CREDIT CARD FRAUD DETECTION USING SUPERVISED MACHINE LEARNING

Shubhadeep Nath (20UCS001)
Sahil Dey (20UCS008)
Sourajita Chanda (20UCS041)
Tithi Majumder (20UCS131)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, AGARTALA
INDIA – 799046
NOVEMBER–2023

CREDIT CARD FRAUD DETECTION USING SUPERVISED MACHINE LEARNING

Report Submitted to the
National Institute of Technology, Agartala
for the award of the degree

of

Bachelor of Technology

by

Shubhadeep Nath(20UCS001)
Sahil Dey(20UCS008)
Sourajita Chanda(20UCS041)
Tithi Majumder(20UCS131))

Under the Guidance of
Dr. Kunal Chakma
Assistant Professor, CSE
NIT Agartala, India

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, AGARTALA
INDIA–799046
NOVEMBER–2023

DEDICATED TO

To our Project Supervisor **Dr. Kunal Chakma, Assistant Professor, CSED, NIT Agartala** for sharing his valuable knowledge, and encouragement and for showing confidence in us all the time. Each of the faculties of the department to contribute in ourdevelopment as a professional and help us to achieve this goal. To all those people who have somehow contributed to the creation of this project and who have supported us.

REPORT APPROVAL

Date:.....

The report entitled ***'CREDIT CARD FRAUD DETECTION'*** submitted by ***by Shubhadeep Nath(20UCS001), Sahil Dey(20UCS008),Sourajita Chanda(20UCS041),Tithi Majumder(20UCS131)*** to the National Institute of Technology, Agartala has been approved for the award of Bachelor of Technology in Computer Science and Engineering.

Dr. Kunal Chakma	**Dr. Suman Deb**
(**Supervisor**)	(**Head of the Department**)

DECLARATION

We declare that the work presented in this report proposal titled "CREDIT CARD FRAUD DETECTION'", submitted to the Computer Science and Engineering Department, National Institute of Technology, Agartala, for the award of the Bachelor of Technology degree in Computer Science and Engineering, represents our ideas in our own words and where others' ideas or words have been included, and we have adequately cited and referenced the sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented, fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of theabove will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Shubhadeep Nath (**20UCS001**)	**Sahil Dey** (**20UCS008**)
Sourajita Chanda (**20UCS041**)	**Tithi Majumder** (**20UCS131**)

Date:_____

CERTIFICATE

This is to certify that the report entitled ***'CREDIT CARD FRAUD DETECTION'***, submitted by ***Shubhadeep Nath(20UCS001), Sahil Dey(20UCS008), Sourajita Chanda(20UCS041), Tithi Majumder(20UCS131)***,has been carried out under my supervision and this work has not been submitted elsewhere for a degree.

Dr. Kunal Chakma
Supervisor
Assistant Professor, NIT
Agartala
Computer Science and
Engineering Department

ACKNOWLEDGMENTS

We would like to take this opportunity to express our deep sense of gratitude to all who helped us directly or indirectly during this thesis work. Firstly, we would like to thank our supervisor, **Dr. Kunal Chakma**, for being a great mentor and the best adviser we could ever have. His advice, encouragement and criticism are sources of innovative ideas, inspiration and causes behind the successful completion of this report. His confidence shown to us was the biggest source of inspiration for us. It has been a privilege working with him for the past 6 months. We are highly obliged to all the faculty members of Computer Science and Engineering Department for their support and encouragement. We also thank **Prof.(Dr.) S. K. Patra**, Director,NIT Agartala and **Dr. Suman Deb**, H.O.D, CSED for providing excellent computing and other facilities without which this work could not achieve its quality goal.

Shubhadeep Nath(20UCS001)
Sahil Dey(20UCS008)
Sourajita Chanda(20UCS041)
Tithi Majumder(20UCS131)

Contents

List of Figures

List of Tables

Chapter 1

Abstract

The project explores the creation of a sophisticated credit card fraud detection system through the application of machine learning methods, namely Random Forest, Support Vector Machines (SVM), and Logistic Regression. Analysis is based on transactional data, and each algorithm is carefully fine-tuned and trained to identify patterns suggestive of fraudulent activity. The model's performance is thoroughly examined, including measures for accuracy, precision, recall, and F1 score. The research findings provide light on the advantages and disadvantages of each method, assisting in the careful selection of the best model for real-world application. These results support the practicality of using machine learning for fraud detection and provide insightful information on algorithmic subtleties that will aid in the development of a credit card fraud detection system that operates at peak efficiency.

Chapter 2

INTRODUCTION

As the number of digital transactions grows, credit card theft has developed as a persistent and sophisticated danger to the financial system. With the rise of online commerce and electronic payment systems, fraudsters are constantly devising new ways to exploit vulnerabilities in the transactional environment, resulting in significant financial losses. As traditional payment methods migrate to the digital arena, effective and adaptable credit card fraud detection systems become critical. Machine Learning has become a household term when it comes to real-time identification of fraudulent instances because to the sheer volume of complicated data collected everyday. From conventional statistical models to advanced deep learning architectures, machine learning algorithms have the capacity to identify trends and abnormalities in large datasets, making it possible to identify fraudulent transactions more accurately and efficiently. Through investigating the field of machine learning-based credit card fraud detection, this research makes a valuable contribution to the development of sophisticated, flexible, and expandable solutions. By enabling financial institutions to outperform criminals proactively, these innovations foster a dependable and safe digital financial environment.

2.1 Motivation

The motivation for this project lies in the escalating threat of fraudulent activities in the digital financial landscape. With the rise of digital transactions, the financial industry faces increased vulnerabilities to sophisticated fraud schemes. The imperative to mitigate financial losses, sustain consumer trust, comply with regulations, and counter adaptive fraud techniques propels the ongoing development and implementation of robust and innovative

fraud detection systems. These efforts aim to ensure the security and resilience of electronic payment systems, fostering confidence among consumers and fortifying the overall integrity of the financial ecosystem.

2.2 Objective

The objective of the project is to employ advanced technologies and analytical methods to swiftly and accurately identify and prevent fraudulent transactions. This proactive approach aims to mitigate financial losses, protect consumer trust, and ensure the overall security and reliability of digital payment systems.

2.3 Advantages

Credit card fraud detection offers the following advantages:

- **Financial Loss Prevention**: Minimizes monetary losses by identifying and preventing fraudulent transactions.
- **Enhanced Customer Trust**: Builds and maintains trust among customers, encouraging continued card usage.
- **Compliance with Regulations**: Ensures adherence to regulatory standards related to fraud prevention.
- **Risk Mitigation**: Manages and mitigates risks associated with fraudulent activities.
- **Data Security**: Safeguards sensitive information, reducing the risk of data breaches.
- **Adaptive Measures**: Uses machine learning to adapt to evolving fraud tactics.
- **Operational Efficiency**: Operates in real-time, minimizing manual intervention and ensuring quick response.
- **Reduced False Positives**: Aims to minimize legitimate transactions being mistakenly flagged as fraudulent.
- **Strategic Decision-Making**: Analyzes data to identify trends and vulnerabilities for informed decision-making.

- **Global Impact**: Contributes to the global fight against financial crimes through collaborative efforts.

2.4 Contributions of the Work

This work contributes towards the ongoing and evolving efforts to identify, prevent, and mitigate fraudulent activities. This involves the application of data analytics and machine learning techniques to detect anomalies, unauthorized access, and fraudulent patterns.This work further contributes to:

- **Real-time Monitoring**: Continuously monitoring credit card transactions in real-time to identify suspicious activities promptly.
- **Pattern Recognition**: Utilizing data analytics to recognize patterns indicative of fraud, including unusual spending behavior, geographic anomalies, and transaction irregularities.
- **Predictive Modeling**:Employing predictive modeling and machine learning algorithms to anticipate potential fraud based on historical data and emerging patterns.
- **Behavioral Analysis**: Analyzing user behavior to establish a baseline and detect deviations that might indicate fraudulent activities.
- **Collaborative Solutions**: Encouraging collaboration among financial institutions, payment processors, and law enforcement agencies to share information and enhance the collective ability to combat fraud.
- **User Education**: Including initiatives to educate users about safe practices, security measures, and how to recognize and report suspicious activities.
- **Continuous Improvement**: Embracing a culture of continuous improvement by regularly updating and refining fraud detection systems to address new challenges and vulnerabilities.

