A Mini Project Report

On

COST SENSITIVE PAYMENT FRAUD DETECTION BASED ON DYNAMIC RANDOM FOREST AND KNN

Submitted to JNTU
HYDERABAD

In Partial Fulfilment of the requirements for the Award of Degree of
BACHELOR OF TECHNOLOGY IN
CSE-AI&DS
Submitted by

Gattu Shivani	**(218R1A7225)**
Kanakapuram Kalyani	**(218R1A7234)**
Mohammad Haroon	**(218R1A7244)**
Boredsy Purushotham	**(218R1A7213)**

Under the Esteemed guidance
of

Mr. V. Kiran Kumar (Ph.D.)

Assistant Professor, Department of CSE(AI&DS)

Department of CSE-AI&DS

CMR ENGINEERING COLLEGE
(UGC AUTONOMOUS)
(Accredited by NBA, Approved by AICTE NEW DELHI, Affiliated to JNTU, Hyderabad) (Kandlakoya, Medchal Road, R.R. Dist. Hyderabad-501 401)

(2024-2025)

CMR ENGINEERINGCOLLEGE

(UGC AUTONOMOUS)

(Accredited by NBA, Approved by AICTE NEW DELHI, Affiliated to JNTU, Hyderabad) (Kandlakoya, Medchal Road, R.R. Dist. Hyderabad-501 401)

Department of CSE-AI&DS

CERTIFICATE

This is to certify that the project entitled **"Cost Sensitive Payment Fraud Detection Based On Dynamic Random Forest And KNN"** is a bonafide work carried out by

Gattu Shivani	**(218R1A7225)**
Kanakapuram Kalyani	**(218R1A7234)**
Mohammad Haroon	**(218R1A7244)**
Boredsy Purushotham	**(218R1A7213)**

In partial fulfilment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in **ARTIFICIAL INTELLIGENCE & DATA SCIENCE** from CMR Engineering College, affiliated to JNTU, Hyderabad, under our guidance and supervision.

The results presented in this project have been verified and are found to be satisfactory. The results embodied in this project have not been submitted to any other university for the award of any other degree or diploma.

Internal Guide	**Project Coordinator**	**Head of the Department**
Mr. V. Kiran Kumar (Ph.D) **Assistant Professor** **CSE(AI&DS),CMREC**	**Mr.R.Srikanth** **Assistant Professor** **CSE(AI&DS),CMREC**	**Dr.M.Laxmaiah** **Professor &H.O.D** **CSE(AI&DS),CMREC**

DECLARATION

This is to certify that the work reported in the present project entitled **“Cost Sensitive Payment Fraud Detection Based On Dynamic Random Forest And KNN”** is a record of bonafide work done by us in the Department of Computer science and Engineering (AI&DS), CMR Engineering College, JNTU Hyderabad. The reports are based on the project work done entirely by us and not copied from any other source. We submit our project for further development by any interested students who share similar interests to improve the project in the future.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

Gattu Shivani	**(218R1A7225)**
Kanakapuram Kalyani	**(218R1A7234)**
Mohammad Haroon	**(218R1A7244)**
Boredsy Purushotham	**(218R1A7213)**

ACKNOWLEDGEMENT

We are extremely grateful to **Dr. A. Srinivasula Reddy**, Principal and **Dr.M.Laxmaiah,** HOD, **Department of CSE-AI&DS, CMR Engineering College** for their constant support.

We are extremely thankful to **Mr. V.Kiran Kumar,** Assistant Professor, Internal Guide, Department of CSE-AI&DS, for his constant guidance, encouragement and moral support throughout the project.

We will be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred in this Project.

We thank **Mr. R.Srikanth,** Assistant professor, Mini project Coordinator for his constant support in carrying out the project activities and reviews.

We express our thanks to all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time.

Finally, We are very much thankful to our parents who guided us for every step.

Gattu Shivani	**(218R1A7225)**
Kanakapuram Kalyani	**(218R1A7234)**
Mohammad Haroon	**(218R1A7244)**
Boredsy Purushotham	**(218R1A7213)**

CONTENTS

ABSTRACT

The act of fraudulent credit card transaction has been increased over the past recent years, as the era of digitization hits our day-to-day life, with people are getting more involved in online banking and online transaction system. Machine learning algorithms have played a significant role in detection of credit card frauds. However, the unbalanced nature of the real-life datasets causes the traditional classification algorithms to perform low in detection of credit card fraud. In this work, a cost-sensitive weighted random forest algorithm has been proposed for effective credit card fraud detection. A cost-function has been defined in the training phase of each tree, in bagging which emphasizes to assign more weight to the minority instances during training. The trees are ranked according to their predictive ability of the minority class instances. The proposed work has been compared with two existing random-forest based techniques for two binary credit card datasets. The efficiency of the model has been evaluated in terms G-mean, F-measure and AUC values. The experimental results have established the proficiency of the proposed model, than the existing ones.

Keywords: Fraudulent Credit Card, Random Forest, Cost-sensitive

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

1.Introduction

Payment card fraud leads to heavy annual financial losses around the world, thus giving rise to the need for improvements to the fraud detection systems used by banks and financial institutions. In the academe, as well, payment card fraud detection has become an important research topic in recent years. With these considerations in mind, we developed a method that involves two stages of detecting fraudulent payment card transactions. The extraction of suitable transactional features is one of the key issues in constructing an effective fraud detection model. In this method, additional transaction features are derived from primary transactional data. A better understanding of cardholders spending behaviours is created by these features. After which the first stage of detection is initiated. A cardholders spending behaviours vary over time so that new behaviour of a cardholder is closer to his/her recent behaviours. Accordingly, a new similarity measure is established on the basis of transaction time in this stage. This measure assigns greater weight to recent transactions. In the second stage, the dynamic random forest algorithm is employed for the first time in initial detection, and the minimum risk model is applied in cost-sensitive detection. We tested the proposed method on a real transactional dataset obtained from a private bank. The results showed that the recent behaviour of cardholders exerts a considerable effect on decision-making regarding the evaluation of transactions as fraudulent or legitimate. The findings also indicated that using both primary and derived transactional features increases the F-measure. Finally, an average 23% increase in prevention of damage (POD) is achieved with the proposed cost- sensitive approach. The term credit card fraud signifies any act of theft and fraud, occurred in case of any payment card (debit or credit), due to physical loss of the card by the owner, or stealing of the card by fraudsters, or by means techniques like phishing, skimmer, identity theft etc. Financial fraud of this type can greatly affect corporate, organizational and government sectors of a country. The rate of fraudulent transactions has increased in today's era of internet technology, where credit card transaction has become the most convenient way of transaction, whether online of offline. As discussed in there are two types of credit card fraud can happen, internal fraud and external fraud. The first type depicts a situation where there is leakage of information between the cardholder and bank, by means of false identity; whereas in the later case, fraud happens due to stolen or lost credit card get into some fraudsters' hands. To solve this problem, credit card fraud detection techniques have been developed by researchers over the past years. Basically it involves of classifying the credit card transactions either as "legitimate" or "fraudulent". A number of Machine Learning (ML) techniques have been employed for this task, such as Decision tree Support Vector Machine (SVM), Naïve Bayes (NB), Artificial Neural network (ANN), and optimization techniques such as genetic algorithm, migrating birds optimization algorithm. However, the task of credit card fraud has to face some adversities such as, (a) unavailability of real-life credit transaction dataset, (b) imbalanced ratio of "legitimate" vs. "fraudulent" transactions, (c) enormous size of the dataset, and (d) dynamic behaviour of the fraudsters. As a result, effective credit card fraud detection demands effective pre-processing of the dataset, prior to applying of any ML techniques. Each tree is then trained by using a weak-classifier (C4.5/J48), and then the test set is validated with trained trees.

1.2 Project Objectives

The objective of this project is to develop a cost-sensitive payment fraud detection system using Dynamic Random Forest and K-Nearest Neighbors to minimize financial losses while maintaining accurate fraud detection.

1.3 Purpose of the Project

The purpose of this project is to enhance payment security by developing a system that effectively detects and mitigates payment fraud while considering the financial impact of false alarms

1.4 Existing System with Disadvantages

However, the chances of achieving adequate result is still questionable as the cases of imbalanced data are present, in each tree. So, to improve the prediction ability of each tree, and the overall performance of the ensemble, a cost- function is defined, based on the misclassification rate of the instances, both majority and minority. The trees are given weightage based on the quantity of error, they are yielding to the ensemble. The trees with lower error-rate are given higher weightage while determining the final outcome of the test object.

Disadvantages

- Traditional methods may not handle imbalanced datasets effectively, leading to a higher likelihood of false negatives (missed fraud) or false positives (legitimate transactions flagged as fraud).

- They might not adapt well to changing fraud patterns, making it difficult to keep up with evolving fraud techniques and reducing their overall effectiveness.
- Conventional models may not readily incorporate cost-sensitive learning techniques, leading to higher operational costs due to manual intervention in fraud cases and potential revenue loss in the case of false positives.

1.5 Proposed System with Features

In this paper, the imbalanced nature of the credit card data has been taken into consideration, and a cost-sensitive weighted random forest technique has been proposed to overcome the issue. The Random Forest (RF) algorithm is basically an ensemble learning technique which works on the principle of Bagging, i.e. the dataset has been divided into n-bags or samples, with randomly selected instances, and each of which is termed as “Tree” The final output of the model is achieved based on majority voting (in case of classification model), or averaging (in case of regression model) of all the test outcomes, with respect to all the trained trees. The foremost advantage of using RF technique for credit card fraud detection is that it can readily deal with the enormous size the training data, as RF involves to divide the dataset into a number samples, and then learn. Through the proposed approach, a cost- driven learning scheme is adapted to give more emphasis on learning the minority class instances. The rest of the paper is organized as follows. In the proposed scheme, a cost- sensitive weighted RF algorithm has been proposed for credit card fraud detection. A confusion -matrix based cost-function is defined to put more emphasis on the minority class instances during training. Instead of conventional majority voting strategy, a weighing strategy, based on minimum error achieved for a single tree.

