16] Yang Sheng-gang, Wang Peng, He Xue-hui, Exploring Decision Trees as a Tool to Investigate Money Laundering. Journal of Hunan University Social Sciences, vol.20, No.1, Jau. 2006, pp.65- 71.

[17] Eui-Hong Han. Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification. PhD thesis□University of Minnesota,1999.