CREDIT CARD FRAUD DETECTION

USING MACHINE LEARNING SCHEMA

A PROJECT REPORT

Submitted by

G V CHAITHANYA 210420104060

Y BALA SAI VIVEK 210420104190

in partial fulfillment for the award of

the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE ENGINEERING

CHENNAI INSTITUTE OF TECHNOLOGY

CHENNAI 600 069

ANNA UNIVERSITY CHENNAI 600 025

APRIL 2024

BONAFIDE CERTIFICATE

Certified that this project report titled *"***CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING SCHEMA***"*, is the bonafide work of "**Y.BALA SAI VIVEK" AND "G.V.CHAITHANYA"**

who carried out the project work under my supervision.

SIGNATURE OF SUPERVISOR

Dr. GOWRI, M.E., Ph.D.,
Professor and Head
Dept of Computer Science Engineering
Chennai Institute of Technology
Kundrathur – 600069

SIGNATURE OF HEAD OF THE DEPARTMENT

Dr. S.PAVITHRA, M.E., Ph.D.,
Professor and Head
Dept of Computer Sciene Engineering
Chennai Institute of Technology
Kundrathur – 600069

Submitted for the ANNA UNIVERSITY examination held on_____ at Chennai Institute of Technology, Kundrathur.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We thank our beloved **Chairman Shri. P. SRIRAM** and all the trust members of Chennai Institute of Technology at this high time for providing us with a plethora of facilities to complete my project successfully.

We take the privilege to express my thanks to our Principal **Dr A. RAMESH, M.E., Ph.D.,** who has been a bastion of moral strength and a source of incessant encouragement to us.

We express our sincere thanks to **Dr. S.PAVITHRA, M.E., Ph.D.,** Head of the Department, **Dr. S.PAVITHRA, M.E., Ph.D.,** Project Guide and **Dr. GOWRI, M.E., Ph.D.,** Project Supervisor. We take immense pleasure to have them also as our mentor providing valuable suggestions, excellent guidance, and constant support all through the course of our project.

We also thank the teaching and non-teaching staff members of the Information Technology Department and all our fellow students who stood with us to complete our project successfully.

Finally, we extend our deep gratitude to our beloved family members for their moral coordination, encouragement, and financial support to carry out this project.

ABSTRACT

A credit card is issued by a bank or financial services company that allows cardholders to borrow funds with which to pay for goods and services with merchants that accept cards for payment. Nowadays as everything is made cyber so there is a chance of misuse of cards and the account holder can lose the money so it is vital that credit card companies are able to identify fraudulent credit card transactions so that customers are not charged for items that they did not purchase. This type of problems can be solved through data science by applying machine learning techniques. It deals with modelling of the dataset using machine learning with Credit Card Fraud Detection. In machine learning the main key is the data so modelling the past credit card transactions with the data of the ones that turned out to be fraud. The built model is then used to recognize whether a new transaction is fraudulent or not. The objective is to classify whether the fraud had happened or not. The first step involves analyzing and pre-processing data and then applying machine learning algorithm on the credit card dataset and find the parameters of the algorithm and calculate their performance metrics.

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Domain overview

1.1 Data Science

Data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from structured and unstructured data and apply knowledge and actionable insights from data across a broad range of application domains.

The term "data science" has been traced back to 1974, when Peter Naur proposed it as an alternative name for computer science. In 1996, the International Federation of Classification Societies became the first conference to specificallyfeature data science as a topic. However, the definition was still influx.

The term “data science” was first coined in 2008 by D.J. Patil, and Jeff Hammerbacher, the pioneer leads of data and analytics efforts at LinkedIn and Facebook. In less than a decade, it has become one of the hottest and most trending professions in the market.

Data science is the field of study that combines domain expertise, programming skills, and knowledge of mathematics and statistics to extract meaningful insights from data.

Data science can be defined as a blend of mathematics, business acumen, tools, all of which help us in finding out the hidden insights or patterns from rawdata which can be of major use in the formation

Data Scientist:

Data scientists examine which questions need answering and where to find the related data. They have business acumen and analytical skills as well as the ability to mine, clean, and present data. Businesses use data scientists to source, manage, and analyze large amounts of unstructured data.

Required Skills for a Data Scientist:

Programming: Python, SQL, Scala, Java, R, MATLAB.

Machine Learning: Natural Language Processing, Classification, Clustering.

Big data platforms: MongoDB, Oracle, Microsoft Azure, Cloudera.

1.2 ARTIFICIAL INTELLIGENCE

Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions. Theterm may also be applied to any machine that exhibits traits associated witha human mind such as learning and problem- solving. Artificial intelligence (AI) isintelligence demonstrated by machines, as opposed to the natural intelligence displayed by humans or animals. Leading AI textbooks define the field as the study of "intelligent agents" any system that perceives its environment and takesactions that maximize its chance of achievingits goals. Some popular accounts use theterm "artificial intelligence" to describe machines that mimic "cognitive" functions that humans associate with the humanmind, such as "learning" and "problem solving", however this definition is rejected by major AI researchers.

1.3 Natural Language Processing (NLP):

Natural language processing (NLP) allows machines to read and understand human language. A sufficiently powerful natural language processingsystem would enable natural- language user interfaces and the acquisition of knowledge directly from human-written sources, such as news

wire texts. Some straightforward applications of natural language processing include information retrieval, text mining, question answering and machine translation. Many currentapproaches use word syntactic representations of text. "Keyword spotting" strategies for search are popular and scalable; a search query for "dog" might only match documents with the literal word "dog" and miss a document with the word"poodle". "Lexical affinity" strategies use the occurrence of words such as"accident" to assess the sentiment of a document. Modern statistical NLP approaches combine all strategies as well as others, and often achieve acceptableaccuracy at the page or paragraph level. Beyond semantic NLP, the ultimate goalof "narrative" NLP is to embody a full understanding of commonsense reasoning. By2019, transformer-based deep learning architectures generate coherent text.

1.4 MACHINE LEARNING

Machine learning is to predict the future from past data. Machine learning (ML) is a type of artificial intelligence (AI) that provides computers with the ability tolearn without being explicitly programmed. Machine learning focuses on the development of Computer Programs that can change when exposed to new data and the basics of Machine Learning, implementation of a simple machine learning algorithm using python. Process of training and prediction involves useof specialized algorithms. It feed the training data to an algorithm,

and the algorithm uses this training data to give predictions on a new test data. Machine learning can be roughly separated in to three categories. There are supervised learning, unsupervised learning and reinforcement learning. Supervised learning program is both given the input data and the corresponding labeling to learn datahas to be labeled by a human being beforehand. Unsupervised learning is no labels. It provided to the learning algorithm. This algorithm has to figure out theclustering of the input data. Finally, Reinforcement learning dynamically interacts with its environment and it receives positive or negative feedback to improve its performance. Data scientists use many different kinds of machine learning algorithms todiscover patterns in python that lead to actionable insights. At a high level, thesedifferent algorithms can be classified into two groups based on the way they "learn" about data to make predictions: supervised and unsupervised learning. Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function from input variables(X)to discrete output variables(y). In machine learning and statistics, classification isa supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation.

Process Of Machine Learning

Supervised Machine Learning **is the** majority of practical machine learning usessupervised learning. Supervised learning is where have input variables (X)

and algorithms include **logistic regression**, **multi-class classification**, **Decision Trees** and **support vector machines etc**. Supervised learning requires that the data used to train the algorithm is already labeled with correct answers. Supervised learning problems can be further grouped into **Classification** problem

Preparing the Dataset :

This dataset contains 3075 records of features , which were then classified into 2 classes:

1.Fraud

2.Not fraud.

They proposed a method and named it as Information-Utilization- Method INUM it was first designed and the accuracy and convergence of an information vector generated by INUM are analyzed. Two D-vectors (i.e., featuresubsets) a and b, where Ai is the ith feature in a data set, are dissimilar in decision space, butcorrespond to the same O-vector y in objective space. Assume that only a is provided to decision-makers, but a becomes inapplicable due to an accident or other reasons (e.g., difficulty to extract from the data set). Then, decisionmakersare in trouble. On the other hand, if all two feature subsets are provided to them,they can have other choices to serve their best interest. In other words, obtainingmore equivalent D-vectors in the decision space can provide more chances for decision-makers to ensure that their interests are best served. Therefore, it is of great significance and importance to solve MMOPs with a good Pareto front approximation and also the largest number of D- vectors given each O-vector.

Disadvantages:

1. They had proposed a mathematical model and machine learning algorithms not used
2. Class Imbalance problem was not addressed and the proper measure were not taken.

CHAPTER 2
LITERATURE SURVEY

General

A literature review is a body of text that aims to review the critical points of current knowledge on and/or methodological approaches to a particular topic.It is secondary sources and discuss published information in a particular subject area and sometimes information in a particular subject area within a certain timeperiod. Its ultimate goal is to bring the reader up to date with current literature ona topic and forms the basis for another goal, such as future research that may be needed in the area and precedes a research proposal and may be just a simple summary of sources. Usually, it has an organizational pattern and combines both summary and synthesis.

A summary is a recap of important information about the source, but a synthesis is a re-organization, reshuffling of information. It might give a new interpretation of old material or combine new with old interpretations or it might trace the intellectual progression of the field, including major debates. Depending on the situation, the literature review may evaluate the sources andadvise the reader on the most pertinent or relevant of them

Review of Literature Survey

1. **Title :** Credit card Fraud Detection Web Application using Streamlit and Machine Learning.
   Author: Vipul Jain, Kavitha H.
   Year : 2022

This research work used a real-world credit card dataset. To detect the fraud transaction within this dataset three machine learning algorithms are used (i.e.

Random Forest, Logistic regression, and AdaBoost) and compared the machine learning algorithms based on their
Accuracy and Mathews Correlation Coefficient (MCC) Score. In these three algorithms, the Random Forest Algorithm achieved the best Accuracy and MCC score. The Streamlit framework is used to create the machine learning web application.

2.Title : An Efficient Techniques for Fraudulent detection in Credit Card Dataset: A Comprehensive study

Author: Akanksha Bansal and Hitendra Garg

Year : 2021

Now a day, credit card transaction is one the famous mode for financial transaction. Increasing trends of financial transactions through credit cards also invite fraud activities that involve the loss of billions of dollars globally. It is alsobeen observed that fraudulent transactions have increased by 35% from 2018. Ahuge amount of transaction data is available to analyze the fraud detection activities that require analysis of behavior/abnormalities in the transaction datasetto detect and ignore the undesirable action of the suspected person. The proposedpaper lists a compressive summary of various techniques for the classification offraud transactions from the various datasets to alert the user for such transactions.In the last decades, online transactions are growing rapidly and the most common tool for financial transactions. The increasing growth of online transactions also increases threats. Therefore, in keeping in mind the security issue, nature, an anomaly in the credit card transaction, the proposed work represents the summaryof various strategies applied to identify the abnormal transaction in the dataset ofcredit card transaction datasets. This dataset contains a mix of normal and fraud transactions; this proposed work classifies and summarizes the various classification methods to classify the transactions using

various Machine Learning-based classifiers. The efficiency of the method depends on the dataset and classifier used. The proposed summary will be beneficial to the banker, creditcard user, and researcher to analyze to prevent credit card frauds. The future scopeof this credit card fraud detection is to explore the things in each and every associations and banks to live safe and happily life. The data must be getting the best results.

3. **Title :** Credit Card Fraud Detection and Prevention using Machine Learning

Author: S. Abinayaa, H. Sangeetha, R. A. Karthikeyan, K. Saran Sriram, D. Piyush

Year : 2020

This research focused mainly on detecting credit card fraud in real world. We must collect the credit card data sets initially for qualified data set. Then provide queries on the user's credit card to test the data set. After random forest algorithm classification method using the already evaluated data set and providing current data set[1]. Finally, the accuracy of the results data is optimised.Then the processing of a number of attributes will be implemented.The techniques efficiency is measured based on accuracy, flexibility, and specificity, precision. The results obtained with the use of the Random Forest Algorithm have proved much more effectively

4. **Title :** A Research on Credit Card Fraudulent Detection System

Author: Devika S P, Nisarga K S, Gagana P Rao, Chandini S B, Rajkumar N

Year : 2019

Nowadays credit card is more popular among the private and publicemployees. By using the credit card, the users purchase the consumable durable products in online, also transferring the amount from one account to other. The fraudster is

detecting the details of the behavior user transaction and doing the illegal activities with the card

by phishing, Trojan virus, etc. The fraudulent maythreaten the users on their sensitive information.