K-Nearest Neighbor (KNN) Algorithm

- K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

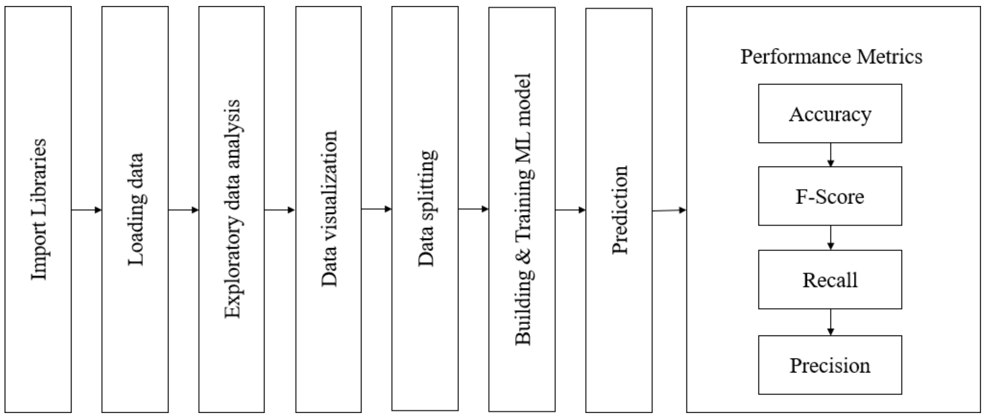


Figure 1.5.1: Block diagram of proposed system.

Advantages

- It minimizes financial losses by considering the financial impact of false alarms, optimizing resources, and reducing manual intervention.
- The system can readily adapt to changing fraud tactics and transaction patterns, staying effective over time.

- Dynamic Random Forest offers the power of ensemble learning, combining multiple models to boost predictive performance.
- It efficiently selects relevant features, contributing to the overall accuracy of fraud detection.

1.6 INPUT AND OUTPUT DESIGN

Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Objectives

- Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
- It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
- When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow.

Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and toother system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

- Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
- Select methods for presenting information.
- Create document, report, or other formats that contain information produced by the system.
- The output form of an information system should accomplish one or more of the following objectives.
- Convey information about past activities, current status or projections of the Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

2. LITERATURE SURVEY

L. Breiman, Random forests. Machine Learning, vol. 45, issue 1, pp: 5-32, 2021 [1] has concluded that Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges a.s. to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them. Using a random selection of features to split each node yields error rates that compare favorably to Adaboost (Y. Freund & R. Schapire, Machine Learning: Proceedings of the Thirteenth International conference, ∗∗∗, 148– 156), but are more robust with respect to noise. Internal estimates monitor error, strength, and correlation and these are used to show the response to increasing the number of features used in the splitting. Internal estimates are also used to measure variable importance. These ideas are also applicable to regression.

G. Singh, R. Gupta, A. Rastogi, M. D. S. Chandel, and A. Riyaz, A Machine Learning Approach for Detection of Fraud based on SVM, International Journal of Scientific Engineering and Technology, vol.1, issue 3, pp. 194-198, 2021.[2] has concluded that the growth of e-commerce increases the money transaction via electronic network which is designed for hassle free fast & easy money transaction but the facility involves greater risk of misuse of facility for fraud one of them is credit card fraud it can be happened by many types as by stolen card, by internet hackers who can hack your system & get important information about your card, or by information leakage during the transaction, although many person has proposed their work for credit card fraud detection by characterizing the user spending profile, but in this paper we are proposing the SVM (support vector machine) based method with multiple kernel involvement also including several fields of user profile instead of only spending profile & the simulation result shows improvement in TP (true positive),TN(true negative) rate, it also decreases the FP(false positive) & FN(false negative) rate.

C. Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, Improving credit card fraud detection with calibrated probabilities. In Proceedings of the 2020 SIAM International Conference on Data Mining, pp. 677-685, 2021.[3] has concluded that credit card fraud detection involves a largely differenced misclassification costs between the legitimate and the fraudulent cases. The misclassification cost is a vital factor while performing any classification task. In presence of imbalanced data, adequate definition of the misclassification cost is desirable, while offering cost-sensitive learning based solutions. Traditionally, the credit card fraud detection needs lot of data pre-processing (feature selection/ extraction, sampling, outlier detection), followed by machine learning algorithms to detect the legitimate/ fraudulent cases. Designing a cost-sensitive learning framework in order to treat the imbalanced cases by incorporating the highly differenced misclassification costs of the credit card datasets is the motivation of this work. The challenge of highly differenced misclassification cost of credit card fraud detection has been treated by incorporating it to the RF-Bagging ensemble learning model so that achieving of low positive (minority) class error can have more emphasis. The tree with lowest error can have more weights while defining the learning function. Hence, a cost-sensitive weighted random forest approach is a promising one to treat two different issues of credit card fraud detection, namely imbalanced nature of the data -space and highly differenced misclassifications costs. The contribution of the proposed work has been focused within this aspect.

K. RamaKalyani, and D. UmaDevi, Fraud Detection of Credit Card Payment System by Genetic Algorithm, International Journal of Scientific & Engineering Research, vol. 3, issue 7, pp. 1 – 6, 2021.[4] has concluded that mainly two approaches namely misuses (supervised) and anomaly detection (unsupervised) technique is being used. After this a classification is also used for checking the capability to process categorical and numerical data. In the first approach the data is classified as fraud based on previous data. With the help of this dataset classification models are also created, which can predict whether the data is fraud or not. The different classification models used are decision tree, neural network, rule induction etc. This has obtained a successful result and this approach is also called as misuse approach. While the second approach is based on account behaviour. A transaction is said to be fraudulent if it possess the features opposite to the user's normal behaviour. The behaviour of user's model are extracted and accordingly classified as fraudulent or not. This technique of finding fraud is also called as anomaly detection.

J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare, Credit card fraud detection using Machine Learning Techniques: A Comparative Analysis, pp: 1-9, 2020.[5] has concluded that as the information technology is developing the fraud is also increasing as a result financial loss due to fraud is also very large. A cost sensitive decision tree approach has been used for fraud detection. A cost called misclassification cost is used which is taken as varying as well as priorities of the fraud also differs according to individual records. So common performance metrics such as accuracy, True Positive Rate (TPR) or even area Under Curve cannot be used to evaluate the performance of the models because they accept each fraud as having the same priority regardless of the amount of that fraudulent transaction or the available usable limit of the card used in the transaction at that time.

P. L., Meshram, and P. Bhanarkar, Credit and ATM Card Fraud Detection 2019 [6] has concluded that As the E-commerce technology is increasing day by day the use of credit card has also been increased. As a result of this the fraud using credit card is also increasing. In all fraud detection systems, fraud will be detected only after the fraud has taken place. In this study a sequence of operations are modelled using Hidden Markov Model (HMM) and this can be used for the detection of fraud. It is trained with the normal behaviour of the card holder. If the incoming transaction is not accepted by the trained HMM with high probability it is considered as fraudulent otherwise not.
A hidden Markov Model represents a finite number of states with sufficiently high probability. The transition between the states are handled by these probability values. A possible outcome will be generated based on the probability distribution. This outcome will be visible to the external users that is the states are hidden to the users hence the name. It is a perfect solution for predicting fraud transactions in addition it also provides extreme decrease in the number of false positive transactions recognized by fraud detection system. For prediction purpose three values are being used namely low, medium, high.

P. L., Meshram, and P. Bhanarkar, Credit and ATM Card Fraud Detection Using Genetic Approach, International Journal of Engineering Research & Technology (IJERT), vol. 1, issue 10, pp. 1 – 5, 2019. [7] has concluded that the main advantage is that it does not require fraud signatures it is capable to detect by bearing in mind card holders spending habit. This is done by creating a set of clusters and identify the spending profile. These data are stored in the form of clusters with low, medium and high values.

The probability is based on the spending behaviour and further the processing is done. If the transaction is found to be fraudulent an alarm is generated, in addition to this a security form will also arise bearing a certain number of questions. This model can detect fraud transactions to an extent. It is scalable in handling large amount of data.

K. R. Seeja, and M. Zareapoor, FraudMiner: A Novel Credit Card Fraud Detection Model Based on Frequent Itemset Mining, The Scientific World Journal, Hindawi Publishing Corporation, vol. 2018 [8] has concluded that in this study mainly two approaches namely misuses (supervised) and anomaly detection (unsupervised) technique is being used. After this a classification is also used for checking the capability to process categorical and numerical data. In the first approach the data is classified as fraud based on previous data. With the help of this dataset classification models are also created, which can predict whether the data is fraud or not. The different classification models used are decision tree, neural network, rule induction etc. This has obtained a successful result and this approach is also called as misuse approach. While the second approach is based on account behaviour. A transaction is said to be fraudulent if it possess the features opposite to the user's normal behaviour. The behaviour of user's model are extracted and accordingly classified as fraudulent or not. This technique of finding fraud is also called as anomaly detection.

V, Patil, U. K. Lilhore, A Survey on Different Data Mining & Machine Learning Methods for Credit Card Fraud Detection, International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Vol. 3, Issue 5, pp:320-325, 2017.[9] has concluded that Different methods are used for cost sensitivity. They mainly include the machine learning approach, decision tree approach. In machine learning approach two techniques called over sampling and under sampling is performed, in which the latter obtained a good result. In decision tree approach, decision tree algorithms are used in which misclassification cost is considered in pruning step. A cost matrix is used to find the varying misclassification cost. After finding the misclassification cost the one with minimum value is used. By finding the misclassification cost not only the node value is obtained but also it predicts whether the transaction is fraudulent or not. This study using misclassification cost has made a significant improvement in fraud detection.