2.5 Organisation of the thesis

The rest of the thesis is organised into the following chapters:

- **Chapter 3:** The literature review is discussed in this chapter.
- **Chapter 4:** This chapter focuses on the dataset description, preprocessing of the given data, handling the imbalances of the classes, Exploratory Data Analysis(EDA),Data Splitting,
- **Chapter 5:** This chapter focuses on the description of the proposed models for the given dataset, the algorithms that are implemented, the experimental setup that is used and evaluation of the models
- **Chapter 6:** This chapter deals with the analysis and comparison between the results of the different proposed models in the previous chapter
- **Chapter 7:**The conclusion of the report is described in this chapter & also described the future scopes of the project.
- **Chapter 8:**All the references & the inspirations that we have taken and countered for this project are mentioned in this chapter.

Chapter 3

LITERATURE REVIEW

3.1 Literature Review

Numerous publications on **Credit Card Fraud Detection System** were looked through during the literature review process. A few of these articles examined and suggested various techniques for detecting fraud using various angles.Some, however, emphasised that, as opposed to depending solely on heuristic-based techniques, fraud detection in credit card transactions is a supervised learning classification job. Data mining applications, automated fraud detection, and adversarial detection are among the methods used in the fraud detection area, according to a thorough assessment carried out by Clifton Phua and his colleagues. This paper published in the IJERT mainly proposes the use of the latest machine learning algorithms to identify the anomalies or the outliers. The proposed solution in their model involves the implementation of two algorthims *Local Outlier Factor(unsupervised)*, *Isolation Forest Algorithm*. Using these two algorithms the researchers were able to produce the desired accuracy This paper taken from the website of Rochester Institute of Technology mainly proposes the use of Supervised machine learning algorithms to detect the frauds in the credit card transactions. The models that are implemented in the published paper are *Naive Bayes*, *Logistic Regression*, *Support Vector Machine*. Comparing the results pf these three proposed models the researcher got the desired accuracy.

Chapter 4

DATASET

4.1 Problem Statement

The problem for detecting a fraud in a transaction of a credit card with the proper knowledge and modeling it based on the record of transactions,& to check if the built model can precisely identify and check if the new transaction is a fraud one or not.

4.2 Description of Dataset

The dataset for the credit card fraud detection project contsists a large collection of transactions, each characterized by various features that provide information about the transaction. The dataset is typically labeled, indicating whether each transaction is fraudulent or legitimate. The dataset was collected from Kaggle website so as to conduct the experiments.

Here's a brief description of the key components in the given dataset:

1. **Transaction Features:**
   • **Time**: Timestamp indicating when the transaction occurred.
   • **Amount**: The monetary value of the transaction.
   • **V1, V2, ..., V28**: Anonymized features representing various aspects of the transaction. These features are often the result of dimensionality reduction techniques like Principal Component Analysis (PCA) to protect sensitive information.

2. **Class Label:**
   • **Class (Fraud or Legitimate)**:A binary label indicating whether the transaction is fraudulent (Class 1) or legitimate (Class 0)

3. **Imbalanced Class Distribution:**
   • Credit card fraud datasets are often imbalanced, with a majority of transactions being legitimate. The imbalance ratio between fraudulent and legitimate transactions can be significant

4. **Real-time vs Historical Data:**
   • The dataset may represent historical transaction data, allowing for retrospective analysis. In some cases, datasets might be designed to simulate real-time processing

4.3 Data Preprocessing

Data preprocessing involves the systematic preparation and transformation of raw data to improve its quality and suitability for analysis or machine learning applications. This process includes addressing missing values, handling duplicates and outliers, scaling numerical features, encoding categorical variables, and performing other adjustments to ensure the data is ready for effective exploration, modeling, and interpretation.

1. **Handling Duplicate Values:** Duplicate values in a dataset can lead to biased analyses and model training. Identifying and removing duplicate rows is essential for integrity and to prevent skewed results.

2. **Handling Null Values:** Null or missing values can disrupt analyses and model training, potentially resulting in biased or incomplete results. Addressing null values through imputation or removal ensures a more complete and reliable dataset.

3. **Handling Outliers:** Outliers can significantly impact statistical summaries and machine learning models, leading to inaccurate insights and predictions. Identifying and appropriately handling outliers is crucial to mitigate their impact and ensure robust and accurate analysis or modeling.

4. **Data Scaling:** Inconsistent scales among features can lead to biased model training and inaccurate predictions in machine learning. Standardizing or normalizing numerical features through data scaling ensures fair treatment of all features during model training, improving the model's performance and interpretability.

4.4 Handling Class Imbalance

A Python library called 'Imabalanced-Learn' is designed to address the challenges posed by imbalanced classes . Imbalance in datasets occur when the classes in the target variable is skewed, leading to issues in training models that may be biased towards the majority class. Imbalanced-learn provides various techniques for handling imbalanced class, which includes oversampling the minority class, undersampling the majority class, and generating synthetic samples using SMOTE (Synthetic Minority Over-sampling Technique). By integrating imbalanced-learn we improve the performance of models on minority classes and achieve more accurate predictions in scenarios where class imbalance is a concern.

4.5 Exploratory Data Analysis

- Data Overview: Initiated Exploratory Data Analysis (EDA) to gain comprehensive insights into the structure and characteristics of our datasets.

- Distribution Analysis: Examined data distributions to understand the central tendencies and spread of key variables, laying the groundwork for subsequent analysis.

- Correlation Assessment: Investigated correlations between variables, unveiling potential relationships and dependencies crucial for informed decision-making.

- Outlier Identification: Implemented outlier detection mechanisms to identify and address anomalies, ensuring data integrity and reliability.

- Pattern Recognition: Leveraged Python tools and libraries for generating visualizations, aiding in the identification of patterns and trends within the datasets.

- Informative Visuals: Created plots, charts, and graphs to communicate complex data relationships in a clear and concise manner, facilitating better comprehension for stakeholders.

- Foundation for Analysis: EDA served as the foundational phase, providing a comprehensive understanding of the data landscape and guiding subsequent analytical endeavors.

4.5.1 Display Dataframe

The `credit_card.head()` method displays the initial rows of the Pandas DataFrame `'credit_card'`. It serves as a quick overview, presenting the first 5 rows by default, to gain insights into the dataset's structure and content.

This method is specifically associated with Pandas DataFrames :

1. Overview:
    - `credit_card` is assumed to be a Pandas DataFrame representing a dataset related to credit card transactions (as suggested by the name).
    - The `head()` method is used to display the first few rows of the DataFrame.
2. Syntax:
    - The `head()` method is called on a DataFrame, and it returns the first 5 rows (by default) of the DataFrame.
3. Purpose:
    - It's primarily used for a quick exploration of the dataset to get a sense of its structure, contents, and the format of the data.
    - Viewing the first few rows is helpful in understanding the column names, data types, and example values in each column.

4.5.2 Dataframe Info

The `credit_card.info()` method in Python, specifically when applied to a Pandas DataFrame named `credit_card`, provides concise information about the DataFrame's structure.

1. Overview:
    - The `credit_card.info()` method in Python, applied to a Pandas DataFrame named `credit_card`, provides a concise summary of the DataFrame's structure.
    - It offers insights into the number of entries, data types, non-null counts, and memory usage, aiding in a quick assessment of the dataset's characteristics.