In this paper, we have discussedvarious methods of detecting and controlling the fraudulent activities. This will be helpful to improve the security for card transaction in future. Credit card fraudulent activities which are faced by the people is one of the major issues. Due to these fraudulent activities, many credit card users are losing their money and their sensitive information. In this paper, we have discussed the different fraudulent detection and controlling techniques in credit card and also it will be helpful to improve the security from the fraudsters in future.

5. **Title :** A Review On Credit Card Fraud Detection Using Machine Learning

Author: Suresh K Shirgave, Chetan J. Awati, Rashmi More, Sonam S. Patil

Year : 2019

In recent years credit card fraud has become one of the growing problem. A largefinancial loss has greatly affected individual person using credit card and also themerchants and banks. Machine learning is considered as one of the most successful technique to identify the fraud. This paper reviews different fraud detection techniques using machine learning and compare them using performance measure like accuracy, precision and specificity. The paper also proposes a FDS which uses supervised Random Forest algorithm. With this proposed system the accuracy of detecting fraud in credit card is increased. Further, the proposed system use learning to rank approach to rank the alert and also effectively addresses the problem concept drift in fraud detection. This paperhas reviewed various machine learning algorithm detect fraud in credit

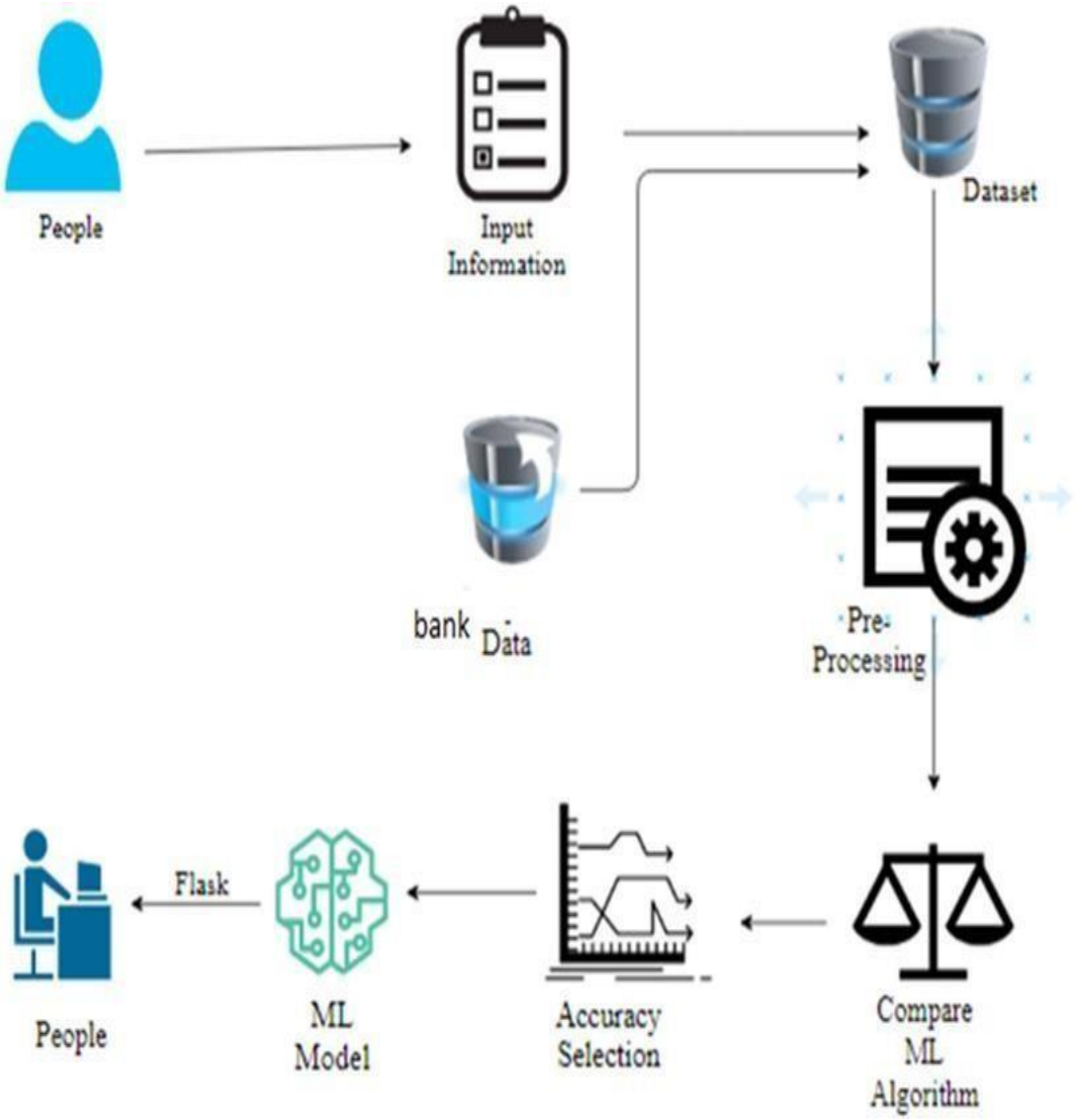
card transaction. The performances of all this techniques are examined based on accuracy, precision and specificity metrics. We have selected supervised learningtechnique Random Forest to classify the alert as fraudulent or authorized. This classifier will be trained using feedback and delayed supervised sample.

Next it will aggregate each probability to detect based on priority. The suggested methodwill be able to solve the class imbalance and concept drift problem. Future work will include applying semi-supervised learning methods for classification of alertin FDS

6. **Title :** Credit Card Fraud Detection using Machine Learning: A Systematic Literature Review
   Author: Harish Paruchuri
   Year : 2017

Companies want to give more and more facilities to their customers. One of thesefacilities is the online mode of buying goods. The customers now can buy the required goods online but this is also an opportunity for criminals to do frauds. The criminals can theft the information purchases until the cardholder contacts the bank to block the card. This paper different algorithms of machine learning that are used for detecting thiskind of transaction. The research shows the CCF is the major issue of financial sector that is increasing with the passage of time. This is an opportunity for criminals to theft the information or cards of other persons to make online transactions. The most popular techniques that are used to theft credit card information are phishing and Trojan. So a fraud detection system is needed to detect such activities.

CHAPTER 3
SYSTEM DESIGN AND IMPLEMENTATION



- Obtain a comprehensive dataset from various sources such as financial institutions, online repositories, or data providers.

- Ensure the dataset contains a diverse set of transactions, including both fraudulent and legitimate ones.

- Include attributes such as transaction amount, merchant, location (IP address or GPS coordinates), time of day, type of transaction (online, in-person), and any additional metadata available.

- Consider factors like imbalanced classes (fraudulent vs. legitimate transactions) and data quality during dataset acquisition.

- Handle missing values by imputation techniques such as mean, median, or using predictive models.

- Detect and deal with outliers which may skew the model's performance.

- Normalize numerical features to bring them to a similar scale, preventing certain features from dominating others during model training.

- Encode categorical features using techniques like one-hot encoding or label encoding to represent them numerically.

- Perform feature engineering to create new features or transform existing ones that might improve the model's performance. For example, derive features like transaction frequency, average transaction amount, or time since the last transaction.

- Utilize techniques such as correlation analysis to identify redundant or highly correlated features that may not contribute significantly to the model's predictive power.

- Employ feature importance methods such as RandomForest's feature importance or SHAP values to select the most relevant features for the model.

- Experiment with various machine learning algorithms suitable for classification tasks, considering factors like interpretability, scalability, and performance.

- Evaluate algorithms like Logistic Regression, Decision Trees, Random Forests, Gradient Boosting Machines (e.g., XGBoost, LightGBM), and Neural Networks.

- Compare their performance metrics such as accuracy, precision, recall, and F1-score to determine the most suitable algorithm for the task.

- Split the dataset into training, validation, and test sets to evaluate the model's performance accurately.

- Train the selected models using the training data and fine-tune their hyperparameters using techniques like grid search or random search.

- Utilize techniques like cross-validation to assess the model's robustness and generalization performance.

- Evaluate the trained models on the validation set using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, ROC-AUC, and PR-AUC.

- Visualize the model's performance using tools like confusion matrices, ROC curves, and precision-recall curves to gain insights into its behavior across different thresholds.

- Choose the model with the best overall performance for deployment based on the evaluation metrics.

- Deploy the selected model into a production environment, ensuring scalability, reliability, and low-latency inference.

- Implement an API for real-time scoring of new transactions, allowing seamless integration with existing banking systems or fraud detection pipelines.

- Utilize containerization technologies like Docker for packaging the model and Kubernetes for orchestration to manage deployments efficiently.

- Set up monitoring systems to track the model's performance in real-time, detecting any degradation or drift in its effectiveness.

- Implement mechanisms for retraining the model periodically using updated data to adapt to evolving fraud patterns and maintain its effectiveness over time.

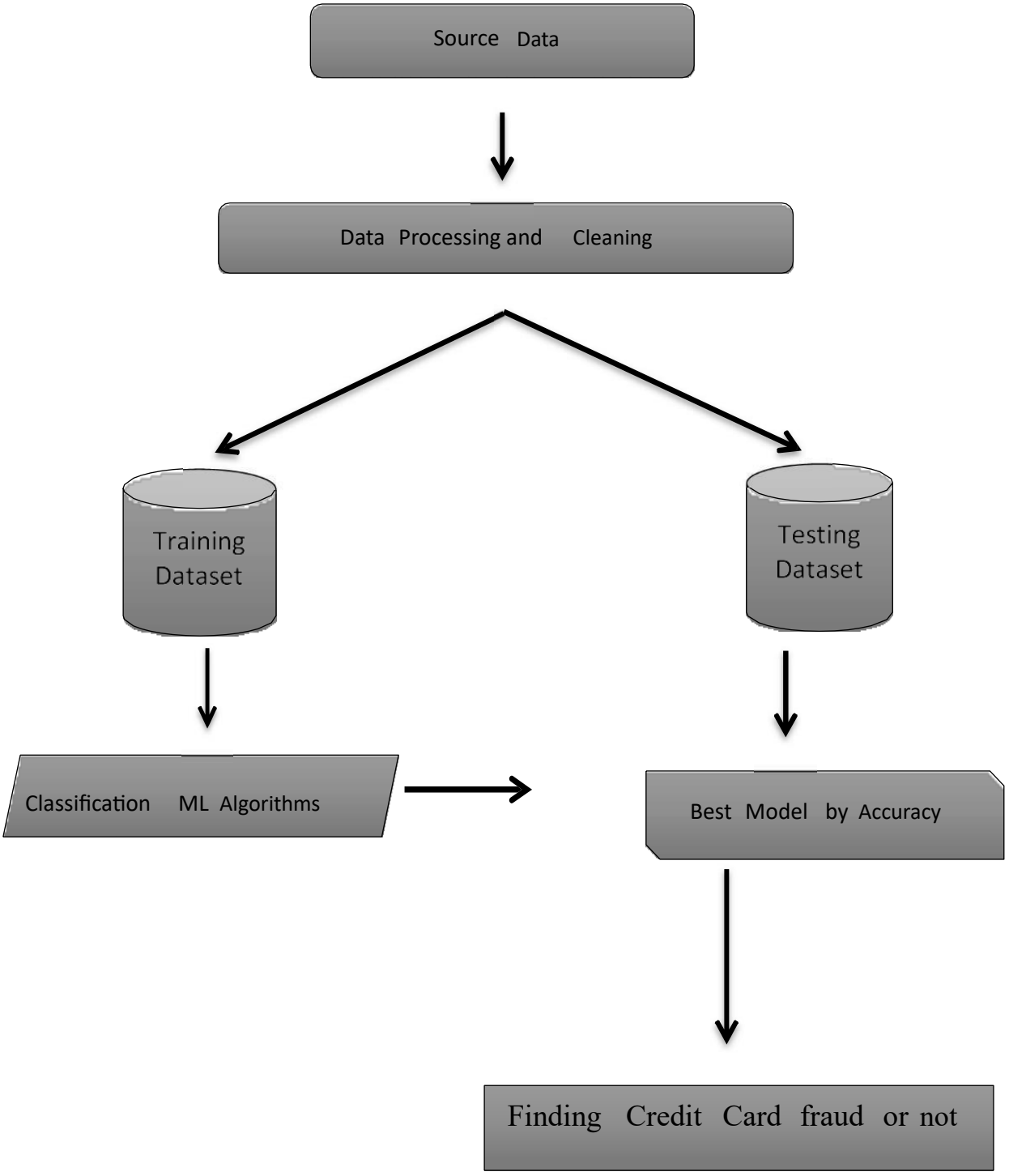
- Utilize techniques like online learning or incremental training to incorporate new data seamlessly without retraining the entire model from scratch.

- Integrate the fraud detection system with an alert mechanism to notify relevant stakeholders (e.g., customers, fraud analysts, financial institutions) in case of suspicious transactions.