M. Zareappor, P. Shamsolmoali, Application of Credit card Fraud Detection: Based on Bagging Ensemble Classifier, International Conference on Intelliegent Computing, Communication, & Convergence (ICCC-2016) [10] has concluded that as the information technology is developing the fraud is also increasing as a result financial loss due to fraud is also very large. A cost sensitive decision tree approach has been used for fraud detection. A cost called misclassification cost is used which is taken as varying as well as priorities of the fraud also differs according to individual records. So common performance metrics such as accuracy, True Positive Rate (TPR) or even area Under Curve cannot be used to evaluate the performance of the models because they accept each fraud as having the same priority regardless of the amount of that fraudulent transaction or the available usable limit of the card used in the transaction at that time. For avoiding this a new performance metric which prioritizes each fraudulent transaction in a meaningful way and it also checks the performance of the model in minimizing the total financial loss. The measure used is Saved Loss Rate (SLR) which is the saved percentage of the potential financial loss that is the sum of the available usable limits of the cards from which fraudulent transactions are committed.

3. SOFTWARE REQUREIMENTS ANALYSIS

3.1 Problem Statement

This project aims to address these challenges by developing a cost-sensitive payment fraud detection system based on Dynamic Random Forest and KNN. The primary objective is to enhance the accuracy of fraud detection while minimizing financial losses. This system will adapt to changing fraud patterns, effectively handle imbalanced data, and incorporate cost- sensitive learning to ensure that both the financial impact of false alarms and operational costs are minimized. By leveraging ensemble learning and distance-based analysis, the proposed solution will provide a more robust defense against payment fraud, ultimately safeguarding the financial interests of businesses and their customers.

3.2 Modules and Their Functionalities

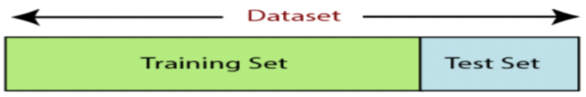
Data Preprocessing

After collecting datasets from various resources. Dataset must be pre-processing before training to the model. The data pre-processing can be done by various stages, begins with reading the collected dataset the process continues to data cleaning. In data cleaning the datasets contain some redundant attributes, those attributes are not considering for phishing detection. So, we have to drop unwanted attributes and datasets containing some missing values we need to drop these missing values in order to get better accuracy.

- Getting the dataset
- Importing libraries
- Importing datasets
- Splitting dataset into training and test set
- Train the classifier
- Test the classifier
- Evaluate

Splitting the Data set into the Training set and Test set

In machine learning data pre-processing, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this, we can enhance the performance of our machine learning model. Suppose if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models. 11 If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

Train the classifier:

Training the classifier with the training data by specifying the value of k. Use k =3 for binary classification, i.e., two labels classification. If used k =1 then it is simply a nearest neighbor classifier. **Test the classifier**: Testing the classifier with the testing data.

Evaluate: Evaluating the classifier using confusion matrix and its evaluation metrics i.e.accuracy, precision, recall, etc.

3.3 Functional Requirements

It provides the users a clear statement of the functions required for the system in order to solve the project information problem it contains a complete set of requirements for the applications. A requirement is condition that the application must meet for the customer to find the application satisfactory. A requirement has the following characteristics:

- It provides a benefit to the origination.
- It describes the capabilities the application must provide in business terms.
- It does not describe how the application provides that capability.

- It is stated in unambiguous words. Its meaning is clear and understandable.
- It is verifiable.

3.4 Non-Functional Requirements

Career recommendation non-functional requirements, like interests he has, how hours he can work likewise, with today's IT projects, to determine non-functional requirements, like availability, the approach requires that the designer 1st determine the scope: does the whole solution or only part of it need to be architected to meet minimum levels?

- This is done through 4 steps:
- Identify the critical areas of solutions
- Identify the critical components within each critical area.
- Determine each components availability and risk.
- Model worst-case failure scenarios.

3.5 Feasibility Study

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

4. SOFTWARE AND HARDWARE REQUIREMENTS

4.1 Software Requirements

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

Operating system	:	Windows 10
Coding Language	:	Python
Tool	:	PyCharm
Database	:	MYSQL
Server	:	Flask

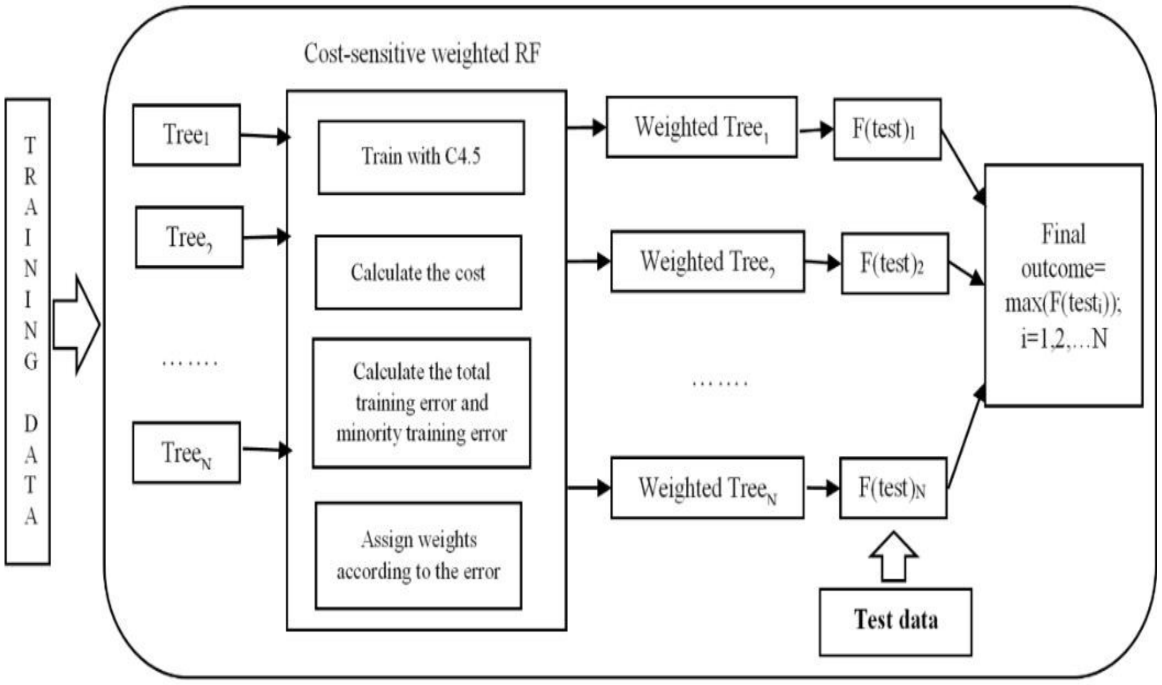
4.2 Hardware Requirements

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

System	:	Pentium Dual Core.
Hard Disk	:	120 GB.
Monitor	:	15” LED
Input Devices	:	Keyboard,Mouse
Ram	:	4 GB

5. SOFTWARE DESIGN

5.1 System Architecture



TRAINING DATA
Cost-sensitive weighted RF
Tree1
Tree,
....
TreeN
Train with C4.5
Calculate the cost
Calculate the total training error and minority training error
Assign weights according to the error
Weighted Tree1
Weighted Tree,
Weighted TreeN
F(test)1
F(test)2
F(test)N
Final outcome= max(F(testi)); i=1,2,...N
Test data

Figure:5.1 System Architecture

We need to upload the dataset of selected feature values where feature extraction is done from the Phishing URLs and non-Phishing URLs which is called as trained data set. This is used to train the KNN model. Dataset of selected feature values is given as input to the KNN model and output is generated whether the given input is Phishing or No phishing. Accuracy is generated based on the KNN Model.

5.2 Dataflow Diagram

The data flow diagram (DFD) is one of the most important modelling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

- DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

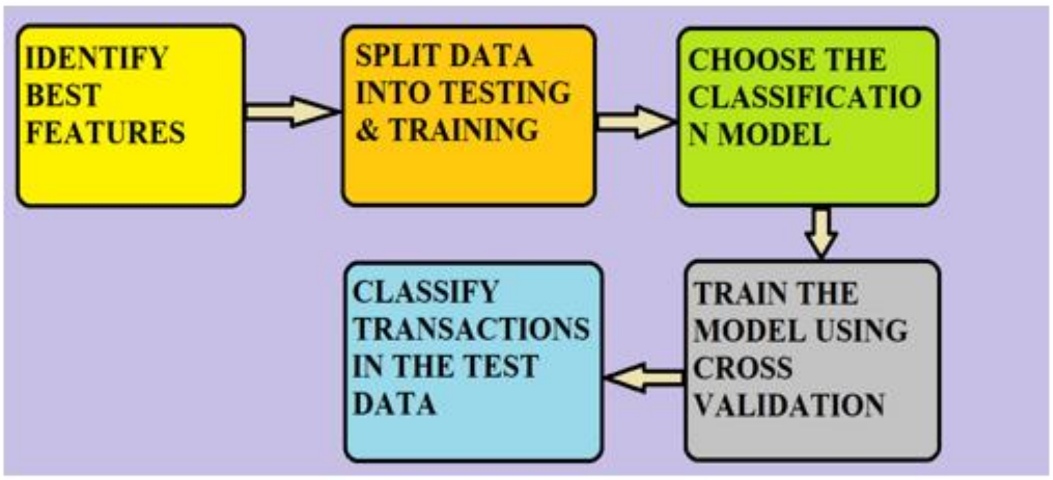

IDENTIFY BEST FEATURES
SPLIT DATA INTO TESTING & TRAINING
CHOOSE THE CLASSIFICATIO N MODEL
TRAIN THE MODEL USING CROSS VALIDATION
CLASSIFY TRANSACTIONS IN THE TEST DATA

Figure:5.2 Dataflow diagram

5.3 UML Diagrams

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

There are several types of UML diagrams and each one of them serves a different purpose regardless of whether it is being designed before the implementation or after (as part of documentation). UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard. The two broadest categories that encompass all other types are:

- Behavioural UML diagram and
- Structural UML diagram.