2. Purpose:

    - Gain a rapid overview of the DataFrame's structure without displaying the entire dataset.
    - Identify the data types of each column, facilitating comprehension of the data.
    - Obtain insights into the DataFrame's index, including the type and range of index values.
    - Facilitate data cleaning tasks by pinpointing columns with missing values or requiring type conversions.

4.5.3 Dataframe Missing Value Summary

The `credit_card.isna().sum()` method in Python, is used to calculate the number of missing values (NaN or null values) in each column of the `credit_card` Pandas DataFrame.

1. Overview:

    - The expression `credit_card.isna().sum()` is used in Python with Pandas to generate a summary of missing values within each column of the `credit_card` DataFrame. It provides a quick overview of the count of missing values, aiding in the identification of areas in the dataset that require attention or imputation.
    - It offers insights into the number of entries, data types, non-null counts, and memory usage, aiding in a quick assessment of the dataset's characteristics.

2. Purpose:

    - It efficiently counts the number of missing values in each column of the DataFrame.
    - Offers a quick overview of where data might be incomplete, helping to identify columns with missing information.
    - Essential for data cleaning tasks, enabling prioritization of columns for imputation or removal of missing values.

4.5.4 Numerical Column Summary

The `credit_card.describe()` method in Python, when applied to a Pandas DataFrame named `credit_card`, provides a statistical summary of the numerical columns within the DataFrame.

1. Overview:
    - The `credit_card.describe()` method in Python, when applied to a Pandas DataFrame such as `credit_card`, generates a statistical summary of the numerical columns within the DataFrame.
    - The output provides key measures such as the mean, standard deviation, and quartiles, offering a quick and informative overview of the distribution and central tendencies of the data.
2. Purpose:
    - Describes the central location of the data, providing the mean (average) value.
    - Offers insights into the distribution by presenting values at different percentiles.
    - Assists in making informed decisions about data preprocessing, cleaning, and understanding the overall characteristics of the dataset.
    - Indicates the spread or variability of the data through measures like standard deviation.

4.5.5 Transaction Class Distribution

The fig 4.1 here visualizes the distribution of the 'Class' column in a DataFrame, presumably the `credit_card` by implementing the `px.pir()` function from the `plotly.express` library to create a pie chart.

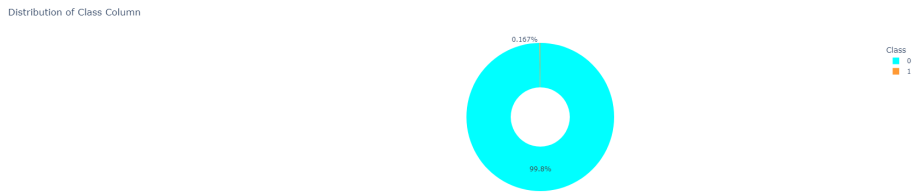

Figure 4.1: Class Distribution

1. Overview:

    - It uses Plotly Express to create a pie chart. The 'Class' column from the `credit_card` DataFrame is used for the pie slices, and additional parameters specify the chart's title, color scheme, and the size of the central hole, creating a donut chart.
    - Modifies the layout settings of the chart. It specifies that the legend should be shown with a title 'Class', and the legend items should have constant sizing. The plot background color is set to white.

2. Purpose:

    - The purpose of this code is to visually represent the distribution of classes in the 'Class' column of the `credit_card` DataFrame using a pie chart.
    - In this context, it provides an intuitive overview of the distribution of the 'Class' column, where classes are represented by different colored segments in the chart.

4.5.6 Time Comparison

4.5.6.1 Fraudulent vs Non-Fraudulent Cases

The fig 4.2 here visualizes the distribution of time among the fraud and non-fraud transactions in the form of bar graph in the DataFrame, presumably the `credit_card` by implementing the `px.bar()` function from the `plotly.express` library to create a bar chart.

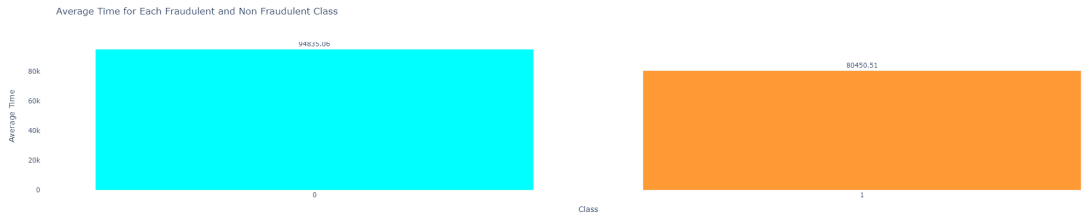

Figure 4.2: Average Transaction Time Comparison: Fraudulent vs. Non-Fraudulent Classes

1. Overview:

    - The code calculates the average time for each class (fraudulent and non-fraudulent) in a credit card dataset.

- It then uses Plotly Express to create a bar chart, with the x-axis representing the classes, the y-axis representing the average time, and text labels displaying the precise average time on each bar.

2. Purpose:

- The chart aims to visually compare and contrast the average times associated with fraudulent and non-fraudulent credit card transactions.
- It provides a quick and intuitive summary of the temporal aspects of these two classes, allowing for easy identification of any patterns or differences in average transaction times.

4.5.7 Median Transaction Amounts

4.5.7.1 Fraudulent vs Non-Fraudulent Cases

The fig 4.3 here visualizes the distribution of total mounts among the fraud and non-fraud transactions in the form of bar graph in the DataFrame, presumably the `credit_card` by implementing the `px.bar()` function from the `plotly.express` library to create a bar chart.

Figure 4.3: Average Amount for Fraud & Non-Fraud Cases

1. Overview:

- The code groups the credit card dataset by the 'Class' column and calculates the median transaction amount for each class.
- It utilizes Plotly Express to create a bar chart, where the x-axis represents the classes, the y-axis represents the median transaction amount, and text labels display the precise median amounts on each bar.

2. Purpose:

- The bar chart aims to provide a quick and intuitive comparison of median transaction amounts between fraudulent and non-fraudulent credit card transactions.
- The visualization facilitates the identification of potential differences in transaction amounts associated with different classes, aiding in exploratory data analysis.
- By using medians, the chart is less sensitive to extreme values and provides a robust measure of central tendency for transaction amounts in each class.

4.5.8 Hourly Transaction Distribution

4.5.8.1 Hourly Transaction Comparison

The fig 4.4 here visualizes the number of total transactions with respect to the hour of the day in the form of grouped bar graph in the DataFrame, presumably the `credit_card` by implementing the `px.bar()` function from the `plotly.express` library to create a grouped bar chart.

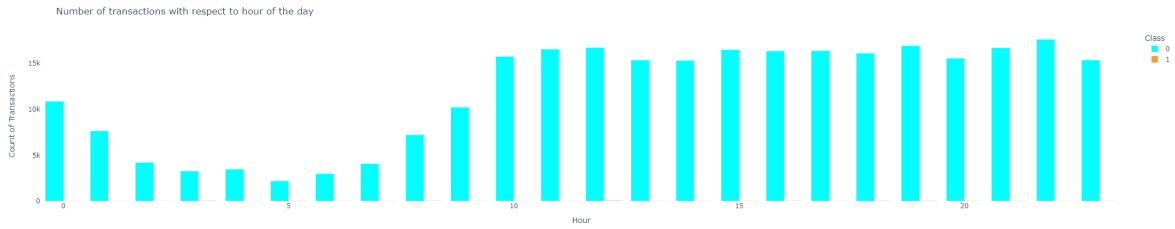

Figure 4.4: Transaction Patterns Across Hours

1. Overview:
   - The code uses `hourly_transaction`, presumably a DataFrame, to represent transaction counts grouped by hour.
   - The chart provides insights into hourly transaction patterns, aiding in the identification of trends and variations throughout the day.
2. Purpose:
   - The chart visually represents the distribution of transaction counts across different hours of the day.