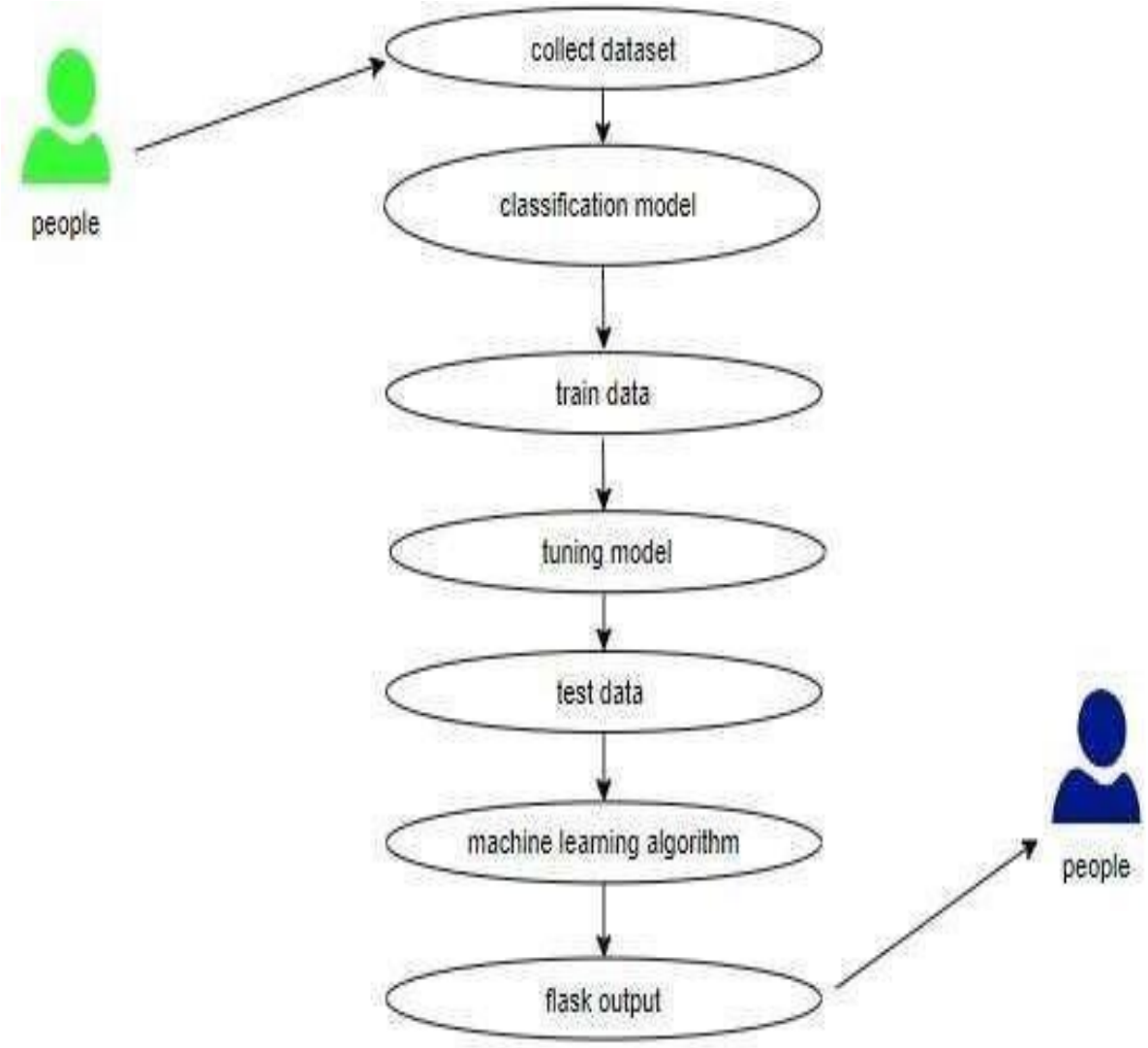
- Set dynamic thresholds based on the model's confidence score or risk tolerance to trigger alerts while minimizing false positives and negatives.

CHAPTER 4
METHODOLOGY

4.1 Block diagram

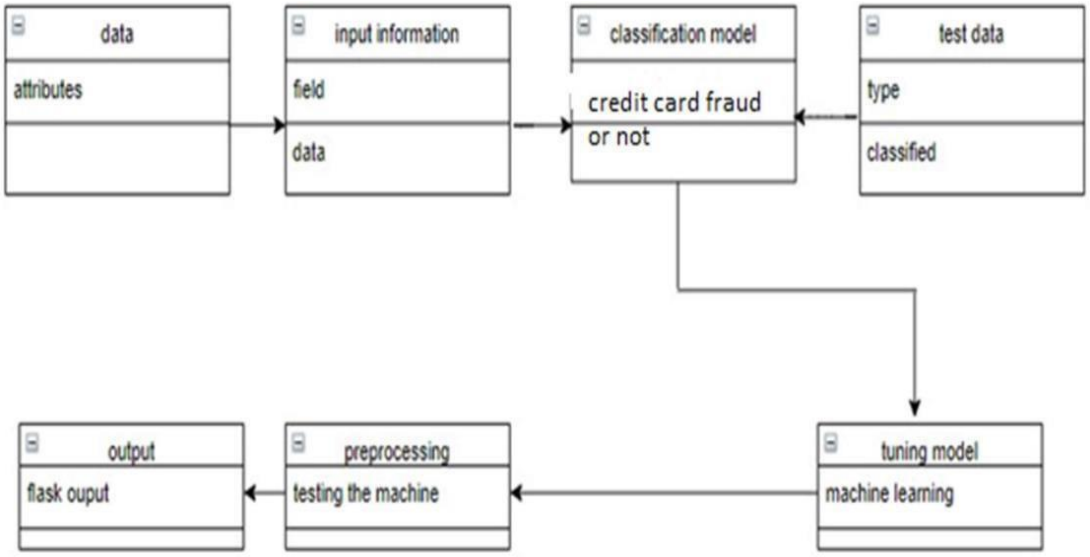


4.2 Use Case Diagram



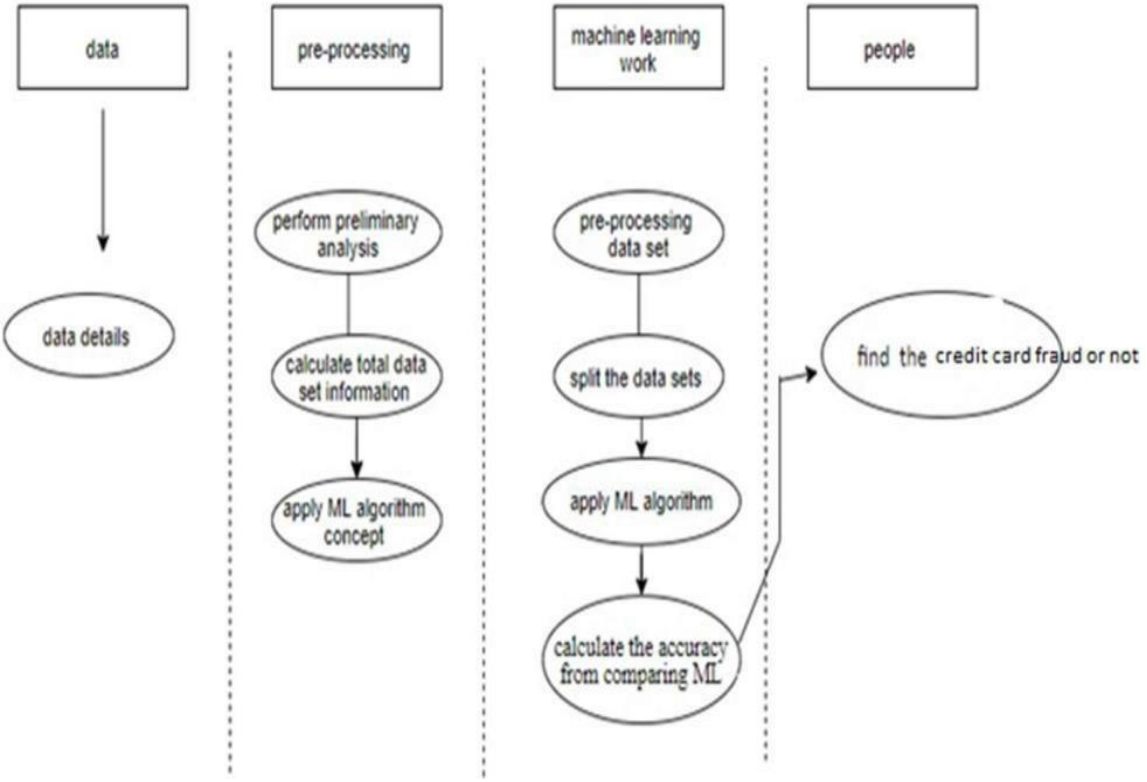
Use case diagrams are considered for high level requirement analysis of asystem. So when the requirements of a system are analyzed the functionalities arecaptured in use cases. So, it can say that uses cases are nothing but the system functionalities written in an organized manner.

4.3 CLASS DIAGRAM:



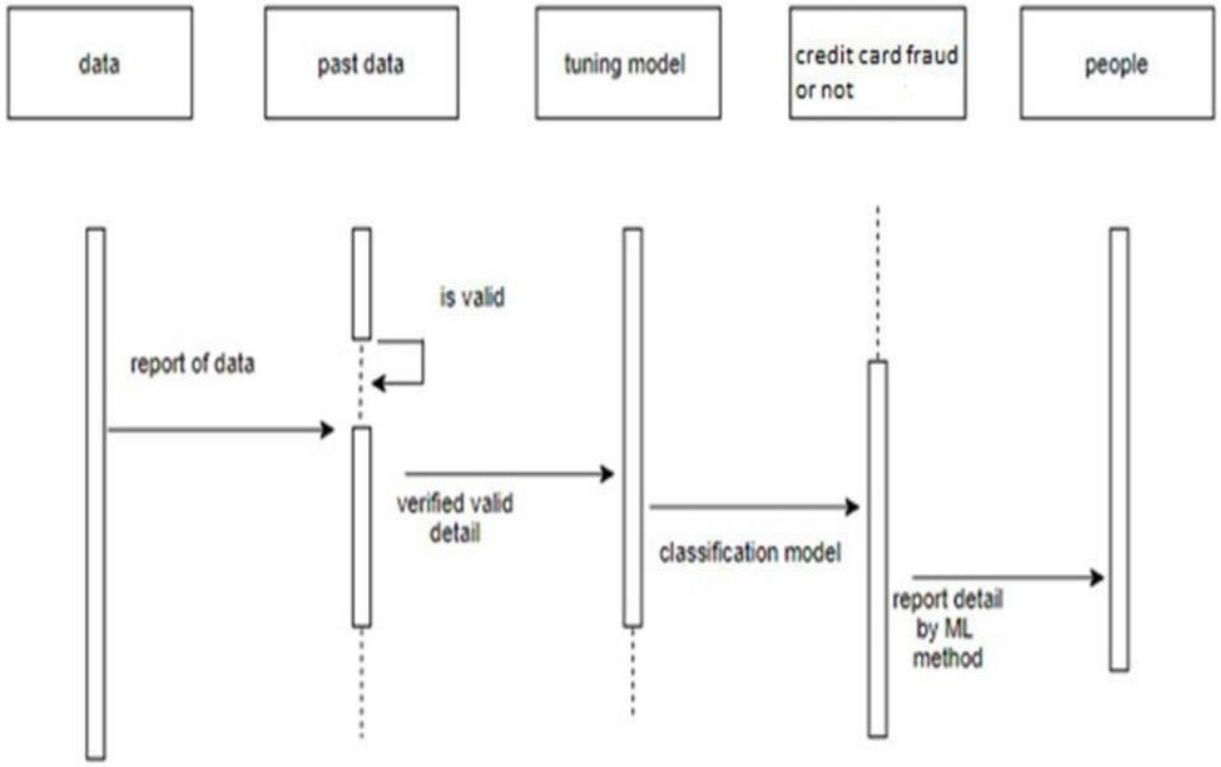
Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. So a collection ofclass diagrams represent the whole system. The name of the class diagram shouldbe meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance Responsibility (attributes andmethods) of each class should be clearly identified for each class minimum number of properties should be specified and because, unnecessary properties will make the diagram complicated. Use notes whenever required to describe some aspect of the diagram and at the end of the drawing it should be understandable to the developer/coder. Finally, before making the final version,the diagram should be drawn on plain paper and rework as many times as possibleto make it correct.