As the name suggests, some UML diagrams try to analyses and depict the structure of a system or process, whereas other describe the behaviour of the system, its actors, and its building components.

GOALS: The Primary goals in the design of the UML are as follows:

• Provide users a ready-to-use, expressive visual modelling Language so that they can

• Provide extendibility and specialization mechanisms to extend the core concepts.

• Be independent of particular programming languages and development process.

• Provide a formal basis for understanding the modelling language.

• Encourage the growth of OO tools market.

• Support higher level development concepts such as collaborations, frameworks, patterns and components.

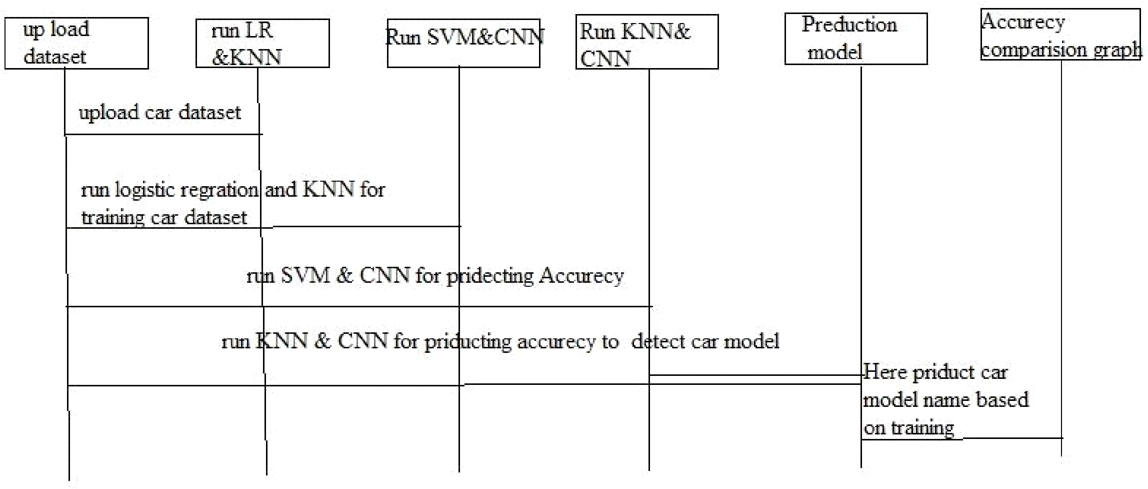
• Integrate best practices.

The different types are as follows:

- Sequence diagram
- Use case Diagram
- Activity diagram
- Class diagram

Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.



up load dataset
run LR &KNN
Run SVM&CNN
Run KNN& CNN
Preduction model
Accurecy comparision graph
upload car dataset
run logistic regration and KNN for training car dataset
run SVM & CNN for pridecting Accurecy
run KNN & CNN for pricucting accurecy to detect car model
Here priduct car model name based on training

Figure 5.3.1 Sequence diagram

List of actions User:

User need to press any of the given three (i.e., Prediction data, prediction Skills) then he will get the output accordingly.

System:

System will give the output as he enters according to the given data.

Result:

As per user enters the data it will give whether it is Fraud or not fraud

Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use case in which the user is involved. A use case diagram is used to structure of the behaviour thing in a model. The usecases are represented by either circles or ellipses.

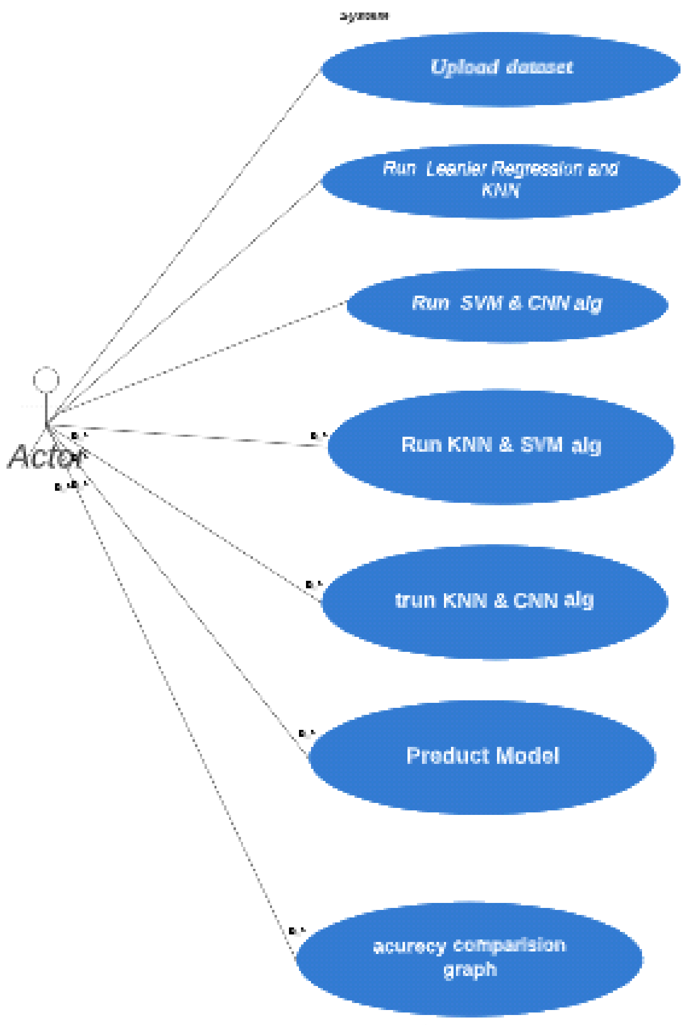


Figure 5.3.2 Use case diagram

Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

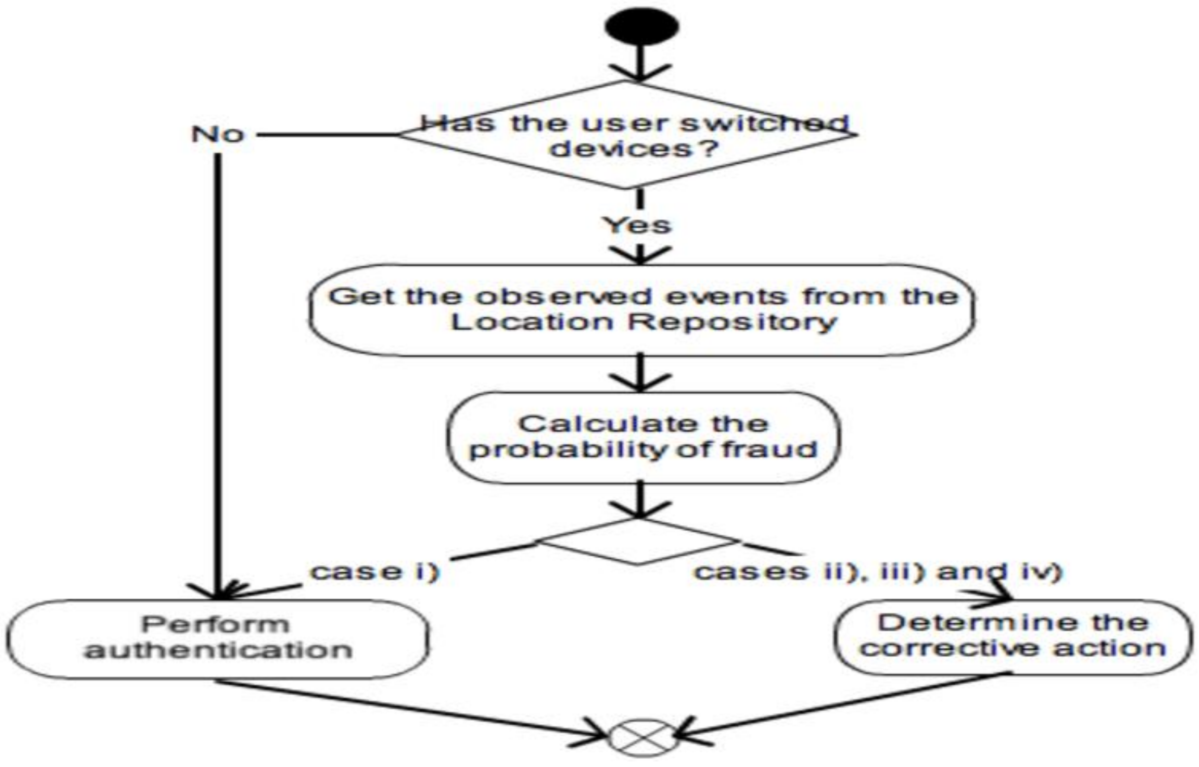

Has the user switched devices?
No
Yes
Get the observed events from the Location Repository
Calculate the probability of fraud
case i)
cases ii), iii) and iv)
Perform authentication
Determine the corrective action