- The chart offers insights into the temporal dynamics of transactions, aiding in understanding daily patterns.

4.5.8.2 Distribution of Non-Fraudulent Transactions Across Hours

The fig 4.5 here visualizes the number of non-fradulent transactions with respect to the hour of the day in the form of grouped bar graph in the DataFrame, presumably the `credit_card` by implementing the `px.bar()` function from the `plotly.express` library to create a grouped bar chart.

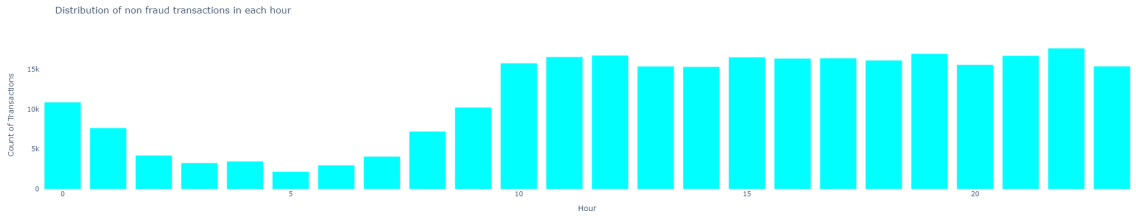

Figure 4.5: Non-Fraudulent Transactions Patterns Across Hours

1. Overview:
   - Extracts the count of non-fraudulent transactions (assuming 0 represents non-fraudulent) from the `hourly_transaction` DataFrame and resets the index, creating a new DataFrame `class_0_df`.
   - Uses Plotly Express to create a bar chart specifically for non-fraudulent transactions.
2. Purpose:
   - The code aims to provide a focused visualization of the distribution of non-fraudulent transactions across different hours of the day.
   - The bar chart helps in understanding the patterns and variations in non-fraudulent transaction counts, with a clear emphasis on visual clarity through color and layout adjustments.

4.5.8.3 Distribution of Fraudulent Transactions Across Hours

The fig 4.6 here visualizes the number of fraudulent transactions with respect to the hour of the day in the form of grouped bar graph in the DataFrame, presumably the `credit_card`

by implementing the `px.bar()` function from the `plotly.express` library to create a grouped bar chart.

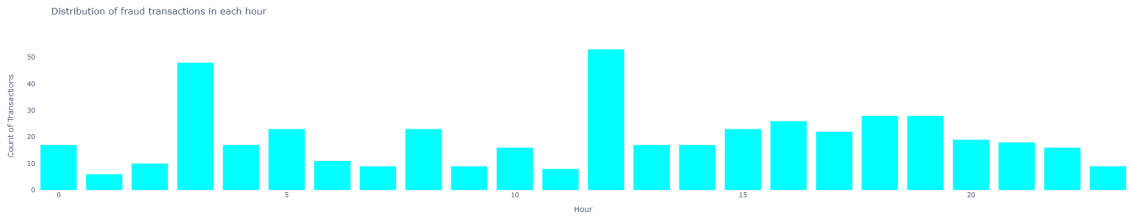

Figure 4.6: Fraudulent Transactions Patterns Across Hours

1. Overview:
    - Extracts data for fraudulent transactions (Class 1) from `hourly_transaction`, creating `class_1_df`.
    - Uses Plotly Express (px.bar) to generate a bar chart for fraudulent transactions.
2. Purpose:
    - The code aims to visually represent and compare the distribution of fraudulent transactions across different hours of the day.
    - The bar chart provides a clear and focused visualization of hourly transaction patterns specifically for fraudulent transactions, with distinctive colors and a white background for improved clarity.

4.5.9 Hourly Average Transactions Amounts

The fig 4.7 here visualizes the number of average transactions by creating a line chart representing the average amount of transactions for each hour of the day. in the DataFrame, presumably the `credit_card` by implementing the `px.line()` function from the `plotly.express` library to create a line chart.

1. Overview:
    - Groups the credit card data by the 'Hour' column and calculates the median transaction amount for each hour.
    - Utilizes Plotly Express `(px.line)` to create a line chart.
2. Purpose:

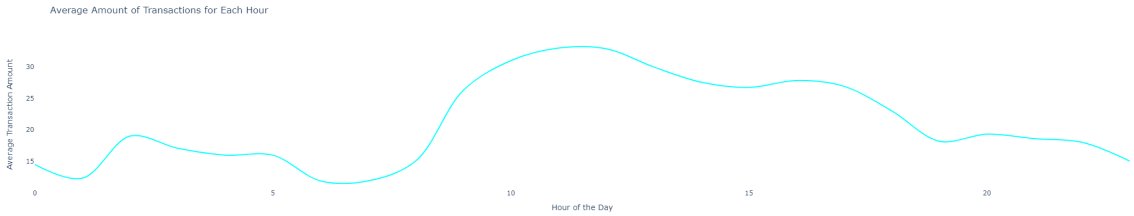

Figure 4.7: Average Transaction Amounts Across Hours

- The code aims to visually convey the average transaction amounts throughout the day, providing insights into potential patterns or variations in transaction values.
- By calculating the median transaction amount for each hour and presenting it in a smooth line chart, the code aims to provide insights into potential patterns or variations in transaction values over different hours.

4.5.10 Dual Perspective: Hourly Fraud vs Non-Fraud Transaction Amounts

The fig 4.8 here visualizes the number of average transactions for both fraudulent & non-fraudulent by creating two line chart simultaneously representing the average amount of transactions for each hour of the day. in the DataFrame, presumably the `credit_card` by implementing the `px.line()` function from the `plotly.express` library to create a line chart.

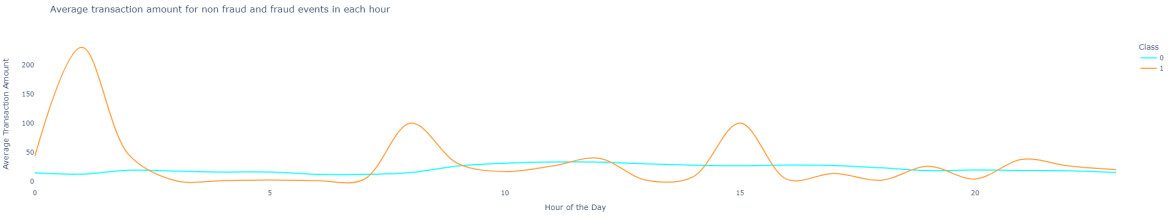

Figure 4.8: Average Fraud and Non-Fraud Transactional Amounts Across Hours

1. Overview:
   - Groups the credit card data by both 'Hour' and 'Class' columns, calculating the median transaction amount for each combination.

- Lines are differentiated by color for non-fraudulent (Class 0) and fraudulent (Class 1) transactions.

2. Purpose:

- The code aims to visually compare and analyze the average transaction amounts for non-fraudulent and fraudulent events across different hours of the day.
- The visual representation enhances the interpretability of the data, aiding in the exploration and communication of trends in average transaction amounts over the course of the day.

4.5.11 Histograms of Numerical Features

The fig 4.9 visualizes the numerical features in the dataframe by utilising the `hist()` function.