4.4 ACTIVITY DIAGRAM:



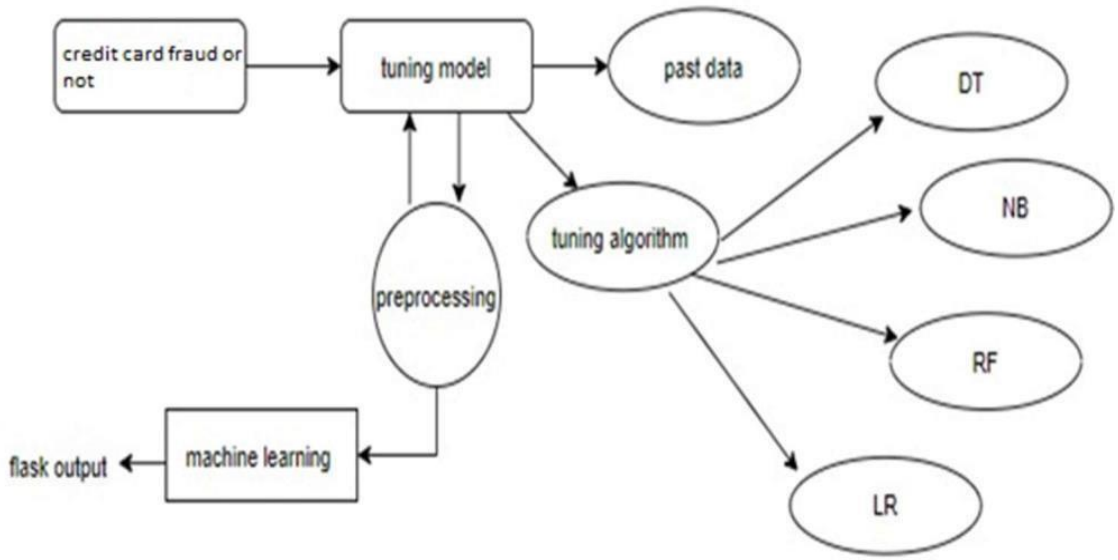
Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineeringtechniques. The only missing thing in activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagramis some time considered as the flow chart. Although the diagrams looks like a flow chart but it is not. It shows different flow like parallel, branched, concurrentand single.

4.5 SEQUENCE DIAGRAM:



Sequence diagrams model the flow of logic within your system in a visualmanner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modelling, which focuses on identifying the behaviour within your system. Other dynamic modelling techniques include activity diagramming, communication diagramming, timingdiagramming, and interaction overview diagramming. Sequence diagrams, along with class diagrams and physical data models are in my opinion the mostimportant design-level models for modern business application development.

4.6 ENTITY RELATION DIAGRAM(ERD):



An entity relationship diagram (ERD), also known as an entity relationshipmodel, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within thatsystem. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database. Entity relationship diagrams provide a visual starting point for database design that canalso be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an ERD can still serve as areferral point, should any debugging or business process re-engineering beneeded later.

4.7 ALGORITHM AND TECHNIQUES

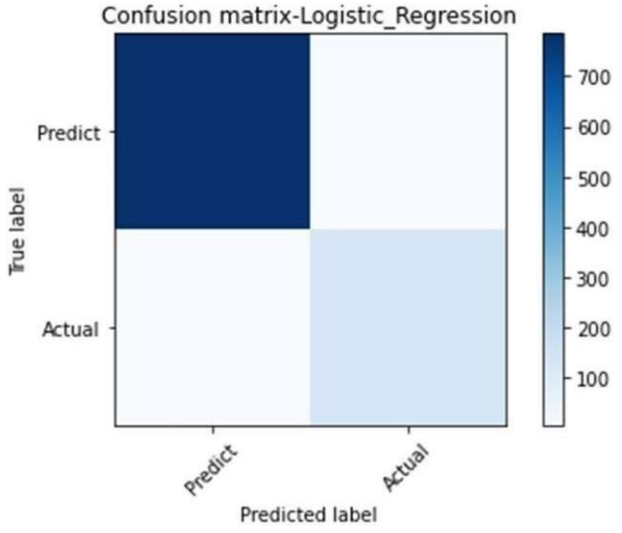
It is a statistical method for analysing a data set in which there are one or more independent variables that determine an outcome. The outcome is measuredwith a dichotomous variable (in which there are only two possible outcomes). Thegoal of logistic regression is to find the best fitting model to describe therelationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent (predictor or explanatory) variables. Logistic regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable.

In other words, the logistic regression model predicts P(Y=1) as a function of X. Logistic regression Assumptions:

- Binary logistic regression requires the dependent variable to be binary.
- For a binary regression, the factor level 1 of the dependent variable shouldrepresent the desired outcome.
- The independent variables should be independent of each other. That is, themodel

```
Classification report of Logistic Regression Results:

              precision    recall  f1-score   support

           0       0.99      0.99      0.99       789
           1       0.97      0.96      0.97       134

    accuracy                           0.99       923
   macro avg       0.98      0.98      0.98       923
weighted avg       0.99      0.99      0.99       923


Confusion Matrix result of Logistic Regression is:
 [[785   4]
 [  5 129]]

Sensitivity :  0.9949302915082383

Specificity :  0.9626865671641791

Cross validation test results of accuracy:
[0.9902439  0.98211382 0.9804878  0.98211382 0.98699187]

Accuracy result of Logistic Regression is: 98.4390243902439
```

```
Confusion matrix-Logistic_Regression:
[[785   4]
 [  5 129]]
```



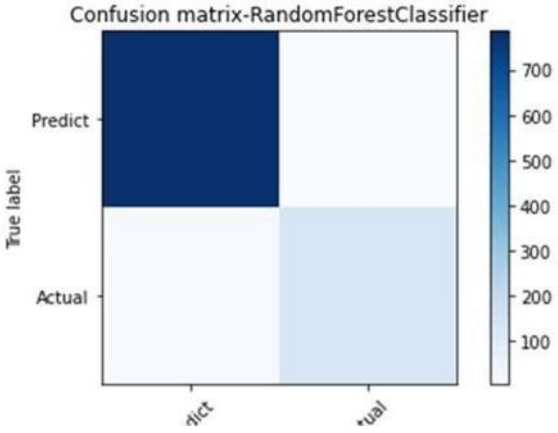
Random forests or random decision forests are an ensemble learningmethod for classification, regression and other tasks, that operate by constructinga multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random forest is a type of supervised machine learning algorithm based on ensemble learning. The random forest algorithm combines multiple algorithm of the same type i.e. multiple decision *trees,* resulting ina forest of trees, hence the name "Random Forest". The random forest algorithmcan be used for both regression and classification tasks.

The following are the basic steps involved in performing the random forest algorithm:

- Pick N random records from the dataset.
- Build a decision tree based on these N records.
- Choose the number of trees you want in your algorithm and repeat steps 1 and 2.

In case of a regression problem, for a new record, each tree in the forest predictsa value for Y (output). The final value can be calculated by taking the average ofall the values predicted by all the trees in forest. Finally, the new record is assigned to the category that wins the majority vote.

```
Confusion matrix-RandomForestClassifier:
[[786   3]
 [  7 127]]
```




```
Classification report of Random Forest Results:

              precision    recall  f1-score   support

           0       0.99      1.00      0.99       789
           1       0.98      0.95      0.96       134

    accuracy                           0.99       923
   macro avg       0.98      0.97      0.98       923
weighted avg       0.99      0.99      0.99       923


Confusion Matrix result of Random Forest Classifier is:
 [[786   3]
 [  7 127]]

Sensitivity :  0.9961977186311787

Specificity :  0.9477611940298507

Cross validation test results of accuracy:
[0.97723577 0.99349593 0.9804878  0.98536585 0.98211382]

Accuracy result of Random Forest Classifier is: 98.3739837398374
```

It is one of the most powerful and popular algorithm. Decision-tree algorithm falls under the category of supervised learning algorithm. It works for both continuous as well as categorical in output variables. Assumptions of Decision tree: At the beginning, we consider the whole training set as the root.

- Attributes are assumed to be categorical for information gain, attributes are assumed to be continuous. ➢ On the basis of attribute values records are distributed recursively.

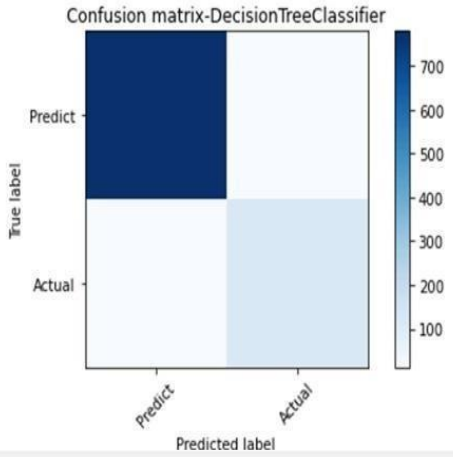
➢ We use statistical methods for ordering attributes as root or internal node.

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. A decision node has two or more branches and a leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data. Decision tree builds classification.

It utilizes an if-then rule set which is mutually exclusive and exhaustive for classification. The rules are learned sequentially using the training data one at a time. Each time a rule is learned, the tuples covered by the rules are removed.

This process is continued on the training set until meeting a termination condition. It is constructed in a top-down recursive divide-and-conquer manner. All the attributes should be categorical. Otherwise, they should be discretized in advance. Attributes in the top of the tree have more impact towards in the classification and they are identified using the information gain concept. A decision tree can be easily over-fitted generating too many branches and may reflect anomalies due to noise or outliers.

```
Confusion matrix-DecisionTreeClassifier:
[[779  10]
 [ 10 124]]
```




```
Classification report of Decision Tree Results:

              precision    recall  f1-score   support

           0       0.99      0.99      0.99       789
           1       0.93      0.93      0.93       134

    accuracy                           0.98       923
   macro avg       0.96      0.96      0.96       923
weighted avg       0.98      0.98      0.98       923


Confusion Matrix result of Decision Tree Classifier is:
 [[779  10]
 [ 10 124]]

Sensitivity :  0.9873257287705957

Specificity :  0.9253731343283582

Cross validation test results of accuracy:
[0.98373984 0.9804878  0.97723577 0.98373984 0.97398374]

Accuracy result of Decision Tree Classifier is: 97.98373983739837
```

It utilizes an if-then rule set which is mutually exclusive and exhaustive for classification. The rules are learned sequentially using the training data one at a time. Each time a rule is learned, the tuples covered by the rules are removed.

This process is continued on the training set until meeting a termination condition. It is constructed in a top-down recursive divide-and-conquer manner. All the attributes should be categorical. Otherwise, they should be discretized in advance. Attributes in the top of the tree have more impact towards in the classification and they are

identified using the information gain concept. A decision tree can be easily over-fitted generating too many branches and may reflect anomalies due to noise or outliers.

It utilizes an if-then rule set which is mutually exclusive and exhaustive for classification. The rules are learned sequentially using the training data one at a time. Each time a rule is learned, the tuples covered by the rules are removed.

This process is continued on the training set until meeting a termination condition. It is constructed in a top-down recursive divide-and-conquer manner. All the attributes should be categorical. Otherwise, they should be discretized in advance. Attributes in the top of the tree have more impact towards in the classification and they are identified using the information gain concept. A decision tree can be easily over-fitted generating too many branches and may reflect anomalies due to noise or outliers.

- The Naive Bayes algorithm is an intuitive method that uses the probabilities of each attribute belonging to each class to make a prediction.It is the supervised learning approach you would come up with if you wanted to model a predictive modeling problem probabilistically.
- The probability of a class value given a value of an attribute is called the conditional probability. By multiplying the conditional probabilities together for each attribute for a given class value, we have a probability of a datainstance belonging to that class. To make a prediction we can calculate probabilities of the instance belonging to each class and select the class value with the highest probability.
- Naive Bayes is a statistical classification technique based on Bayes Theorem. It is one of the simplest supervised learning algorithms.Naive Bayes classifier is the fast, accurate and reliable algorithm. Naive Bayes classifiers have high accuracy and speed on large datasets.