Figure 5.3.3 Activity diagram

Class Diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

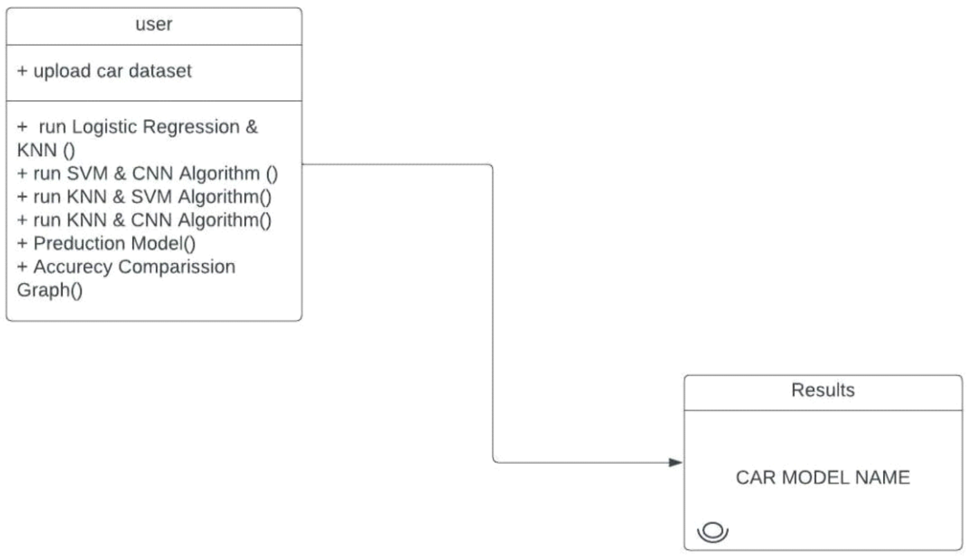

user
+ upload car dataset
+ run Logistic Regression & KNN ()
+ run SVM & CNN Algorithm ()
+ run KNN & SVM Algorithm()
+ run KNN & CNN Algorithm()
+ Preduction Model()
+ Accurecy Comparission Graph()
Results
CAR MODEL NAME

Figure 5.3.4 Class diagram

6. CODING AND ITS IMPLEMENTATION

6.1 Source code

```
# -*- coding: utf-8 -*-
from flask import Flask, render_template,
request import numpy as np
import pandas
as pd
import joblib
app=Flask(_na
me__)
_@app.route('/'
)
def index():
   return
render_template('login.html')
@app.route("/signup")
def signup():
   name=request.args.get('username
   1)
   mail  =  request.args.get('mail1','')
   password                         =
   request.args.get('password1','')    if
   len(name) == 0 and len(password)
   == 0:
      return
   render_template("login.html")
   else:
        return
render_template("login.html")
@app.route("/signin")
def signin():
    mail1 =
    request.args.get('username','')
    password1 =
    request.args.get('password','') if
    len(mail1) == 0 and len(password1)
    == 0:
        return render_template("login.html")
    else:
        return render_template("index.html")
@app.route('/predict', methods=['GET',
'POST']) def predict():
    if request.method ==
        'POST': try:
            Time =
            float(request.form['Time'])
            V1 =
            float(request.form['V1'])
            V2 =
            float(request.form['V2'])
            V3 =
            float(request.form['V3'])
            V4 =
            float(request.form['V4'])
            V5 =
float(request.form['V5'])
V6 =
float(request.form['V6'])
V7 =
float(request.form['V7'])
V8 =
float(request.form['V8'])
V9 =
float(request.form['V9'])
V10 =
float(request.form['V10']
) V11 =
float(request.form['V11']
) V12 =
float(request.form['V12']
) V13 =
float(request.form['V13']
) V14 =
float(request.form['V14']
) V15 =
float(request.form['V15']
) V16 =
float(request.form['V16']
) V17 =
float(request.form['V17']
) V18 =
float(request.form['V18']
) V19 =
float(request.form['V19']
) V20 =
float(request.form['V20']
) V21 =
float(request.form['V21'}
)

V22                    =
float(request.form['V22']
)          V23         =
float(request.form['V23']
)          V24         =
float(request.form['V24']
)          V25         =
float(request.form['V25']
)          V26         =
float(request.form['V26']
)          V27         =
float(request.form['V27']
)          V28         =
float(request.form['V28']
)
Amount = float(request.form['Amount'])
# Now we will create the list in order to pass the value to the
model pred_args = [Time, V1, V2, V3, V4, V5, V6, V7, V8,
V9, V10, V11, \
        V12, V13, V14, V15, V16, V17, V18, V19,V20,
        V21, V22,\ V23, V24, V25, V26, V27, V28,
        Amount]
```

```
            pred_agrs_arr = np.array(pred_args)
            pred_agrs_arr =
            pred_agrs_arr.reshape(1,-1) ml_rdm_frt
            = open("Random_forest.pkl", "rb")
            ml_model = joblib.load(ml_rdm_frt)
            model_prediction =
            ml_model.predict(pred_agrs_arr)
            model_prediction = int(model_prediction)
        except Value Error:
            return "Please check if the values are entered correctly"
    return render_template('predict.html', prediction = model_prediction)
@app.route("/home")
def home():
    # return the homepage
    return
render_template("index.html")
@app.route("/Image_Processing")
def image():
    # return the homepage
    return
render_template("image.html")
@app.route("/Model")
def model():
    # return the homepage
    return
render_template("model.html")
if __name___== '__main__':
    app.run()
```

• Implementation

• Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- 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 also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviours. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Modules Used in

Project Tensor Flow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance multi- dimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object-oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

7. SYSTEM TESTING

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 tests. Each test type addresses a specific testing requirement.

7.1. TYPES OF TESTS

Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce 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.

Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional Test

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.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

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.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach:

Field testing will be performed manually and functional tests will be written in detail.

Test objectives:

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

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.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

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.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

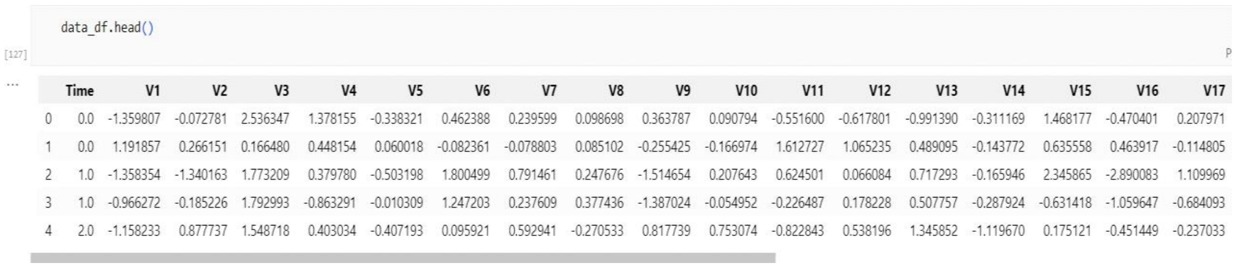
7.2 Test Cases:

S.no	Test Case	Excepted Result	Result	Remarks(IF Fails)
1.	Data uploading	Initially data need to be uploaded.	Pass	Executed successfully
2.	Data Splitting	After Uploading the Data, it should divide into train and test data.	Pass	Algorithm implemented
3.	Giving Input	Input should be Uploaded.	Pass	Input uploaded successfully
4.	Fraud Detection	After giving values from v1 to v28 it will find out whether it is detecting fraud or not.	Pass	If Data in the Dataset is wrong then it gives negative results.
5.	Data uploading		Fail	Cannot be executed

		Initially data need to be Uploaded.		
6.	Fraud Detection	fraud detection need to be done.	Fail	fraud is not detected

Table no 7.2 Test Cases

Test Case 1:

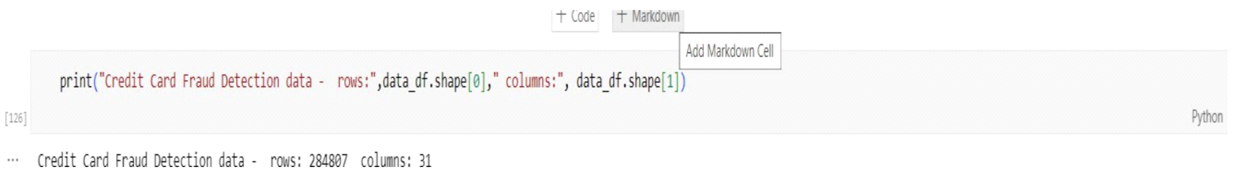


```
data_df.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	0.090794	-0.551600	-0.617801	-0.991390	-0.311169	1.468177	-0.470401	0.207971
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	-0.166974	1.612727	1.065235	0.489095	-0.143772	0.635558	0.463917	-0.114805
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	0.207643	0.624501	0.066084	0.717293	-0.165946	2.345865	-2.890083	1.109969
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	-0.054952	-0.226487	0.178228	0.507757	-0.287924	-0.631418	-1.059647	-0.684093
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	0.753074	-0.822843	0.538196	1.345852	-1.119670	0.175121	-0.451449	-0.237033

Figure 7.2.2: Test Case 1

Test Case 2:

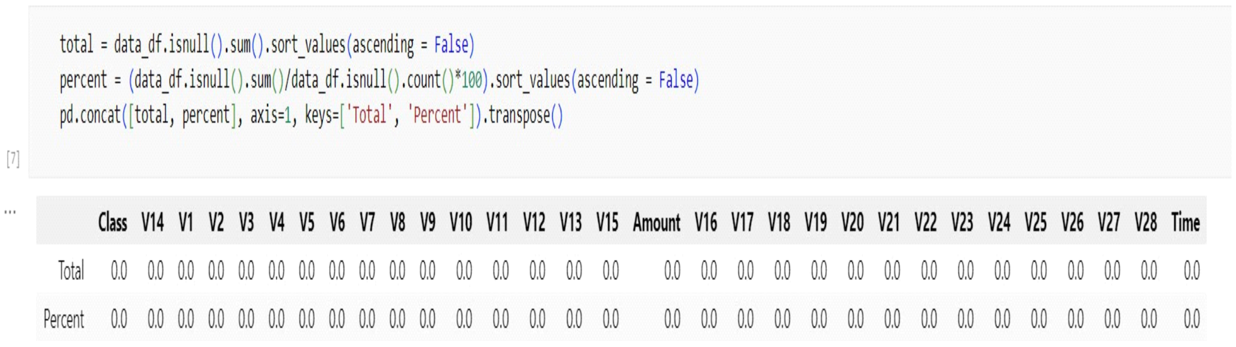