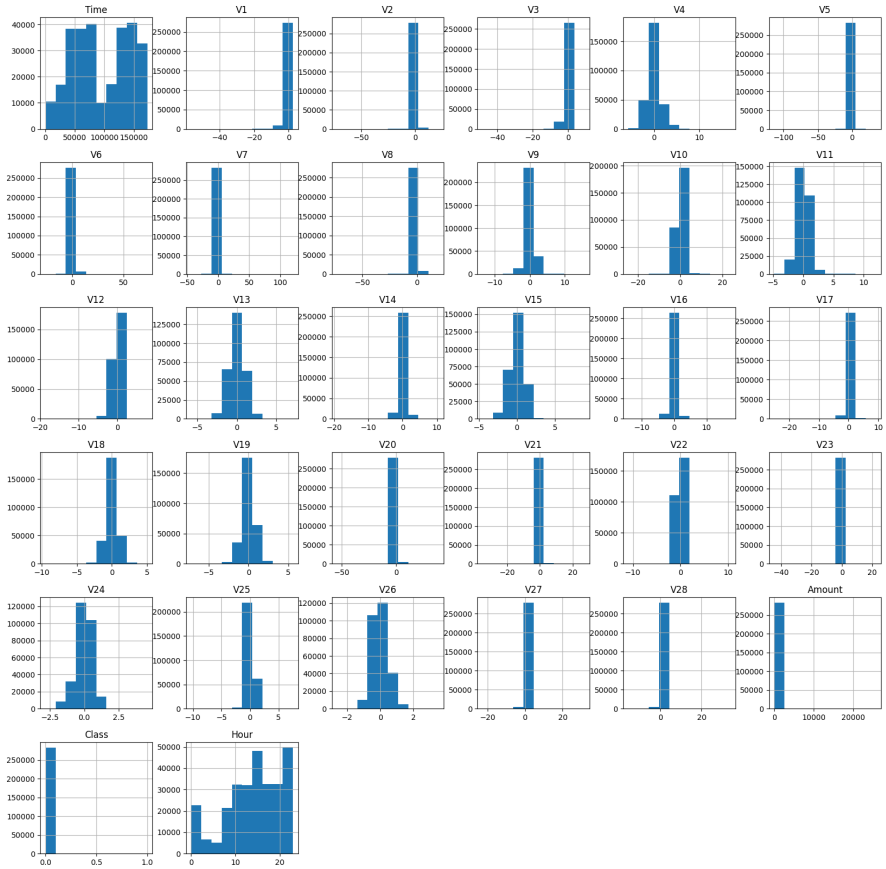

Figure 4.9: Histogram

1. Overview:

- The hist method is applied directly to the DataFrame `(credit_card)`.
- This function generates histograms for each numerical feature in the dataset.

2. Purpose:

- The purpose of this code is to provide a quick visual overview of the distribution of numerical features in the `credit_card` dataset.
- The larger figure size enhances visibility and detail in the visual representation.

4.5.12 Correlation Heatmap

The fig 4.10 represents the heatmap for visualization of the correlation matrix for the numerical features in the `credit_card` dataset.The heatmap visually represents the pairwise correlations between variables, offering insights into potential relationships. With a figure size of 12 by 9, the heatmap provides a clear overview of the strength and direction of correlations, aiding in the exploration of inter-feature connections within the dataset.

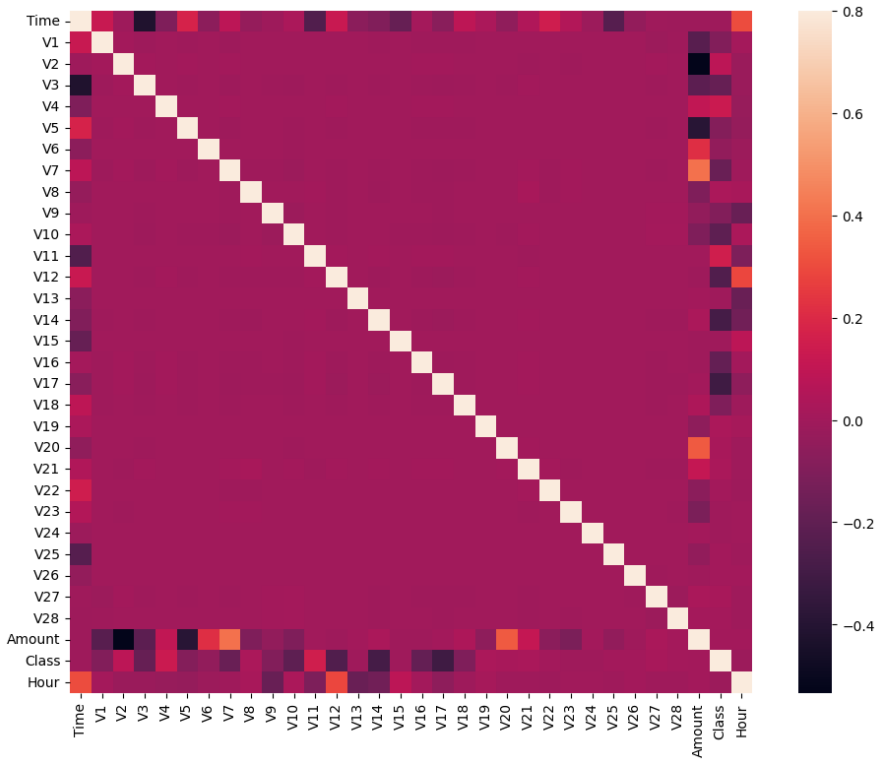

Figure 4.10: Correlation Heatmap

1. Overview:

- Computes the correlation matrix (`corr_matrix`) for numerical features in the `credit_card` dataset.
- Utilizes Seaborn's heatmap function to visually represent the correlation matrix.
- Adjusts the figure size to 12 by 9 for optimal visualization.

2. Purpose:

- The heatmap helps in identifying and visualizing the strength and direction of relationships between numerical features in the dataset.
- Provides a concise visual summary of correlations, highlighting potential patterns or associations.
- Supports the identification of highly correlated features, which can impact model performance and guide feature selection.
- Facilitates exploratory data analysis by revealing potential dependencies and insights within the dataset.
- Adjusting the figure size enhances visibility, making it easier to interpret and extract meaningful information from the correlation matrix.

4.5.13 Pair Plot Analysis

The fig 4.11 visualizes the pai-plot for a select subset of features from the `crecit_card` dataframe for a sample of 1000 data points.

1. Overview:

- Specifies a subset of features ('V2', 'V3', 'V4', 'V5', 'Class') to be included in the pair plot.
- Randomly samples 1000 data points from the `credit_card` dataset.
- Utilizes Seaborn's pairplot function to create a pair plot.

2. Purpose:

- The pair plot visually explores interactions between selected features, providing scatterplots for pairs of variables in the subset.
- Focusing on a subset of features allows for a more detailed examination of specific variables and their relationships.
- The pair plot provides insights into how the selected features contribute to the separation of classes, aiding in exploratory data analysis.