CHAPTER 5
IMPLEMENTATION

1. Software Requirements:

Operating System : Windows

Tool :Anaconda with Jupyter Notebook

2.Hardware Requirements:

Processor : Intel 3/5

Hard disk : minimum 80 GB

RAM : minimum 8 GB

5.1 ANACONDA NAVIGATOR

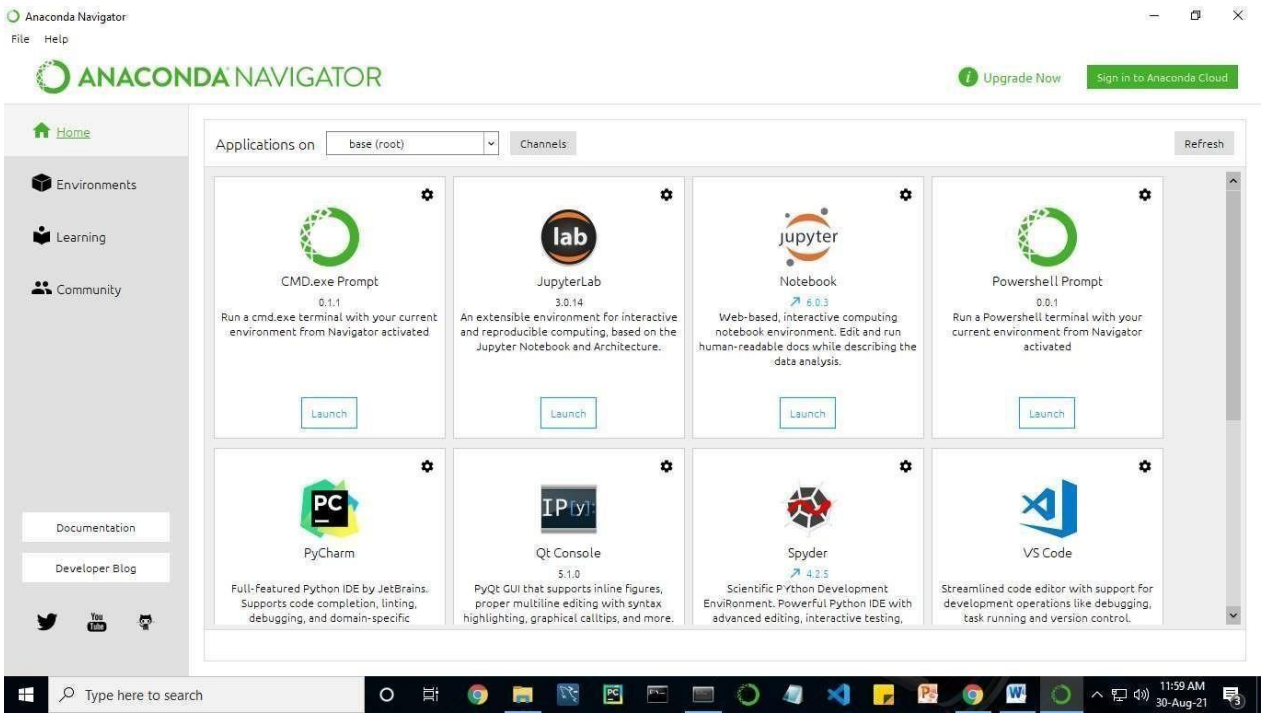
Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command- line commands. Navigator can search for packages on Anaconda.org or in a localAnaconda Repository.

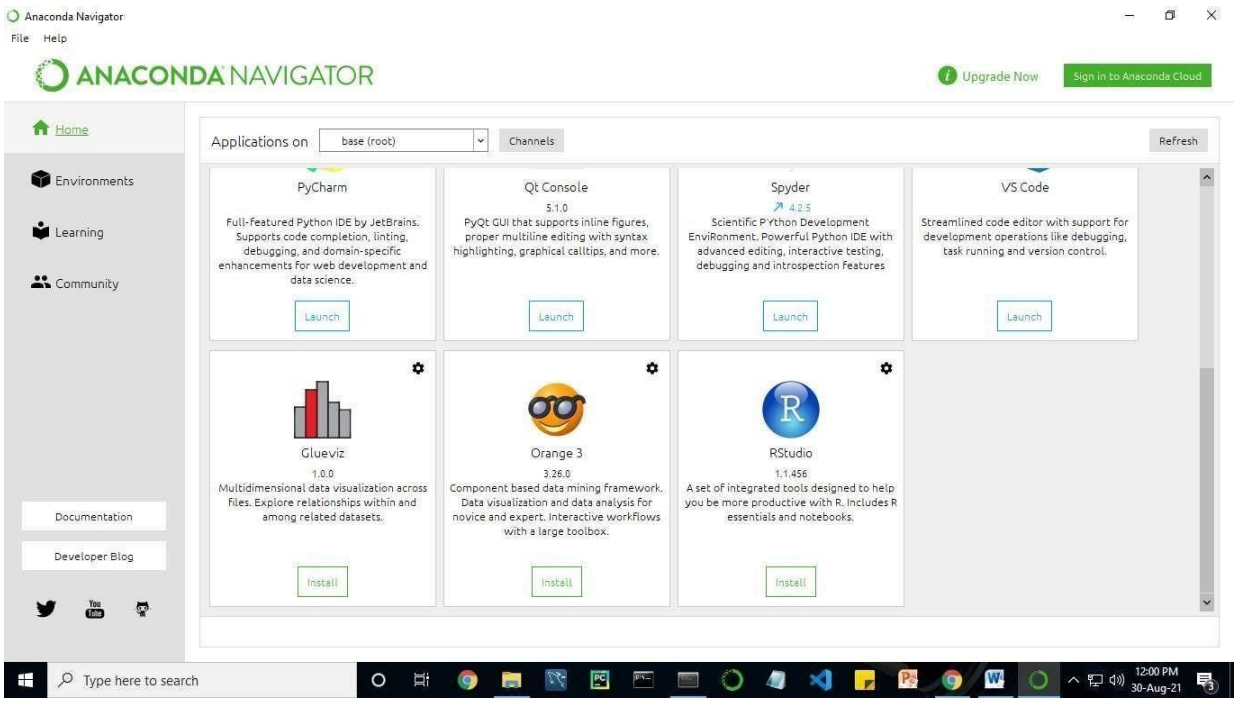
Anaconda. Now, if you are primarily doing data science work, Anaconda is also a great option. Anaconda is created by Continuum Analytics, and it isa Python distribution that comes preinstalled with lots of useful python libraries for data science.

Anaconda is a distribution of the Python and R programming languages forscientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- Spyder
- PyCharm
- VSCode
- Glueviz
- RStudio
- Anaconda Prompt (Windows only)
- Anaconda PowerShell (Windows only)





Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution.

Navigator allows you to launch common Python programs and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository.

Conda :

Conda is an open source, cross-platform, language-agnostic package manager and environment management system that installs, runs, and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language (e.g., R), including multi-language projects. The conda package and environment manager is included in all versions of Anaconda, Miniconda, and Anaconda Repository.

Anaconda is freely available, open source distribution of python and R programming languages which is used for scientific computations. If you are doing any machine learning or deep learning project then this is the best place

For you. It consists of many softwares which will help you to build your machine learning project and deep learning project. these softwares have great graphical user interface and these will make your work easy to do. you can also use it to run your python script. These are the software carried by anaconda navigator.

5.2 JUPYTER NOTEBOOK

This website acts as “meta” documentation for the Jupyter ecosystem. It has a collection of resources to navigate the tools and communities in this ecosystem, and to help you get started.

Project Jupyter is a project and community whose goal is to "develop open- source software, open-standards, and services for interactive computing across dozens of programming languages". It was spun off from IPython in 2014 by Fernando Perez.

Running the Jupyter Notebook

Launching Jupyter Notebook App: The Jupyter Notebook App can be launched by clicking on

the Jupyter Notebook icon installed by Anaconda in the start menu(Windows) or by typing in a terminal (cmd on Windows): “jupyter notebook”

This will launch a new browser window (or a new tab) showingthe Notebook

Dashboard, a sort of control panel that allows (among other things)to select which notebook to open.

Save notebooks: Modifications to the notebooks are automatically saved every few minutes.

To avoid modifying the original notebook, make a copy of the notebook document (menu file -> make a copy…) and save the modifications onthe copy.

Executing a notebook: Download the notebook you want to execute and put it inyour notebook

folder (or a sub-folder of it).

- Launch the jupyter notebook app
- In the Notebook Dashboard navigate to find the notebook: clicking on itsname will open it in a new browser tab.
- Click on the menu Help -> User Interface Tour for an overview of the Jupyter

  Notebook App user interface.
- You can run the notebook document step-by-step (one cell a time) by pressing shift + enter.
- You can run the whole notebook in a single step by clicking on themenu Cell -

  > Run All.
- To restart the kernel (i.e. the computational engine), click on the menu Kernel ->

  Restart. This can be useful to start over a computation fromscratch (e.g. variables are

  deleted, open files are closed, etc…).

Purpose: To support interactive data science and scientific computing across all programming languages.

File Extension: An **IPYNB** file is a notebook document created by Jupyter Notebook, an interactive computational environment that helps scientists manipulate and analyze data by using python.

Working Process

1. Download and Installing the Python anaconda platform.
2. Loading the dataset.
3. Summarizing the dataset

5.3 PYTHON

Introduction:

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its useof significant indentation. Its language constructs as well as its object- oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting.Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible. Python 2 was discontinued with version 2.7.18 in 2020.

History:

Python was conceived in the late 1980s by Guido van Rossum at CentrumWiskunde & Informatica (CWI) in the Netherlands as a successor to

ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's Benevolent Dictator For Life, a title the Python community bestowed upon him to reflect his long-term commitment in as term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a 5-member "Steering Council" to lead the project. As of 2021, the current members of this council are Barry Warsaw, BrettCannon, Carol Willing, Thomas Wouters, and Pablo Galindo Salgado.

Python 2.0 was released on 16 October 2000, with many major new features, including a cycle-detecting garbage collector and support for Unicode.

Python 3.0 was released on 3 December 2008. It was a major revision of the language that is not completely backward-compatible. Many of its major features were backported to Python 2.6.x and 2.7.x version series. Releases of Python 3 include the 2 to 3 utility, which automates (at least partially) the translation of Python 2 code to Python 3.

Design Philosophy & Feature

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming

including by meta-programming and metaobjects (magicmethods)). Many other paradigms are supported via extensions, including designby contract and logic programming.

Python uses dynamic typing and a combination of reference counting anda cycle- detecting garbage collector for memory management. It also features

dynamic name resolution (late binding), which binds method and variable namesduring program execution.

The language's core philosophy is summarized in the document The Zen ofPython (PEP 20), which includes aphorisms such as:

- Beautiful is better than ugly. - Explicit is better than implicit. Simple is better than complex.
- Complex is better than complicated.
- Readability counts.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible (with modules). This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications.

Python's developers aim to keep the language fun to use. This is reflectedin its name a tribute to the British comedy group Monty Python and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (a reference to a Monty Python sketch) instead of the standard foo and bar.

Syntax and Semantics :

Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages

use punctuation. Unlike many other languages, it does not use curly brackets todelimit blocks, and semicolons after statements are allowed but are rarely, if ever, used. It has fewer syntactic exceptions and special cases than C or Pascal.

Indentation :

Main article: Python syntax and semantics & Indentation

Python uses whitespace indentation, rather than curly brackets or keywords, to delimit blocks. An increase in indentation comes after certain statements; a decrease in indentation in signifies the end of the current block. Thus,the program's visual structure accurately represent the program's semantic structure.

Statements and control flow :

Python's statements include:

- The if statement, which conditionally executes a block of code, along with else and elif (a contraction of else-if).
- The for statement, which iterates over an iterable object, capturing each element to a local variable for use by the attached block.
- The while statement, which executes a block of code as long as its condition is true.
- The Try statement, which allows exceptions raised in its attached code blockto be caught and handled by except clauses; it also ensures that clean-up codein a finally block will always be run regardless of how the block exits.
- The raise statement, used to raise a specified exception or re-raise a caught exception.
- The with statement, which encloses a code block within a context manager (for example, acquiring a lock before the block of code is run and releasing

the lock afterwards, or opening a file and then closing it), allowing resource-acquisition-is-initialization (RAII) -

like behavior and replaces a common try/finally idiom.

 The continue statement, skips this iteration and continues with the next item.

The pass statement, which serves as a NOP. It is syntactically needed to createan empty code block. The return statement, used to return a value from a function.

The import statement, which is used to import modules whose functions or variables.

Python does not support tail call optimization or first-class continuations, and, according to Guido van Rossum, it never will.[80][81] However, better support for co-routine- like functionality is provided, by extending Python'sgenerators. Before 2.5, generators were lazy iterators; information was passed uni-directionally out of the generator.

Expressions :

Some Python expressions are similar to those found in languages such as Cand Java, while some are not:

 Addition, subtraction, and multiplication are the same, but the behavior of division differs. There are two types of divisions in Python. They are floor division (or integer division) // and floating-point/ division. Python also uses the ** operator for exponentiation..

 From Python 3.8, the syntax :=, called the 'walrus operator' was introduced. Itassigns values to variables as part of a larger.

Conditional expressions in Python are written as x if c else y (different in orderof operands from the c ? x : y operator common to many other languages).