```
print("Credit Card Fraud Detection data -  rows:",data_df.shape[0]," columns:", data_df.shape[1])
```

```
Credit Card Fraud Detection data -  rows: 284807  columns: 31
```

Figure 7.2.3: Test Case 2

Test Case 3:

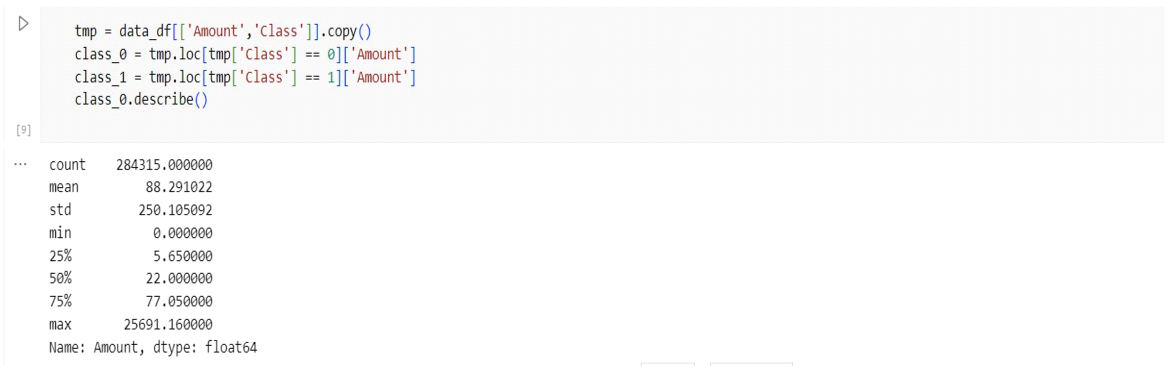


```
total = data_df.isnull().sum().sort_values(ascending = False)
percent = (data_df.isnull().sum()/data_df.isnull().count()*100).sort_values(ascending = False)
pd.concat([total, percent], axis=1, keys=['Total', 'Percent']).transpose()
```

	Class	V14	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V15	Amount	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25	V26	V27	V28	Time
Total	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Percent	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 7.2.4: Test Case 3

Test Case 4:

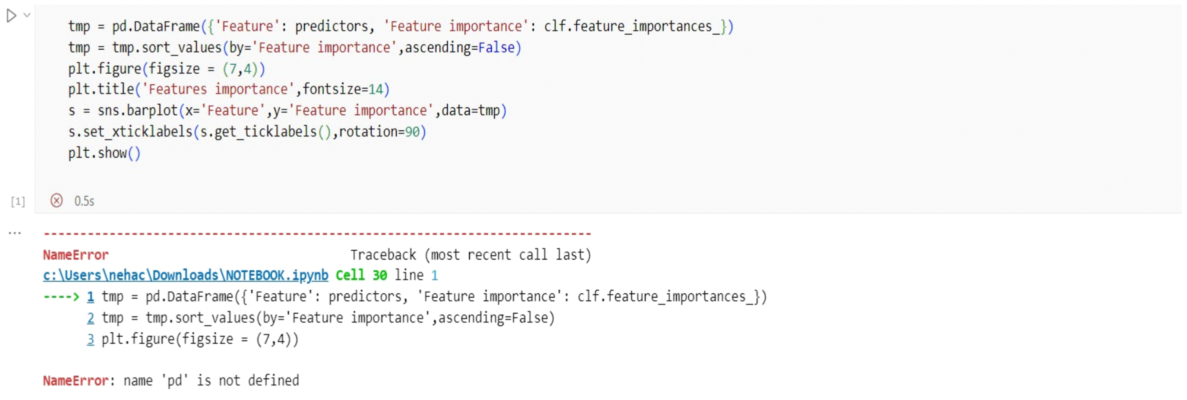


```
tmp = data_df[['Amount','Class']].copy()
class_0 = tmp.loc[tmp['Class'] == 0]['Amount']
class_1 = tmp.loc[tmp['Class'] == 1]['Amount']
class_0.describe()
```

```
count    284315.000000
mean         88.291022
std         250.105092
min           0.000000
25%           5.650000
50%          22.000000
75%          77.050000
max       25691.160000
Name: Amount, dtype: float64
```

Figure 7.2.5: Test Case 4

Test Case 5:



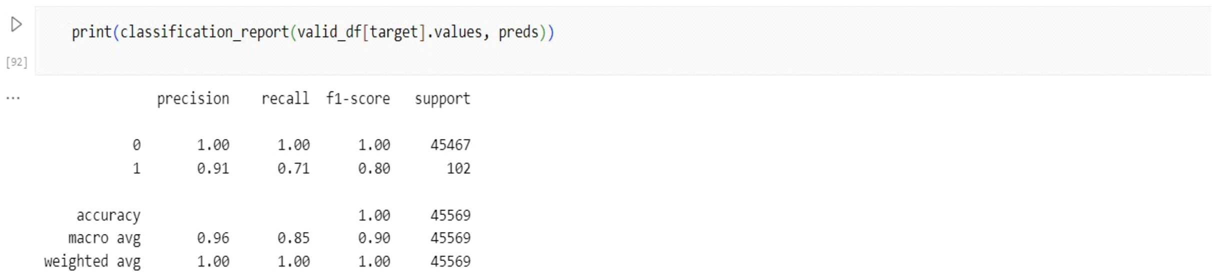
```
tmp = pd.DataFrame({'Feature': predictors, 'Feature importance': clf.feature_importances_})
tmp = tmp.sort_values(by='Feature importance',ascending=False)
plt.figure(figsize = (7,4))
plt.title('Features importance',fontsize=14)
s = sns.barplot(x='Feature',y='Feature importance',data=tmp)
s.set_xticklabels(s.get_ticklabels(),rotation=90)
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
c:\Users\nehac\Downloads\NOTEBOOK.ipynb Cell 30 line 1
----> 1 tmp = pd.DataFrame({'Feature': predictors, 'Feature importance': clf.feature_importances_})
      2 tmp = tmp.sort_values(by='Feature importance',ascending=False)
      3 plt.figure(figsize = (7,4))

NameError: name 'pd' is not defined
```

Figure 7.2.6: Test Case 5

Test Case 6:



```
print(classification_report(valid_df[target].values, preds))
```

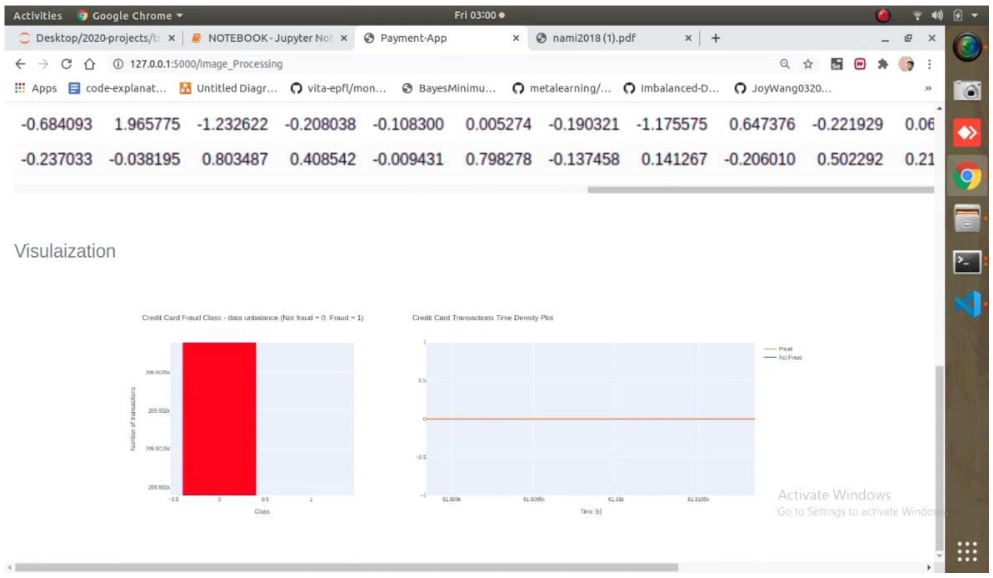
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     45467
           1       0.91      0.71      0.80       102

    accuracy                           1.00     45569
   macro avg       0.96      0.85      0.90     45569
weighted avg       1.00      1.00      1.00     45569
```

Figure 7.2.7: Test Case 6

8.OUTPUT SCREENS


Activities
Google Chrome
Fri 03:00
Payment-App
127.0.0.1:5000/Image_Processing
Visulaization
Credit Card Fraud Class - data unbalance (Not fraud = 0, Fraud = 1)
Credit Card Transactions Time Density Plot
Activate Windows
Go to Settings to activate Windows

```
-0.684093  1.965775  -1.232622  -0.208038  -0.108300  0.005274  -0.190321  -1.175575  0.647376  -0.221929  0.06
-0.237033  -0.038195  0.803487  0.408542  -0.009431  0.798278  -0.137458  0.141267  -0.206010  0.502292  0.21
```