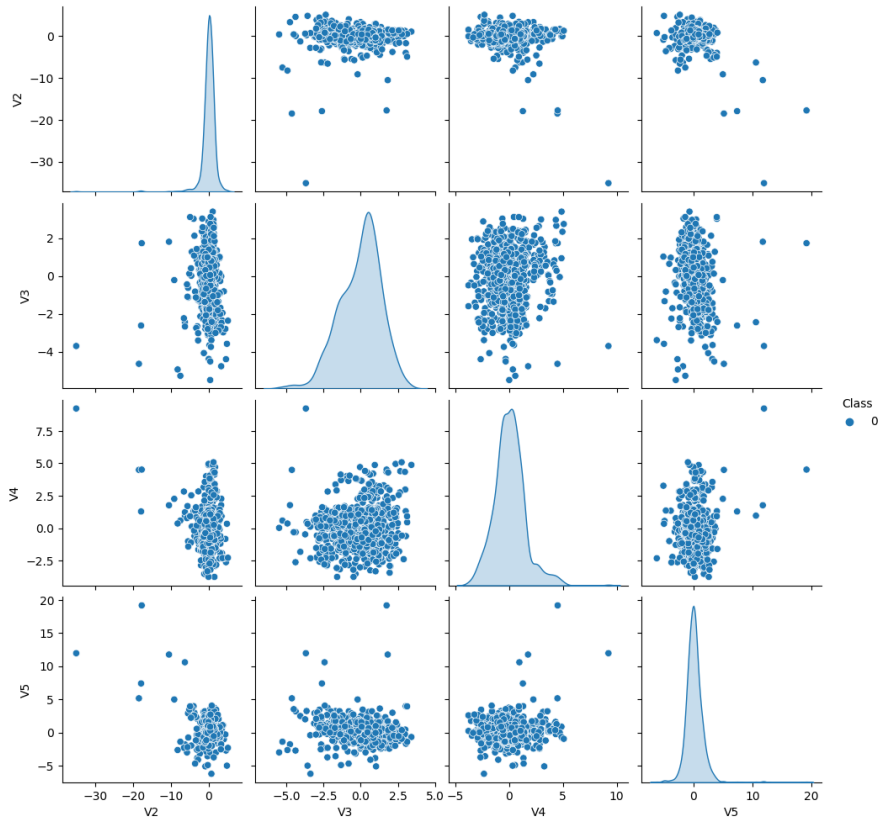

Figure 4.11: Pair-Plot

4.5.14 Insights from EDA

1. The dataset is highly imbalanced, as the class for non-fraud transactions constitutes 99.8% of the data
2. It can be clearly seen that the average time for fraudulent transactions is less than fraudulent ones.
3. Following this, average transaction amount is also less in fraudulent transactions compared to non-fraudulent ones.
4. The least number of transactions done are between 3am to 4am
5. The highest number of non-fraud transactions are done around 10pm
6. The highest number of fraud transactions are done at 1am and 11pm
7. Irrespective of fraud and non-fraud class, the minimum average transaction amount is 9.99K, around 5am to 7am. However, the highest average transaction amount is 35 around 10am

8. In case of fraudulent transactions, it can be clearly seen that the highest average amount of transaction is done around 11am with a value of 43. Furthermore, the average transaction amount fluctuates a lot in different hours of the day

9. In case of non-fraud transactions, the average transaction amount doesn't seem to have any noticeable fluctuations throughout the day.

4.6 Data Splitting

Data splitting is a fundamental practice in machine learning to assess the performance of models on unseen data. The process involves dividing the available dataset into two subsets: one for training the model and another for testing its performance. The primary goal is to ensure that the model generalizes well to new, unseen data. Here for this project the split is in the ratio of 80:20 , which means 80% of the data would be used to for the training set and the rest 20% will be used for the validation/testing set.

Chapter 5

PROPOSED MODEL

5.1 Description

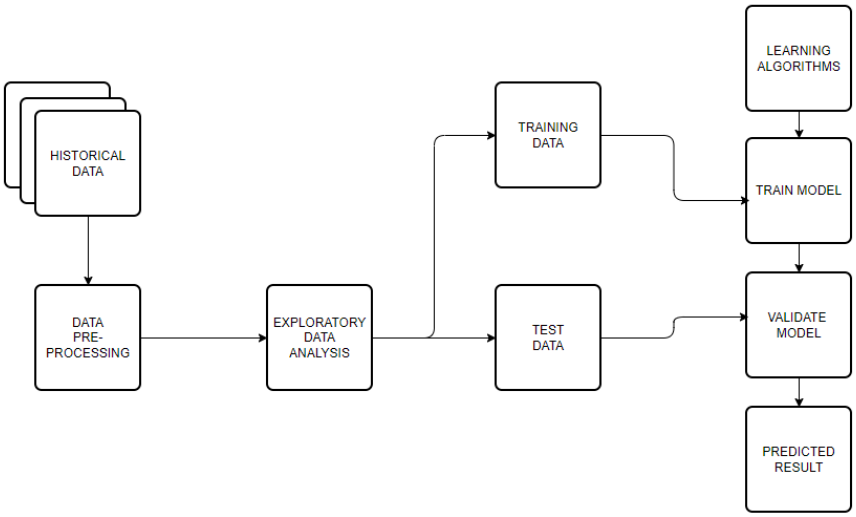

Figure 5.1: System Flowchart

In today's world of technological advancement, the transaction of amounts through credit card has increased by leaps and bounds.In order to keep in check whether the actions are perfect or not we also need a program or a model which can deal with every other obstacle.For such purpose a high-precision machine learning model is highly required. In our project we have dealt with a huge dataset of credit card transactions to build our model.Building a model as such has more issues in itself which needs to be checked and should be countered properly.The collected data must be preprocessed to ensure that it is clean, consistent, and in a suitable format for analysis.The preprocessing steps include removing null values,dropping

duplicated values and make the data more consistent and free of redundancy.We implemented many machine learning features to deal with it,like correlation matrix to understand variablity between different variables of the data.The dataset has many features like amount,time,class etc.to build the model the dataset is divided into teo sets training and testing sets after preprocessing.The ratio of partitioning is 8:2 i.e 80% for training set and rest 20% for testing set.This splitting technique is used for measuring the level of performance given out by the built machine learning models by implementing different algorithms.We have implemented multiple algorithm to give us the idea of which model would perform the best along certain different conditions.We used three algirthms to build the models namely - Random Forest Classifier Algorithm, Support Vector Machine & Logistic regression also.The results of predictions are then thoroughly analyzed. This said model is shown in the fig 5.1

5.2 Algorithms

5.2.1 Logistic Regression

Logistic regression is a kind of classification model, which learn and predict the parameters in the given dataset using regression analysis. In logistic regression, the input features are utilised to compute an input weighted sum, which is then passed through a logistic function to determine the likelihood of the binary outcome. Any real-valued input is translated by the logistic function into a value between 0 and 1, using an S-shaped curve. Logistic regression model requires class variable that should be binary classified. Likewise, in this dataset the ”target” column has the two type of binary numbers, ”0” for the transaction which has less chances of being non-fradulent, and ”1” for the transactions which has more chance of being fraudulent. The independent variables, on the other hand, can be polynomial, nominal, or binary classified.. The logistic regression equation can be written as:

$$P(y = 1|x) = \frac{1}{1 + e^{-z}}$$

The ’LogisticRegression’ class from sklearn library is used for implementing logistic regression and it provides several options for regularization, which can help to prevent overfitting and improve the generalization performance of the model.

5.2.2 Support Vector Machine

SVMs or Support Vector Machine are methods,is a basic yet effective Supervised Machine Learning approach that may be used to generate both regression and classification models. The SVM method works effectively with both linearly and non-linearly separable datasets.In Python, we can use libraries like sklearn. For classification, Sklearn provides functions like SVC, NuSVC & LinearSVC.Here in our project we have used `SVC` library.Due to their great capacity to generalise to new unseen data objects, lack of local minima, flexible non-linear decision boundary, and dependency on a small number of hyper-parameters,SVM models are commonly utilised in classification tasks.The SVM algorithm's purpose is to find the optimal line or decision boundary for categorising n-dimensional space so that we may simply place fresh data points in the proper category in the future. A hyperplane is the optimal choice of boundary.SVM selects the extreme points/vectors that aid in the creation of the hyperplane. These extreme examples are referred to as support vectors, and the method is known as the Support Vector Machine.

5.2.3 Random Forest

Leo Breiman and Adele Cutler created the Random Forest machine learning algorithm, which combines the output of many decision trees to produce a single conclusion. Its ease of use and versatility, as well as its ability to tackle classification and regression challenges, have boosted its popularity.A popular approach for regression and classification is Random Forest. We may claim that the Random Forest Algorithm is one of the most crucial algorithms in machine learning, as classification and regression are the two most essential components of machine learning.The ability to accurately categorise observations is useful to detect fraud and also for a variety of commercial applications, such as forecasting a user's propensity to purchase a product or to fail on a loan.To put it simply, Random Forest is a classifier that builds many decision trees on different dataset subsets and averages them to improve the dataset's projected accuracy. The random forest predicts the final result based on the majority votes of projections, gathering results from several decision trees rather than depending on just one.We have implemented the Random Forest algorithm in our model by utilising the RandomForestClassifier class from sklearn library.The random forescast algorithm is an extension of the bagging method as it utilizes both bagging and feature randomness to create an uncorrelated forest of decision trees.For this algorithm it's easy to determine the feature importance and are used to measure model accuracy.