- Python features sequence unpacking wherein multiple expressions, each evaluating to anything that can be assigned to (a variable, a writable property,etc.), are associated in an identical manner to that forming tuple literals and, as a whole, are put on the left-hand side of the equal sign in an assignment statement. The statement expects an iterable object on the right-hand side of the equal sign that produces the same number of values as the provided writable expressions when iterated through and will iterate through it, assigning each of the produced values to the corresponding expression on theleft.

Python has various kinds of string literals:

- o Strings delimited by single or double quote marks. Unlike in Unix shells, Perl and Perl-influenced languages, single quote marks and double quote marks function identically.

- Python has array index and array slicing expressions on lists, denoted as a[Key], a[start:stop] or a[start:stop:step]. Indexes are zero-based, and negative indexes are relative to the end. In Python, a distinction between expressions and statements is rigidly enforced, in contrast to languages

5.4 CODING

Pre-Processing

Module 1: Data validation and pre-processing technique

In [ ]

In [ ]:

df.columns

df.info() In [ ]:
Checking duplicate value from dataframe

#Checking for duplicate data df.duplicated() In [ ]:

#find sum of duplicate data In [ ]:
sum(df.duplicated())

#Checking sum of missing values In [ ]:

df.isnull().sum() df.isForeignTransaction. In [ ]:

unique() df.TransactionAmount.unique() df.

isHighRiskCountry.unique() df. In [ ]:

DailyChargebackAvgAmt.unique() p. In [ ]:

Categorical(df['isFradulent']).describe() In [ ]:

In [ ]:
p.Categorical(df['6_MonthAvgChbkAmt']).describe() p.

In [ ]:

In [ ]:

Categorical(df['AverageAmountTransactionDay']).describe() df.

In [ ]:

columns df['6_MonthChbkFreq'].value_counts()

In [ ]:

df['6_MonthAvgChbkAmt'].value_counts() df.

In [ ]:

corr()

After Pre-processing

In [ ]:

df.head()

In [ ]:

df.columns

In [ ]:

```
from sklearn.preprocessing import LabelEncoder var_mod = ['AverageAmountTransactionDay', 'TransactionAmount', 'Is_declined', 'TotalNumberOfDeclinesDay', 'isForeignTransaction', 'isHighRiskCountry', 'DailyChargebackAvgAmt', '6_MonthAvgChbkAmt', '6_MonthChbkFreq',
    'isFradulent']
le = LabelEncoder()  for i in var_mod:
  df[i] = le.fit_transform(df[i]).astype(int) df.
```

In [ ]:

head(10)

In [ ]:

df.isnull().sum() df.

In [ ]:

tail(10)

Visualization

```
import pandas as p
import matplotlib.pyplot as plt import seaborn as s
import numpy as n

import warnings warnings.filterwarnings('ignore')
```

```
data = p.read_csv("creditcard.csv")

del data['Merchant_id'] del
    data['Tr

ansactionDate']




    PropByVar(df, 'isFradulent')


    # Heatmap plot diagram fig, ax =
    plt.subplots(figsize=(15,7)) s.heatmap(df.
    corr(), ax=ax, annot=True)

plt.boxplot(df['AverageAmountTransactionDay'])
    plt.
    show()

import seaborn as s s.
    boxplot(df['AverageAmountTransactionDay'], color='m')

from sklearn.preprocessing import LabelEncoder var_mod
    =['AverageAmountTransactionDay', 'TransactionAmount',
    'Is_declined',
          'TotalNumberOfDeclinesDay',          'isForeignTransaction',
    'isHighRiskCountry',
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: In [ ]:

```
df.columns
```

In [ ]:

```
#Histogram  Plot  of  Age  distribution  df['TransactionAmount'].
hist(figsize=(7,6),        color='b',        alpha=0.7)        plt.
xlabel('TransactionAmount')    plt.ylabel('Is_declined')         plt.
title('Transaction Amount & Declines')
```

In [ ]:

```
 #Propagation by variable
def         PropByVar(df,
variable):
    dataframe_pie = df[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(8,8), autopct='%1.2f%%', fontsize
= 10) ax.set_title(variable + ' \n', fontsize =
    15)                                  return
    n.round(dataframe_pie/df.shape[0]*100,2)
PropByVar(df, 'isHighRiskCountry')
```

In [ ]:

```
#Propagation by variable
def        PropByVar(df,
variable):
    dataframe_pie = df[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(8,8), autopct='%1.2f%%', fontsize
= 12) ax.set_title(variable + ' \n', fontsize =
    15)                                  return
    n.round(dataframe_pie/df.shape[0]*100,2)
```

```
PropByVar(df, 'isFradulent')
```
In [ ]:

```
# Heatmap plot diagram fig, ax =
plt.subplots(figsize=(15,7)) s.heatmap(df.
corr(), ax=ax, annot=True)
```
In [ ]:

```
plt.boxplot(df['AverageAmountTransactionDay'])
plt.
show()
```

```
import seaborn as s s.
boxplot(df['AverageAmountTransactionDay'], color='m')
```
In [ ]:

```
from sklearn.preprocessing import LabelEncoder var_mod
=['AverageAmountTransactionDay', 'TransactionAmount',
'Is_declined',
       'TotalNumberOfDeclinesDay',          'isForeignTransaction',
'isHighskCountry',
```
In [ ]:

```
       'DailyChargebackAvgAmt', '6_MonthAvgChbkAmt', '6_MonthChbkFreq',
       'isFradulent']
```
In [ ]:

```
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)
fig, ax = plt.subplots(figsize=(16,8)) ax.
scatter(df['AverageAmountTransactionDay'],df['DailyChargebackAvgAmt'])  ax.
set_xlabel('AverageAmountTransactionDay') ax.
set_ylabel('DailyChargebackAvgAmt')
ax.set_title('Daily Transaction & Chargeback Amount')
plt.
show()
```
In [ ]:

```
df.columns
```
In [ ]:

```
plt.plot(df["TransactionAmount"], df["DailyChargebackAvgAmt"], color='g') plt.
xlabel('TransactionAmount')
plt.ylabel('DailyChargebackAvgAmt')
plt. title('Credit Card Transaction')
plt.show()
```

Splitting Train / Test

In [ ]:

```
    #preprocessing, split test and dataset, split response variable
    X = df.drop(labels='isFradulent', axis=1)
    #Response variable y = df.loc[:,'isFradulent']
```

In [ ]:

```
            ax.text(rect.get_x() + rect.get_width()/2,
                    height + 2,
                    str(height)+'%',          ha=
                    'center', va=
                    'bottom', fontsize
                    = 12)
        return dataframe_by_Group
```

In [ ]:

```
qul_No_qul_bar_plot(df,'TotalNumberOfDeclinesDay')
```

Logistic Regression Algorithm

In [ ]:

```
    #import library packages import
    pandas as p
    import matplotlib.pyplot as plt
    import  seaborn  as  s  import
    numpy as n

import  warnings  warnings.
    filterwarnings('ignore')


    #Load given dataset
    data = p.read_csv('creditcard.csv')

del     data['TransactionDate']
    del data['Merchant_id']

df=data.dropna()

from sklearn.preprocessing import LabelEncoder
```

```
    #We'll use a test size of 20%. We also stratify the split on the response
    variable, which is very important to do because there are so few
    fraudulent transactions.
    from sklearn.model_selection import train_test_split
    X_train,  X_test,  y_train,  y_test  =  train_test_split(X,  y,  test_size=0.20,
    random_state=
```

```
1,  stratify=y)  print("Number  of  training
dataset: ", len(X_train)) print("Number of test
dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))
```

In [ ]:

```
def qul_No_qul_bar_plot(df, bygroup):
    dataframe_by_Group = p.crosstab(df[bygroup], columns=df["isFradulent"],
normalize       =       'index')       dataframe_by_Group       =
    n.round((dataframe_by_Group   *   100),   decimals=2)   ax   =
    dataframe_by_Group.plot.bar(figsize=(15,7));       vals       =
    ax.get_yticks() ax.set_yticklabels(['{:3.0f}%'.
    format(x) for x in vals]);
    ax.set_xticklabels(dataframe_by_Group.index,rotation = 0, fontsize =
15); ax.set_title('Credit Card Transaction (%) (by
    ' +
dataframe_by_Group.index.name + ')\n', fontsize = 15) ax.
    set_xlabel(dataframe_by_Group.index.name, fontsize = 12) ax.
    set_ylabel('(%)', fontsize = 12)
    ax.legend(loc = 'upper left',bbox_to_anchor=(1.0,1.0), fontsize= 12)
    rects = ax.patches

    # Add Data Labels

    for rect in rects:
        height = rect.get_height()
```

In [ ]:

```
var_mod = ['Is_declined', 'isForeignTransaction', 'isHighRiskCountry',
'isFradulent']
le            =
LabelEncoder()
for     i     in
var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)



#preprocessing, split test and dataset, split response variable
X = df.drop(labels='isFradulent', axis=1)
#Response variable
```

In [ ]:

```
y = df.loc[:,'isFradulent']
'''We'll use a test size of 30%. We also stratify the split on the
response variable, which is very important to do because there are so
few fraudulent transactions'''
from        sklearn.model_selection        import
train_test_split
X_train,  X_test,  y_train,  y_test  =  train_test_split(X,  y,  test_size=0.30,
random_state=
1,  stratify=y)  print("Number  of  training
dataset: ", len(X_train)) print("Number of test
dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))
```

In [ ]:

```
    #According to the cross-validated MCC scores, the random forest is the

    best-performing model, so now let's evaluate its performance on the
    test set.     from    sklearn.metrics    import    confusion_matrix,
    classification_report,       matthews_corrcoef,    cohen_kappa_score,
    accuracy_score,  average_precision_score, roc_auc_score
```

In [ ]:

```
from      sklearn.metrics      import      accuracy_score,
    confusion_matrix   from   sklearn.linear_model   import
    LogisticRegression  from  sklearn.model_selection  import
    cross_val_score   logR=   LogisticRegression()   logR.
    fit(X_train,y_train)

    predictLR = logR.predict(X_test)
print("") print('Classification report of Logistic Regression
    Results:') print("")
    print(classification_report(y_test,predictLR))
print("") cm1=
    confusion_matrix(y_test,predictLR)  print('Confusion  Matrix
    result of Logistic Regression is:\n',cm1) print("")
    sensitivity1                              =
    cm1[0,0]/(cm1[0,0]+cm1[0,1])
    print('Sensitivity : ', sensitivity1 )
    print("")
    specificity1                              =
    cm1[1,1]/(cm1[1,0]+cm1[1,1])
    print('Specificity  :  ',  specificity1)
    print("")


    accuracy     =     cross_val_score(logR,     X,     y,
    scoring='accuracy')   print('Cross   validation   test
    results of accuracy:') print(accuracy)
    #get the mean of each fold print("") print("Accuracy result of Logistic Regression
    is:",accuracy.mean() * 100) LR=accuracy.
    mean() * 100
```

In [ ]:

```
    def graph():
        import  matplotlib.pyplot  as  plt  data=[LR]  alg="Logistic
        Regression"     plt.     figure(figsize=(5,5))     b=plt.
        bar(alg,data,color=("c"))  plt.title("Accuracy  comparison
        of Earth Quake",fontsize=15)
```

In [ ]:

In [ ]:
```
        plt.legend(b,data,fontsize=9)
graph()
```

In [ ]:
```
    TP = cm1[0][0]
    FP = cm1[1][0]
    FN = cm1[1][1]
    TN = cm1[0][1] print("True
    Positive :",TP) print("True
    Negative             :",TN)
    print("False       Positive
    :",FP)         print("False
    Negative  :",FN)  print("")
    TPR = TP/(TP+FN)
    TNR = TN/(TN+FP)
    FPR  =  FP/(FP+TN)  FNR  =  FN/(TP+FN)
    print("True   Positive   Rate   :",TPR)
    print("True   Negative   Rate   :",TNR)
    print("False   Positive   Rate   :",FPR)
    print("False   Negative   Rate   :",FNR)
    print("")   PPV  =  TP/(TP+FP)   NPV  =
    TN/(TN+FN)  print("Positive  Predictive
    Value :",PPV) print("Negative predictive
    value :",NPV)
```

In [ ]:
```
    def    plot_confusion_matrix(cm1,    title='Confusion
    matrix-   Logistic_Regression',   cmap=plt.cm.Blues):
    target_names=['Predict','Actual']
```

In [ ]:
```
        plt.imshow(cm1, interpolation='nearest', cmap=cmap) plt.
        title(title)   plt.colorbar()   tick_marks   =
        n.arange(len(target_names)) plt.
        xticks(tick_marks, target_names, rotation=45) plt.
        yticks(tick_marks, target_names) plt.
        tight_layout() plt.ylabel('True
        label') plt.
        xlabel('Predicted label')
```

In [ ]:
```
cm1=confusion_matrix(y_test,        predictLR)
    print('Confusion                  matrix-
    Logistic_Regression:')          print(cm1)
    plot_confusion_matrix(cm1)
```

Random Forest Algorithm

In [ ]:

```
#import library packages import
pandas as p
import matplotlib.pyplot as plt
import seaborn as s import
numpy as n
```

In [ ]:

```
import warnings warnings.
filterwarnings('ignore')
```

In [ ]:

```
data=p.read_csv('creditcard.csv')
del data['Merchant_id'] del
data['TransactionDate']
```

In [ ]:

```
df=data.dropna()
```

In [ ]:

```
from sklearn.preprocessing import LabelEncoder
var_mod = ['Is_declined','isForeignTransaction',
'isHighRiskCountry','isFradulent']
le = LabelEncoder() for i in
var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)
```

In [ ]:

In [ ]:

```
#preprocessing, split test and dataset, split response variable
X = df.drop(labels='isFradulent', axis=1)
#Response variable
y = df.loc[:,'isFradulent']


'''We'll use a test size of 30%. We also stratify the split on the
response variable, which is very important to do because there are so
few fraudulent transactions'''
from sklearn.model_selection import
train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30,
random_state=
1, stratify=y) print("Number of training
dataset: ", len(X_train)) print("Number of test
dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))
```

In [ ]:

```
#According to the cross-validated MCC scores, the random forest is
the best-performing model, so now let's evaluate its performance on
the test set. from sklearn.metrics import confusion_matrix,
```

```
    classification_report,      matthews_corrcoef,   cohen_kappa_score,
    accuracy_score,  average_precision_score, roc_auc_score
```

In [ ]:

```
    from     sklearn.metrics     import     accuracy_score,
    confusion_matrix    from    sklearn.ensemble    import
    RandomForestClassifier   from   sklearn.model_selection
    import  cross_val_score  rfc=  RandomForestClassifier()
    rfc.

    fit(X_train,y_train)

    predictRF = rfc.predict(X_test)
 print("") print('Classification report of Random Forest
    Results:') print("")
    print(classification_report(y_test,predictRF))
 print("") cm1=
    confusion_matrix(y_test,predictRF) print('Confusion Matrix result
    of Random Forest Classifier is:\n',cm1) print("") sensitivity1 =
    cm1[0,0]/(cm1[0,0]+cm1[0,1]) print('Sensitivity : ', sensitivity1
    )   print("")   specificity1   =   cm1[1,1]/(cm1[1,0]+cm1[1,1])
    print('Specificity : ', specificity1) print("")


    accuracy    =    cross_val_score(rfc,    X,    y,
    scoring='accuracy')  print('Cross  validation  test
    results of accuracy:') print(accuracy)
    #get  the  mean  of  each  fold
    print("")
    print("Accuracy result of Random Forest Classifier is:",accuracy.mean() *
    100)
    RF=accuracy.mean() * 100
```

In [ ]:

```
     def graph():       import matplotlib.pyplot as plt
           =[RF]
        data
        alg="Random  Forest  Classifier"  plt.figure(figsize=(5,5))  b=plt.
bar(alg,data,color=("r"))     plt.title("Accuracy  comparison  of  Earth
Quake",fontsize=15)  plt.legend(b,data,fontsize=9)
```

In [ ]:

```
    graph()
```

In [ ]:

```
    port      warnings      warnings.
    filterwarnings('ignore')
```

In [ ]:

In [ ]:

```
data=p.read_csv('creditcard.csv')
 del data['Merchant_id']
```

In [ ]:

```
    del data['TransactionDate'] df=data.
```

In [ ]:
```
dropna()
```

In [ ]:
```
df.columns
```

```
from sklearn.preprocessing import LabelEncoder var_mod =
['Is_declined','isForeignTransaction', 'isHighRiskCountry',
'isFradulent'] le = LabelEncoder()
```

In [ ]:
```
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)
```

```
#preprocessing, split test and dataset, split response variable
X = df.drop(labels='isFradulent', axis=1)
#Response variable
y = df.loc[:,'isFradulent']
```

In [ ]:
```
'''We'll use a test size of 30%. We also stratify the split on the
response variable, which is very important to do because there are so
few fraudulent transactions'''
from sklearn.model_selection import
train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30,
random_state=
1, stratify=y) print("Number of training
dataset: ", len(X_train)) print("Number of test
dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))
```

In [ ]:
```
#According to the cross-validated MCC scores, the random forest is
the best-performing model, so now let's evaluate its performance on
the test set. from sklearn.metrics import confusion_matrix,
classification_report, matthews_corrcoef, cohen_kappa_score,
accuracy_score, average_precision_score, roc_auc_score
```

In [ ]:
```
from sklearn.metrics import accuracy_score,
confusion_matrix from sklearn.tree import
DecisionTreeClassifier from sklearn.model_selection
import cross_val_score dtree= DecisionTreeClassifier()
dtree.

fit(X_train,y_train)

predictDT =

dtree.predict(X_test)

print("")
```

```
    print('Classification  report

    of  Decision  Tree  Results:')

    print("")

    print(classification_report(y_test,predictDT))
print("") cm1=
    confusion_matrix(y_test,predictDT)
    print('Confusion  Matrix  result  of  Decision  Tree  Classifier
    is:\n',cm1) print("") sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
    print('Sensitivity : ', sensitivity1 ) print("") specificity1 =
    cm1[1,1]/(cm1[1,0]+cm1[1,1]) print('Specificity : ', specificity1)
    print("")
accuracy = cross_val_score(dtree, X, y, scoring='accuracy')
    print('Cross  validation  test  results  of  accuracy:')
    print(accuracy)
    #get  the  mean  of  each  fold
    print("")
    print("Accuracy result of Decision Tree Classifier is:",accuracy.mean() *
    100)
    DT=accuracy.mean() * 100
```

In [ ]:

```
    def graph():  import matplotlib.pyplot as plt
        data=[DT]       alg="Decision        Tree
        Classification"                      plt.
        figure(figsize=(5,5))
        b=plt.bar(alg,data,color=
        ("pink"))       plt.title("Accuracy     comparison     of     Earth
        Quake",fontsize=15) plt.legend(b,data,fontsize=9)
```

In [ ]:

```
    graph()
```

In [ ]:

```
    TP = cm1[0][0]
    FP = cm1[1][0]
    FN = cm1[1][1]
    TN      =      cm1[0][1]
    print("True     Positive
    :",TP)       print("True
    Negative          :",TN)
    print("False    Positive
    :",FP)      print("False
    Negative          :",FN)
    print("")      TPR     =
    TP/(TP+FN)
    TNR = TN/(TN+FP)
    FPR = FP/(FP+TN)
    FNR = FN/(TP+FN)
    print("True    Positive    Rate
    :",TPR)  print("True  Negative
    Rate    :",TNR)    print("False
    Positive      Rate      :",FPR)
```

```
print("False Negative Rate
:",FNR) print("")
PPV = TP/(TP+FP) NPV = TN/(TN+FN)
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)
```

In [ ]:

```
def plot_confusion_matrix(cm1, title='Confusion matrix-
DecisionTreeClassifier', cmap=plt.cm.Blues):
    target_names=['Predict','Actual'] plt.imshow(cm1,
    interpolation='nearest', cmap=cmap)
    plt.title(title) plt.colorbar() tick_marks =
    n.arange(len(target_names)) plt.
    xticks(tick_marks, target_names, rotation=45) plt.
    yticks(tick_marks, target_names) plt.tight_layout()
    plt.ylabel('True label') plt.
    xlabel('Predicted label')
cm1=confusion_matrix(y_test, predictDT)
print('Confusion matrix-
DecisionTreeClassifier:') print(cm1)
plot_confusion_matrix(cm1)
```

Naïve Bayes Algorithm

```
#import library packages import
pandas as p
import matplotlib.pyplot as plt
import seaborn as s import
numpy as n
```

```
import warnings warnings.
filterwarnings('ignore')
```

In [ ]:

```
#Load given dataset
data = p.read_csv('creditcard.csv')
```

In [ ]:

```
del data['TransactionDate']
del data['Merchant_id']
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
df = data.dropna() df.
```

```
columns
```

```
from sklearn.preprocessing import LabelEncoder
```

```
    'isFradulent'] le =
    LabelEncoder()  for
    i in var_mod:
        df[i] = le.fit_transform(df[i]).astype(int)


#preprocessing, split test and dataset, split response variable
    X = df.drop(labels='isFradulent', axis=1)
    #Response variable                                                    In [ ]:
      = .                                                                In [ ]:
'''We'll use a test size of 30%. We also stratify the split on the response variable,
    which is very important to do because there are so few fraudulent
    transactions'''
 from        sklearn.model_selection        import
    train_test_split
    X_train,  X_test,  y_train,  y_test  =  train_test_split(X,  y,  test_size=0.30,
    random_state=
    1,  stratify=y)  print("Number  of  training
    dataset: ", len(X_train)) print("Number of test
    dataset: ", len(X_test))
    print("Total number of dataset: ", len(X_train)+len(X_test))
                                                                         In [ ]:

    #According to the cross-validated MCC scores, the random forest is
    the best-performing model, so now let's evaluate its performance on
    the test set.      from sklearn.metrics import confusion_matrix,
    classification_report,      matthews_corrcoef,   cohen_kappa_score,
    accuracy_score, average_precision_score, roc_auc_score


                                                                         In [ ]:
    from     sklearn.metrics     import     accuracy_score,

    confusion_matrix   from   sklearn.naive_bayes   import

    GaussianNB    from    sklearn.model_selection    import

    cross_val_score        gnb        =        GaussianNB()

    gnb.fit(X_train,y_train)          predictNB          =

    gnb.predict(X_test)


    print("")
    print('Classification   report   of   Naive   Bayes
    Results:')                                print("")
    print(classification_report(y_test,predictNB))
print("") cm1=
    confusion_matrix(y_test,predictNB)  print('Confusion
    Matrix result of Naive Bayes is:\n',cm1) print("")
    sensitivity1                              =
    cm1[0,0]/(cm1[0,0]+cm1[0,1])
    print('Sensitivity : ', sensitivity1 )
    print("")
    specificity1                              =
    cm1[1,1]/(cm1[1,0]+cm1[1,1])
    print('Specificity  :  ',  specificity1)
    print("")
```

```
accuracy = cross_val_score(gnb, X, y,
scoring='accuracy') print('Cross validation test
results of accuracy:') print(accuracy)
#get the mean of each fold print("") print("Accuracy result of Naive Bayes Algorithm
is:",accuracy.mean() * 100) NB=accuracy
.mean() * 100
```

In [ ]:

```
def graph():
    import matplotlib.pyplot as plt data=
    [NB] alg="Naive Bayes" plt.
    figure(figsize=(5,5))
    b=plt.
    bar(alg,data,color=("y"))

    plt.title("Accuracy comparison of Earth Quake",fontsize=15) plt.
    legend(b,data,fontsize=9)
```

In [ ]:

```
graph()
```

In [ ]:

```
TP = cm1[0][0]
FP = cm1[1][0]
FN = cm1[1][1]
TN = cm1[0][1] print("True
Positive :",TP) print("True
Negative :",TN)
print("False Positive
:",FP) print("False
Negative :",FN)
print("") TPR =
TP/(TP+FN)
TNR = TN/(TN+FP)
FPR = FP/(FP+TN) FNR = FN/(TP+FN)
print("True Positive Rate :",TPR)
print("True Negative Rate :",TNR)
print("False Positive Rate :",FPR)
print("False Negative Rate :",FNR)
print("") PPV = TP/(TP+FP) NPV =
TN/(TN+FN)
```

In [ ]:

```
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)


def plot_confusion_matrix(cm1, title='Confusion matrix-Naive Bayes',
cmap=plt.cm.Blues):
    target_names=['Predict','Actual'] plt.imshow(cm1,
    interpolation='nearest', cmap=cmap) plt.
    title(title) plt.colorbar() tick_marks =
    n.arange(len(target_names)) plt.
    xticks(tick_marks, target_names, rotation=45) plt.
    yticks(tick_marks, target_names) plt.
    tight_layout()
    plt.ylabel('True label')
    plt.
    xlabel('Predicted label')
cm1=confusion_matrix(y_test, predictNB)
print('Confusion matrix-Naive Bayes:')
print(cm1)
plot_confusion_matrix(cm1
```

HTML Code:

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>TITLE</title>
<link rel="stylesheet" href="{{ url_for('static', filename='css/bootstrap.min.css')
}}">
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
<link
href='https://fonts.googleapis.com/css?family=Open+Sans+Condense
d:300' rel='stylesheet' type='text/css'>
<style>
.back{ background-image: url("{{ url_for('static',
 filename='image/card.gif') }}");
background-repeat:no-
repeat; background-
size:cover; }
.white{
color:white; } .nspace{
margin:15px 15px 30px
```

```
30px; padding:9px 10px;
background: palegreen;
width:500px
} .space{
margin:10px 30px;
padding:10px
10px; background:
palegreen;
width:500px
}
.gap{
padding:10px
20px;
}
</style>

</head>

<body >
 <div>
        <div class="jumbotron">
<h1 style="text-align:center"> CREDIT CARD FRAUD DETECTION
</h1>

        </div>
<div class="back">
    <!-- Main Input For Receiving Query to our ML -->
   <form class="form-group" action="{{ url_for('predict')}}"method="post">

            <div class="row">
            <div class="gap col-md-6 ">
```