Figure 8.1: Visualization

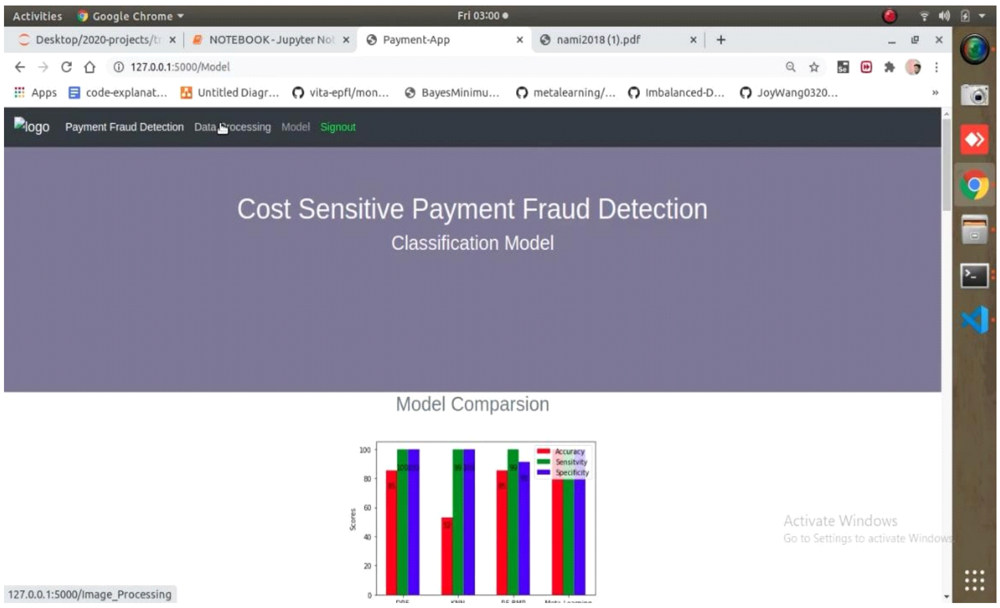
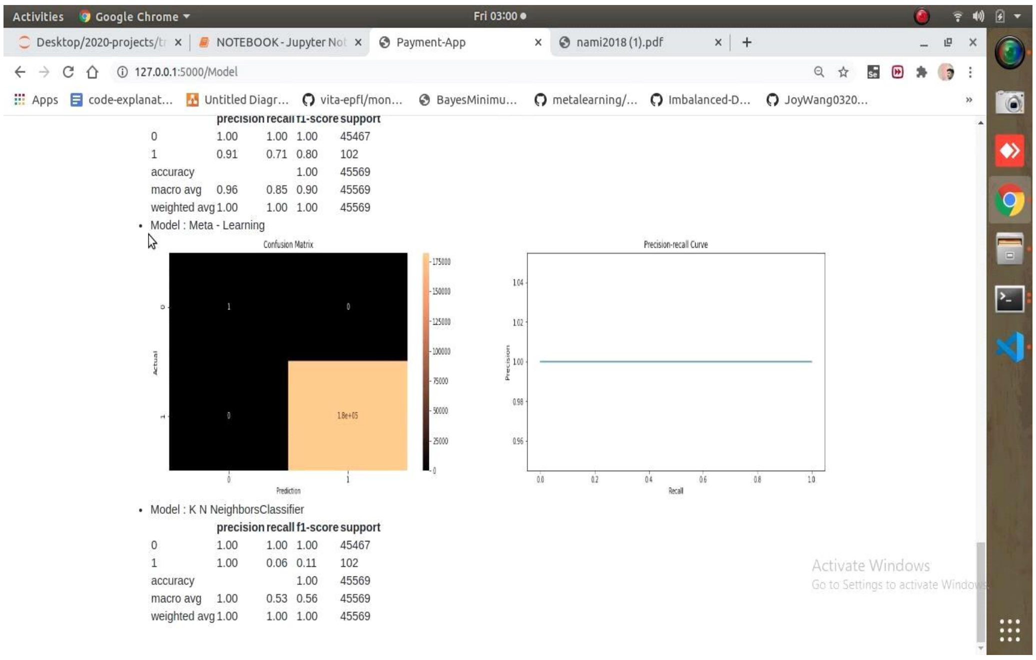


Figure 8.2: Cost Sensitive Payment fraud detection Classification model



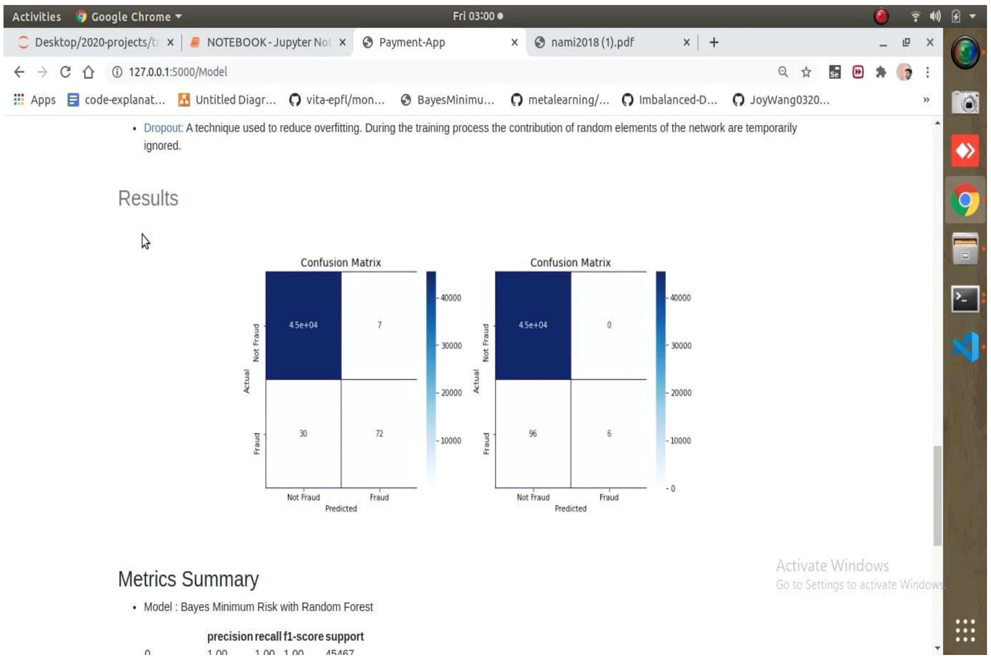
```
                precision recall f1-score support
0               1.00      1.00   1.00     45467
1               0.91      0.71   0.80     102
accuracy                         1.00     45569
macro avg       0.96      0.85   0.90     45569
weighted avg    1.00      1.00   1.00     45569
```

• Model : Meta - Learning

• Model : K N NeighborsClassifier

```
                precision recall f1-score support
0               1.00      1.00   1.00     45467
1               1.00      0.06   0.11     102
accuracy                         1.00     45569
macro avg       1.00      0.53   0.56     45569
weighted avg    1.00      1.00   1.00     45569
```

Figure 8.3: Precision recall Curve



• Dropout: A technique used to reduce overfitting. During the training process the contribution of random elements of the network are temporarily ignored.

Results

Metrics Summary

• Model : Bayes Minimum Risk with Random Forest

```
                precision recall f1-score support
0               1.00      1.00   1.00     45467
```