5.3 Experimental Setup

5.3.1 Import Libraries

1. **NumPy:** NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

2. **Matplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.It is a cross platform which has a variety of usage like data plot,graphical plotting(histograms,scatter plots,bar charts) Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, Python/IPython shells, web application servers, and various graphical user interface toolkits.

3. **Pandas:** This is a versatile and powerful open-source library for Python utilised for data manipulation and analysis.Pandas is built on top of two core Python libraries—matplotlib for data visualization and NumPy for mathematical operations. Pandas acts as a wrapper over these libraries, allowing you to access many of matplotlib's and NumPy's methods with less code. For instance, pandas' .plot() combines multiple matplotlib methods into a single method, enabling you to plot a chart in a few lines. It provides a high-level, easy-to-use features for working with structured data, primarily in the form of tabular data called DataFrames. With Pandas, we can efficiently load, clean, transform, and analyze data. It offers functionalities for data indexing, selection, filtering, grouping, and statistical analysis.

4. **Scikit learn:** scikit-learn is a Python module for machine learning built on top of SciPy and is distributed under the 3-Clause BSD license.Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.The goal of the library is to reach a support and robustness level necessary for use in production systems. This entails giving serious consideration to issues like performance, documentation, collaboration, code quality, and ease of use.

5. **Plotly:** Plotly Express, or PX, is the name given to the plotly.express module (typically imported as px), which provides functions capable of creating whole figures at once. The suggested place to start when making the majority of popular figures is Plotly Express, which is an integrated component of the plotly library.Plotly express is a

high-level data visualization package that allows you to create interactive plots with very little code.

6. **Imbalanced Learn :** This is a Python library designed to address the challenges posed by imbalanced datasets in machine learning. Imbalanced datasets occur when the distribution of classes in the target variable is skewed, leading to issues in training models that may be biased towards the majority class. Imbalanced-learn provides various techniques for handling imbalanced data, including oversampling the minority class, undersampling the majority class, and generating synthetic samples using methods like SMOTE (Synthetic Minority Over-sampling Technique).

7. **Seaborn :** Seaborn (styled as "seaborn") is an open-source Python library used for visualizing the explorative statistical plots of data. It is a basic library used to extract various information that can provide an understanding of a dataset.It is built on top of matplotlib and is used for informative and easily uderstandable statistical graphics.

5.3.2 Model Evaluation

1. **Logistic Regression :** One type of supervised learning method is the logistic regression algorithm.This model is used for a binary classification problem in our project.Here, the procedure starts with importing the scikit-learn library's logistic regression class. The fit technique is then used, using the inputs as `X_train` for training features and `y_train` for matching labels, because the model is now prepared for training.Following the training phase, the model is given the responsibility of generating predictions using the testing data, which is a different collection of data.Here, the testing features `(x_test)` are subjected to the predict algorithm, which yields a set of predicted labels that are saved in the `lg_pred` variable. The practical significance of this process lies in its ability to generate predictions for new, unseen data based on the learned patterns from the training set. The performance of the model can be evaluated by comparing these predictions `(lg_pred)` to the true labels of the testing set `(y_test)`, enabling the assessment of metrics such as accuracy, precision, recall, and others.This process's capacity to forecast fresh, unknown data based on the patterns it has learnt from the training set is what gives it practical relevance. Metrics like accuracy, precision, recall, and others may be assessed by comparing these predictions `(lg_pred)` to the real labels of the testing set `(y_test)`. This allows for an evaluation of the model's performance.

2. **Support Vector Machine :** With a linear kernel chosen in this instance, support vector machines (SVMs) are robust methods for both linear and non-linear classification. The Support Vector Classification (SVC) class from scikit-learn is used to implement the

SVM, and the regularisation parameter 'C' is set at 0.1. Similar to before, the model is trained using the training features `X_train` and matching labels `y_train` through the use of the `fit` technique.At this stage, the algorithm optimises its parameters to discover the hyperplane.After training, the model is used using the 'predict' technique on the testing features `X_test`. The `svc_pred` variable houses the final predictions. Metrics like accuracy, precision, recall, and others may be used to assess the model's performance by comparing these predicted labels with the real labels `y_test`.

3. **Random Forest :** The Random Forest algorithm is a versatile ensemble learning method that combines the predictions of multiple decision trees to enhance overall predictive accuracy and generalization. The `RandomForestClassifier` is initialized with specific settings: it will consist of 100 decision trees (`n_estimators=100`), and a random seed (`random_state=42`) is set to ensure reproducibility of results. Next, the Random Forest classifier undergoes training using a resampled set of training data. Resampling is a technique commonly used to handle imbalances in class distribution within the training dataset. Here, `X_train_res` contains the features of the resampled training set, while `y_train_res` holds the corresponding labels. During training, the algorithm learns intricate patterns and relationships within the data, benefiting from the collective insights of multiple decision trees. After training, the classifier is applied to new, unseen data represented by `X_test`. The `predict` method is employed to generate predicted labels, stored in the variable `y_pred`. These predicted labels are then compared with the true labels of the test set, allowing for the evaluation of the classifier's performance and its ability to generalize knowledge from the training data to previously unseen instances. This process is crucial in assessing the model's effectiveness in making accurate predictions on new data.

5.3.3 Model Tuning

Model tuning is the experimental process of finding the optimal values of hyperparameters to maximize model performance. The purpose of tuning a model is to ensure that it performs at its best. This process involves adjusting various elements of the model to achieve optimal results. Here we have used K-fold cross validation in the context of model tuning provide a precise and fair estimate of model's performance across different subsets of the data.With this technique it involves dividing the dataset into k subsets or folds,and the model is trained and validated k times, each time using a different fold as the validation set.

In summary, model tuning using k-fold cross-validation is a systematic approach to finding the optimal hyperparameter configuration for a machine learning model. By

leveraging multiple validation sets and evaluating performance metrics, this process ensures that the selected model is both accurate and robust across diverse subsets of the data, facilitating improved generalization to unseen instances.

5.3.4 Statistical Significance Test

The objective of a statistical significance test is to determine whether there is a statistically significant difference in mean standardized transaction amount for normal(class = 0) and fraudulent (class = 1)credit card transactions.In simple terms,the observed values are distinct are real and not due to variability or randomness. Here we have performed the t-test on two independent sample of two subsets of the data.The t-test is a common statistical method for comparing means, and its results provide insights into the potential significance of differences between the two classes.

Chapter 6

RESULTS AND ANALYSIS

6.1 Classification Metrics

6.1.1 Confusion Matrix

One approach for assessing a classification model's performance is a confusion matrix. It contrasts the machine learning model's projected values with the actual target values. In our use case, we are attempting to forecast the quantity of credit card transactions that are fraudulent and those that are not.The Test dataset, which makes up 20% of the entire dataset, is what we will use to test our model. The suggested Models will attempt to forecast these, and the confusion matrix will give the full classification report. We can compute the following crucial classification metrics with this tool: recall, accuracy, precision, and F1 score.

Accuracy: It is the ratio of the number of correct predictions to the total number of input samples. It works well only if there are equal number of samples belonging to each class.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

Precision(Weighted): It is defined as the proportion of accurately anticipated positive observations to all positive predictions. It makes sense that the classifier's precision is its capacity to reject labelling a negative sample as positive. The weighted approach determines the average of the metrics for each label, weighted by the total number of true cases for each label.

$$\text{Precision (Weighted)} = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i \cdot \frac{T_{Pi}}{T_{Pi} + F_{Pi}}$$

where w_i represents the number of true occurrences for each label, n represents the number of classes, TP_i represents the number of true positives for each class, and FP_i represents the number of false positives for each class.