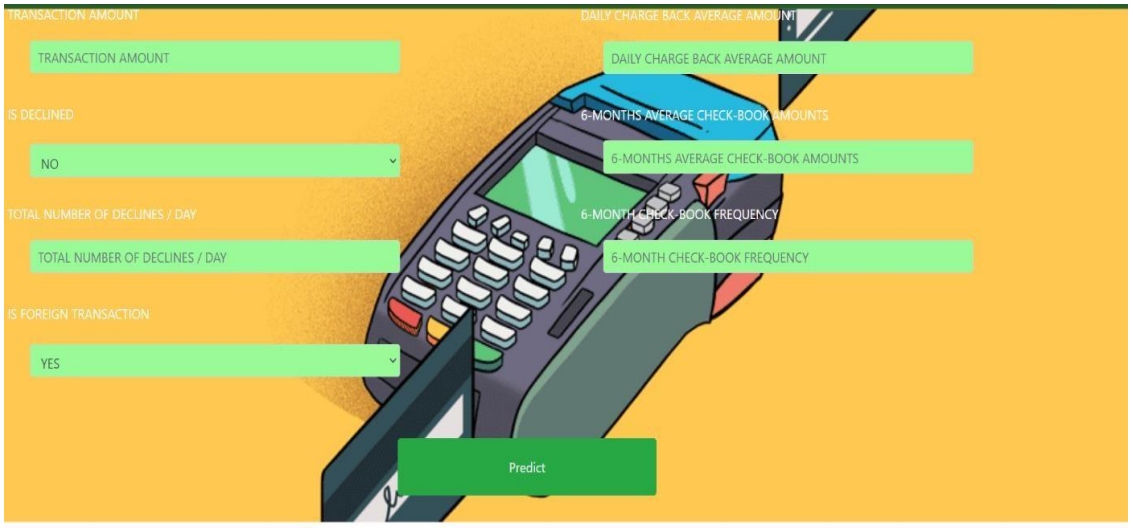
```
<label class="white" for="">AVERAGE AMOUNT TRANSACTION / DAY</label>
<input type="number" class="space form-control" step="0.01" name="AVERAGE AMOUNT TRANSACTION / DAY" placeholder="AVERAGE AMOUNT TRANSACTION / DAY" required="required" /><br>

<label class="white" for="">TRANSACTION AMOUNT</label>
<input type="number" class="space form-control" step="0.01" name="TRANSACTION AMOUNT" placeholder="TRANSACTION AMOUNT" required="required" /><br>

<label class="white" for="">IS DECLINED</label>
<select class="nspace form-control" name="IS DECLINED" id="IS DECLINED">
 <option value=0>NO</option>
 <option value=1>YES</option>
</select>

<label class="white" for="">TOTAL NUMBER OF DECLINES / DAY</label>
<input type="number" class="space form-control" step="0.01" name="TOTAL NUMBER OF DECLINES / DAY" placeholder="TOTAL NUMBER OF DECLINES / DAY" required="required" /><br>

<label class="white" for="">IS FOREIGN TRANSACTION</label>
```

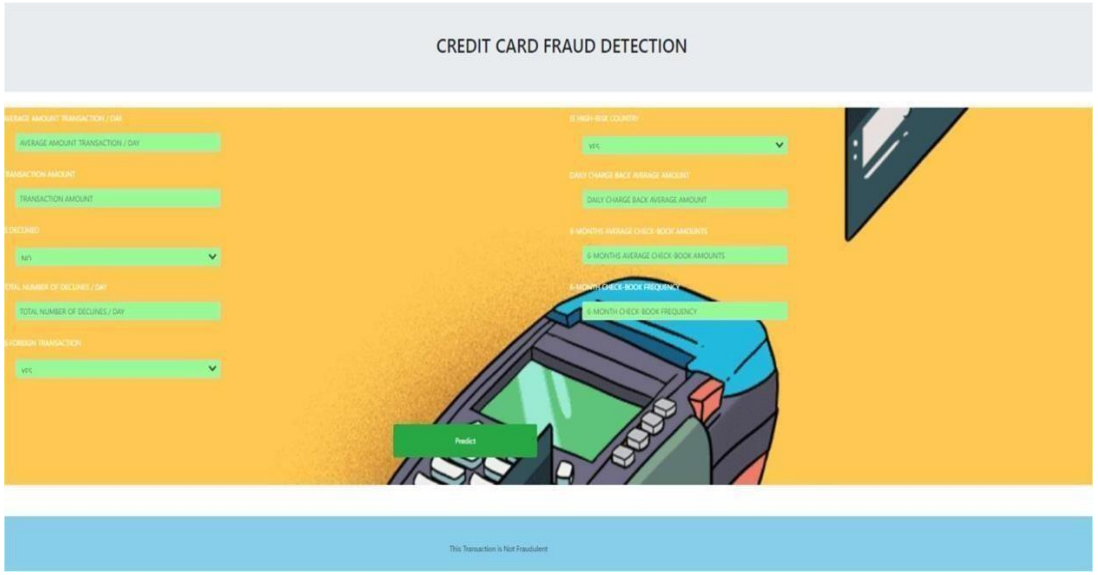
```
                <select class="nspace form-control" name="IS FOREIGN
    TRANSACTION" id="IS FOREIGN TRANSACTION">
      <option value=1>YES</option>
      <option value=0>NO</option>
    </select>

                </div>

    <div class="gap col-md-6">

                <label class="white" for="">IS HIGH-RISK COUNTRY</label>

                <select class="nspace form-control" name="IS HIGH-RISK
    COUNTRY" id="IS HIGH-RISK COUNTRY">
      <option value=1>YES</option>
      <option value=0>NO</option>
    </select>

                <label class="white" for="">DAILY CHARGE BACK
    AVERAGE AMOUNT</label>
                <input type="number" class="space form-control" step="0.01"
    name="DAILY CHARGE BACK AVERAGE AMOUNT" placeholder="DAILY
    CHARGE BACK AVERAGE AMOUNT" required="required" /><br>

<label class="white" for="">6-MONTHS AVERAGE CHECK-
    BOOK AMOUNTS</label>
                <input type="number" class="space form-control"
    step="0.01" name="6-MONTHS AVERAGE CHECK-BOOK
    AMOUNTS" placeholder="6-MONTHS AVERAGE CHECK-BOOK
    AMOUNTS" required="required" /><br>
```

```
                <label class="white" for="">6-MONTH CHECK-BOOK
FREQUENCY</label>
                <input type="number" class="space form-control" step="0.01"
name="6-MONTH CHECK-BOOK FREQUENCY" placeholder="6-MONTH
CHECK-BOOK FREQUENCY" required="required" /><br>


</div>
</div>

<div style="padding:2% 35%">
 <button type="submit" class="btn btn-success btn-block"
style="width:350px;padding:20px">Predict</button>
</div>

  </form>
</div>
  <br>
  <br>

<div style="background:skyblue;padding:2% 40%">
  {{ prediction_text }}
</div>
</div>

</body>
</html>
```

CHAPTER 6
RESULT AND DISCUSSIONS

Output Screenshot





CHAPTER 7
CONCLUSION AND FUTURE ENHANCEMENT

In conclusion, the design and implementation of a credit card fraud detection system using machine learning represent a critical endeavor in safeguarding financial transactions and protecting both consumers and businesses from fraudulent activities. By following a comprehensive schema that encompasses data collection, preprocessing, model selection, deployment, and continuous monitoring, organizations can develop robust and effective fraud detection systems.

Through diligent data preprocessing techniques, including handling missing values, outliers, and feature engineering, the system can extract meaningful insights from transactional data, enhancing the performance of machine learning models. The selection of appropriate algorithms and thorough evaluation ensures the chosen model effectively distinguishes between legitimate and fraudulent transactions, minimizing false positives and negatives.

Deployment of the model into a production environment, coupled with continuous monitoring and improvement mechanisms, guarantees the system's reliability, scalability, and adaptability to evolving fraud patterns. By integrating alert mechanisms and adhering to compliance and security standards, organizations can promptly respond to suspicious activities while safeguarding customer privacy and sensitive information.

Looking ahead, future enhancements such as advanced feature engineering, ensemble methods, anomaly detection, and deep learning architectures promise to

further enhance the system's effectiveness and resilience against emerging threats. Incorporating behavioral biometrics, blockchain technology, and real-time decision-making capabilities will bolster fraud detection accuracy and efficiency, ensuring swift actions to mitigate potential losses.

Ultimately, the collaborative effort among financial institutions, regulatory bodies, and technology providers, coupled with advancements in machine learning and data analytics, will continue to strengthen credit card fraud detection systems, bolstering trust in financial transactions and safeguarding the integrity of the global financial ecosystem.

FUTURE ENHANCEMENTS:

Advanced Feature Engineering: Explore more sophisticated feature engineering techniques, such as time-series analysis, graph-based representations of transaction networks, or embeddings derived from deep learning models. Ensemble Methods: Investigate the effectiveness of ensemble learning techniques, combining multiple models to improve overall performance and resilience against adversarial attacks. Anomaly Detection: Incorporate anomaly detection algorithms alongside traditional classification models to detect previously unseen fraud patterns or novel attack vectors. Deep Learning Architectures: Experiment with more complex deep learning architectures such as graph neural networks or attention mechanisms to capture intricate relationships within transaction data. Online Learning and Adaptive Models: Implement online learning algorithms that can adapt to changing fraud patterns in real-time, allowing the system to continuously learn from incoming data without the need for periodic retraining. Explainability and Interpretability: Enhance model interpretability by incorporating techniques such as SHAP (SHapley Additive exPlanations) values or LIME (Local Interpretable Model-agnostic Explanations), enabling stakeholders to understand the reasoning behind model predictions. Behavioral Biometrics: Integrate behavioral biometrics such as keystroke dynamics or mouse movement patterns to augment existing features and improve fraud detection accuracy. Blockchain and Distributed

Ledgers: Explore the use of blockchain technology and distributed ledgers to enhance transaction transparency, traceability, and immutability, thereby reducing the risk of fraud and enhancing trust in the financial system. Real-Time Decision Making: Develop real-time decision-making capabilities within the system to automatically block suspicious transactions or trigger fraud alerts without human intervention, reducing response time and mitigating potential losses. Cross-Industry Collaboration: Foster collaboration between financial institutions, regulatory bodies, and technology providers to share data, insights, and best practices in combating fraud effectively while ensuring compliance with regulatory requirements.

CHAPTER 8
REFERENCES

1. Akanksha Bansal and Hitendra Garg, An Efficient Techniques for Fraudulent
detection in Credit Card Dataset: A Comprehensive study, IOP Conference series: Materials Science And Engineering, Vol.1116, issue.1, 2021.

2. Devika S P, Nisarga K S, Gagana P Rao, Chandini S B, Rajkumar N, A Research On Credit Card Fraud Detection System, International Journal of Recent Technology and Engineering (IJRTE) Volume-8 Issue-2, July 2019.

3. Harish Paruchuri, Credit Card Fraud Detection using Machine Learning: A Systematic Literature Review, ABC Journal of Advanced Research, Volume 6, No 2, 2017.

4. S. Abinayaa, H. Sangeetha, R. A. Karthikeyan, K. Saran Sriram, D. Piyush, Credit Card Fraud Detection and Prevention using Machine Learning, International Journal of Engineering and Advanced Technology (IJEAT), Volume-9 Issue-4, April, 2020

5. Suresh K Shirgave, Chetan J. Awati, Rashmi More, Sonam S. Patil, A Review On Credit Card Fraud Detection Using Machine Learning, International Journal Of Scientific & Technology Research Volume 8, Issue 10, October 2019.

6. V. Jain, H. Kavitha and S. Mohana Kumar, Credit Card Fraud Detection Web Application using Streamlit and Machine Learning, IEEE International Conference on Data Science and Information System (ICDSIS), Hassan, India,vol.1109, 2022.