Figure 8.4: Confusion Matrix

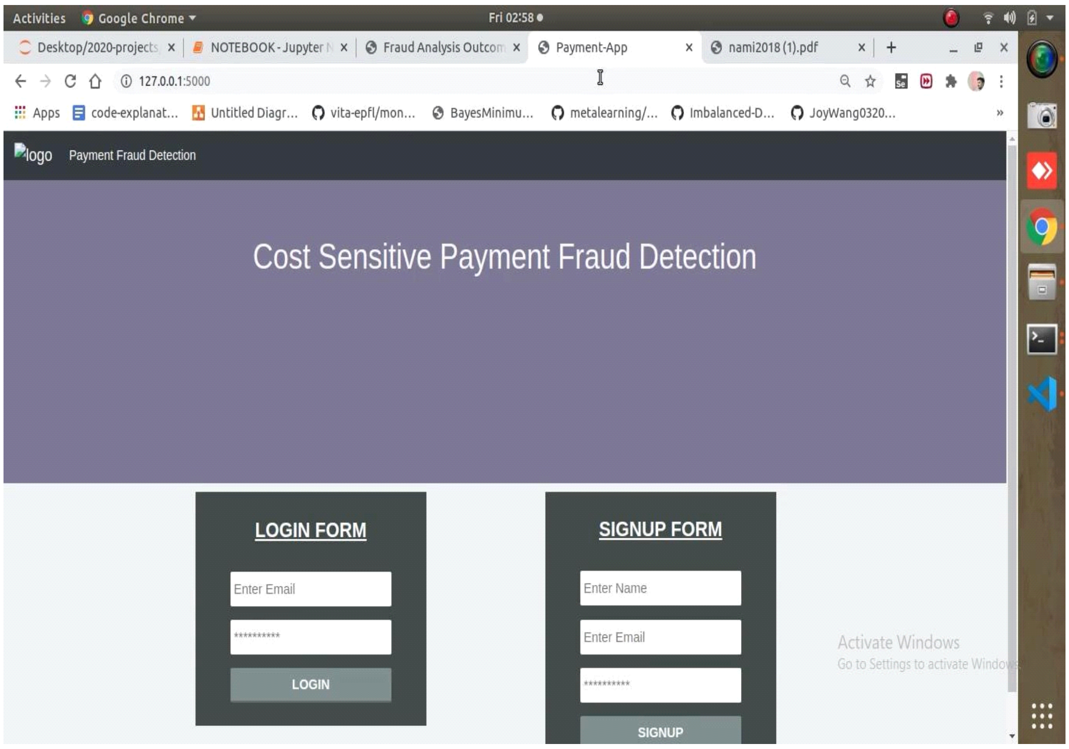
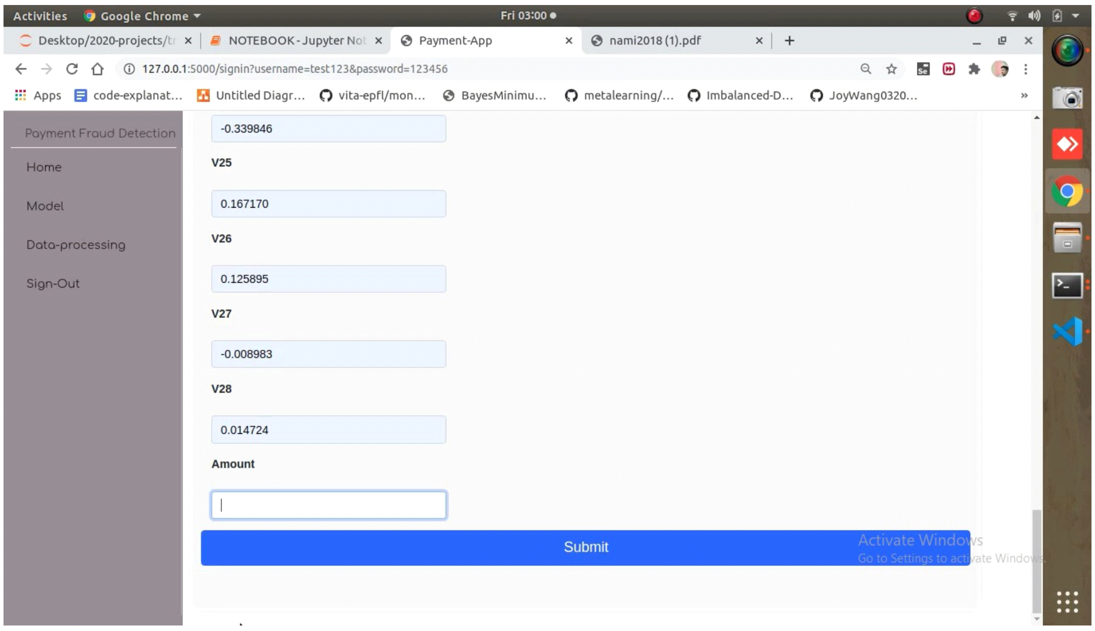
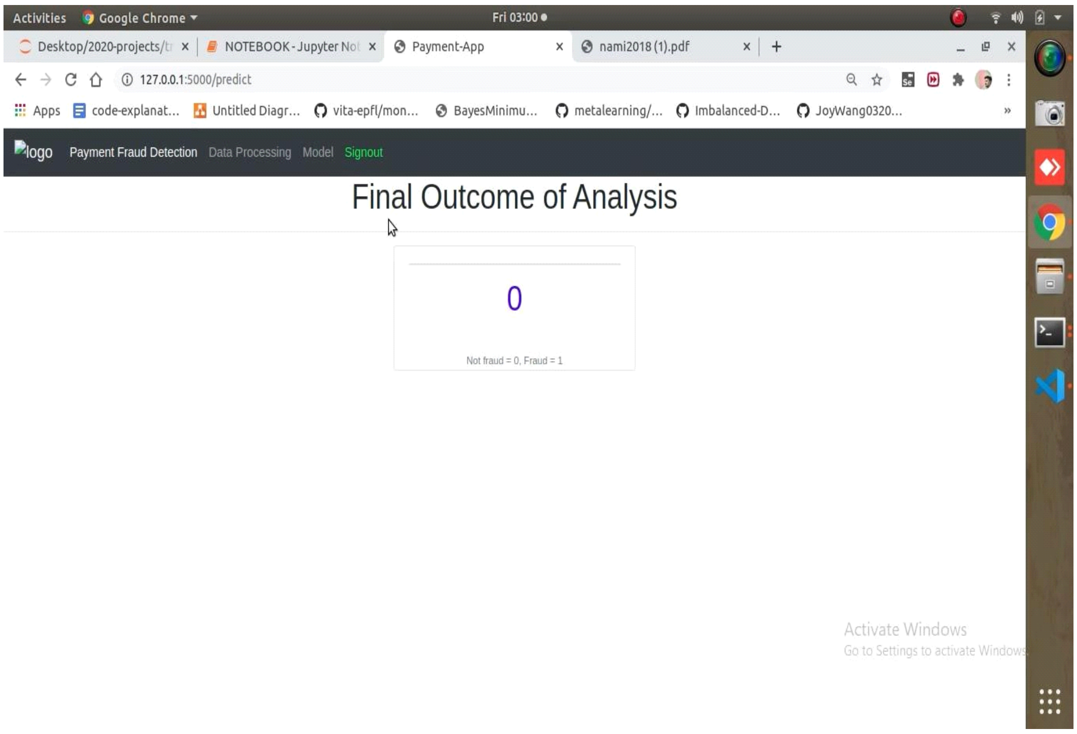


Figure 8.5: Login form & Signup form



-0.339846
V25
0.167170
V26
0.125895
V27
-0.008983
V28
0.014724
Amount

Figure 8.6: Data set values



Final Outcome of Analysis

0

Not fraud = 0, Fraud = 1

Figure 8.7: Final Outcome Analysis

9. CONCLUSION

A cost-sensitive random forest based ensemble learning technique has been proposed for effective detection of credit card fraud. The imbalanced nature of credit card data has been investigated in this work. A misclassification ratio-based cost-function has been integrated into the error-formulation of the generated sub-tress in RF-bagging. The cost-function facilitates to determine the predictive ability of sub-tree, so that the sub-tree with highest predictive ability can have maximum weight age. The final outcome of the test-set is determined based on the outcome, yielded by the maximally weighted tree. The experimental results achieved have manifested the efficiency of the proposed model in effective handling of imbalanced cases in case of credit card fraud detection. The proposed model has not been validated for high-dimensional datasets. The proposed model can be extended by integrating with different data cleaning techniques such as sampling or feature selection (or extraction) algorithm to be implemented in high-dimensional datasets. The imbalanced nature of the detection methods and its effects over the performance has not been investigated in this paper. A deep exploration of the issue in accordance with the computational performance of the credit card fraud detection techniques is another scope of future study.

10. FUTURE ENHANCEMENTS

To enhance the interpretability and transparency of the system, investing in advanced explain ability methods will be pivotal for gaining regulatory compliance and building trust. Furthermore, the incorporation of unsupervised learning techniques, such as autoencoders and isolation forests, can provide the capability to detect novel, previously unseen fraud patterns without the need for labelled data.

Finally, the development of hybrid models, which amalgamate different machine learning and statistical techniques, will allow for the leveraging of multiple approaches to different facets of fraud detection, ultimately creating a more resilient and adaptable payment fraud detection system.

11. REFERENCES

[1] L. Breiman, Random forests. Machine Learning, vol. 45, issue 1, pp: 5-32, 2021.

[2] G. Singh, R. Gupta, A. Rastogi, M. D. S. Chandel, and A. Riyaz, A Machine Learning Approach for Detection of Fraud based on SVM, International Journal of Scientific Engineering and Technology, vol.1, issue 3, pp. 194-198, 2021.

[3] A. C. Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, Improving credit card fraud detection with calibrated probabilities. In Proceedings of the 2020 SIAM International Conference on Data Mining, pp. 677-685, 2021.

[4] J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare, Credit card fraud detection using Machine Learning Techniques: A Comparative Analysis, pp: 1-9, 2020.

[5] F. N. Ogwueleka, Data Mining Application in Credit Card Fraud Detection System, Journal of Engineering Science and Technology, vol. 6, issue 3, pp. 311 – 322, 2020.

[6] P. L., Meshram, and P. Bhanarkar, Credit and ATM Card Fraud Detection Using Genetic Approach, International Journal of Engineering Research & Technology (IJERT), vol. 1, issue 10,
pp. 1 – 5, 2019.

[7] K. R. Seeja, and M. Zareapoor, FraudMiner: A Novel Credit Card Fraud Detection Model Based on Frequent Itemset Mining, The Scientific World Journal, Hindawi Publishing Corporation, vol. 2018,
Article ID 252797, pp. 1 – 10, 2017,

[8] V, Patil, U. K. Lilhore, A Survey on Different Data Mining & Machine Learning Methods for Credit Card Fraud Detection, International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Vol. 3, Issue 5, pp:320-325, 2017.

[9] M. Zareappor, P. Shamsolmoali, Application of Credit card Fraud Detection: Based onBagging Ensemble Classifier, International Conference on Intelliegent Computing, Communication, & Convergence (ICCC-2016), In Procedia Computer Science, Vol. 48, pp: 679 – 685, 2017.

[10] A. Y. Ng, and M. I. Jordan, On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. Advances in neural information processing systems, Vol. 2, pp: 841-848, 2016.

[11] S. Stolfo, D. W. Fan, W. Lee, A. Prodromidis, and P. Chan, Credit card fraud detection using meta-learning: Issues and initial results. In AAAI-97 Workshop on Fraud Detection and Risk Management., 2015.

[12] S. Patil, H. Somavanshi, J. Gaikwad, A. Deshmane, and R. Badgujar,2014 Credit Card Fraud Detection Using Decision Tree Induction Algorithm, International Journal of Computer Science and Mobile Computing (IJCSMC), vol.4, issue 4, pp. 92-95, 2015. ISSN: 2320-088X

[13] Elkan, C. (2014). "The Foundations of Cost-Sensitive Learning. In International Joint Conference on Artificial Intelligence (IJCAI).

[14] Chen, L., Cao, Y., and Luo, X. (2013). Dynamic Random Forest. In International Conference on Machine Learning (ICML).

[15] Bhattacharyya, S., Jha, D., Tharakunnel, K., and Westland, J.C. (2012). A Novel Ensemble Framework for Credit Card Fraud Detection.

[16] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). "The WEKA Data Mining Software: An Update." ACM SIGKDD Explorations Newsletter.

[17] Batista, G. E., Prati, R. C., and Monard, M. C. (2004). "A Study of the Behavior of Several Methods for Balancing Machine Learning Datasets." In ACM SIGKDD Explorations Newsletter.

[18] Breiman, L. (2001). "Random Forests." Machine Learning, 45(1), 5-32.

[19] Japkowicz, N. (2000). "The Class Imbalance Problem: Significance and Strategies." In Proceedings of the 2000 International Conference on Artificial Intelligence.

[20] K. RamaKalyani, and D. UmaDevi, Fraud Detection of Credit Card Payment System by Genetic
Algorithm, International Journal of Scientific & Engineering Research vol 3,issue 2021.