Recall(Weighted): It is the proportion of accurately predicted positive observations to the total number of observations in the class. The recall is intuitively the classifier's capacity to locate all positive samples. The weighted technique computes metrics for each label and averages them by the number of true cases for each label.

$$\text{Recall (Weighted)} = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i \cdot \frac{T_{Pi}}{T_{Pi} + F_{Ni}}$$

where w_i is the number of true instances for each label, n is the number of classes, TP_i is the number of true positives for each class, and FN_i is the number of negatives for each class.

F1 Score(Weighted): The F1 score may be viewed as a weighted average of accuracy and recall, with a greatest value of 1 and a worst value of 0. The weighted technique computes metrics for each label and gets the average of those metrics weighted by the number of true cases for each label.

$$\text{F1 Score (Weighted)} = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i \cdot \frac{2.(precision_i.recall_i)}{precision_i + recall_i}$$

where w_i is the number of true instances for each label, n is the number of classes,$Precision_i$ is the precision for each class and $Recall_i$ is the recall for each class

6.2 Analysis of Results

In this section all the experimental results of different algorithms applied while building the model have been discussed along with all others takeaways from the previous works.

6.2.1 ROC AUC Metrics

The ROC curve and the AUC Curve are two extensively used graphical measures for evaluating the performance of binary classification algorithms.For various threshold settings, it plots the True Positive Rate (Sensitivity) vs the False Positive Rate (1 - Specificity). The area under the ROC curve (AUC) is a single scalar statistic that summarises the model's overall performance across various threshold settings.

The below table records all the ROC AUC Score of different algorithms use

Algorithms	ROC AUC Scores
Logistic Regression	0.77
Support Vector Machine	0.87
Rndom Forest	0.89

Table 6.1: ROC AUC Metrics

6.2.2 Confusion Matrix Analysis

1. **Logistic Regression**: The figure 6.1 shows the values of the Confusion Matrix of all

Figure 6.1: Confusion Matrix for Logistic Regressor

possible predicted outcomes for Logistic Regression. The values along the diagonal

from top left to bottom right represents the samples which are correctly classified. Rest of the values represent miss classification.

2. **Support Vector Machine**: The figure 6.2 shows the values of the Confusion Matrix of

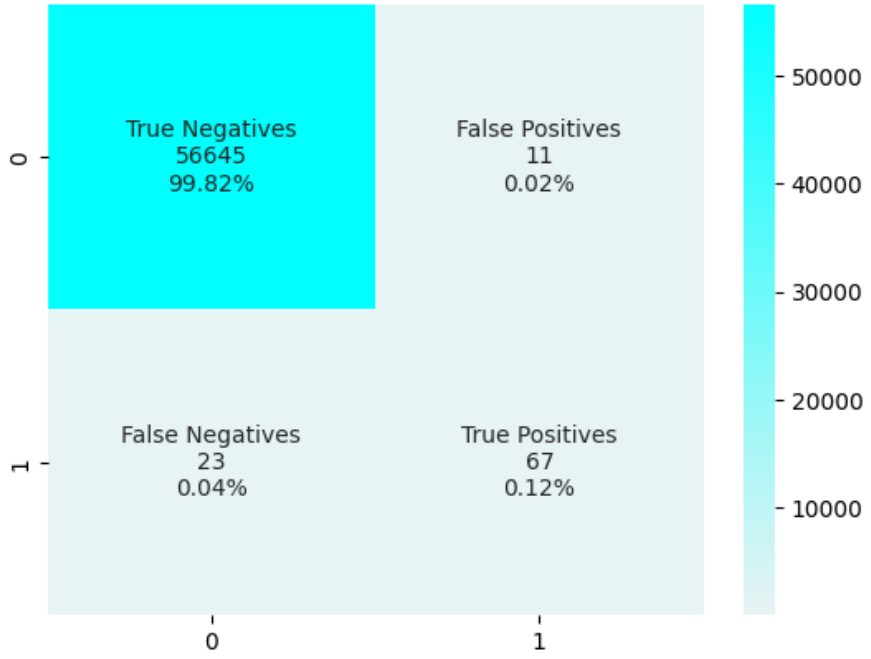

Figure 6.2: Confusion Matrix for SVM

all possible predicted outcomes for SVM. The values along the diagonal from top left to bottom right represents the samples which are correctly classified. Rest of the values represent miss classification.

3. **Random Forest**:

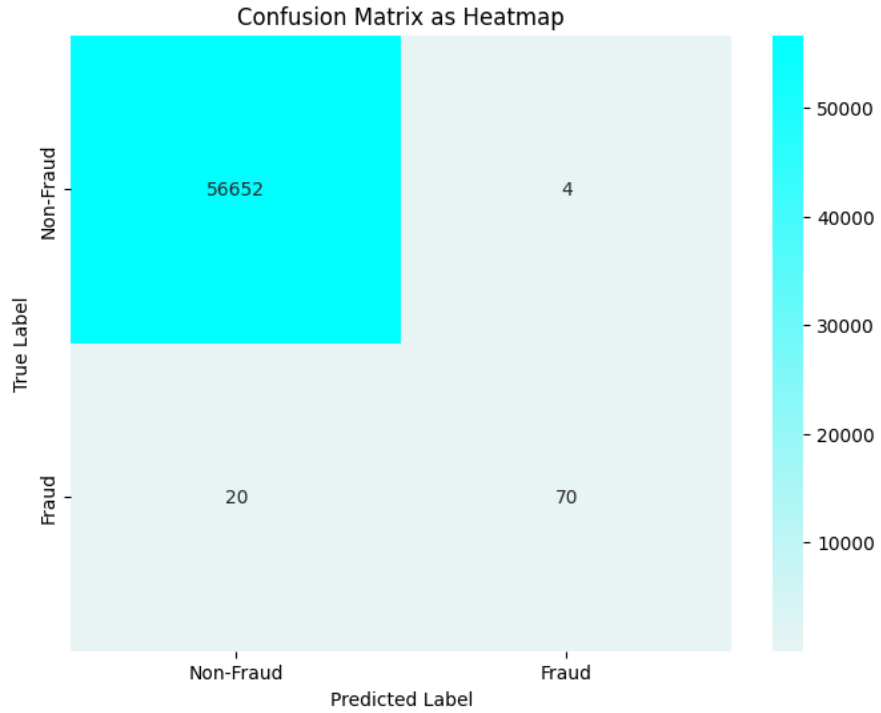

Figure 6.3: Confusion Matrix for Random Forest

The figure 6.3 shows the values of the Confusion Matrix of all possible predicted outcomes for Random Forest. The values along the diagonal from top left to bottom right represents the samples which are correctly classified. Rest of the values represent miss classification.

6.2.3 Tuning Models Results Analysis

K-Folds Cross Validation The k-fold cross-validation is a crucial step in optimizing the performance and ensuring robustness of a machine learning model. Cross-validation is a technique used to assess how well a model will generalize to an independent dataset. The "k" in k-fold cross-validation represents the number of folds or subsets that the dataset is split into. K-fold Cross-Validation with K = 5, was used for all the models and the validation accuracy across all are the folds is given in this chapter along with that the average cross-validated F1-Score is also provided.

1. **Logistic Regression** The following table 6.2 shows the results which are obtained by performing k-fold cross validation on SVM :-

No. of Folds(K)	F1 Score
1st	0.8569443
2nd	0.8092591
3rd	0.87390768
4th	0.87294681
5th	0.87575785

Table 6.2: K-Folds for Logistic Regression

The average K-Fold cross validated F1-Score for Logistic Regression is **0.8577631525920651**.

2. **Support Vector Machine** The following table 6.3 shows the results which are obtained by performing k-fold cross validation on SVM :- The average K-Fold cross validated

No. of Folds(K)	F1 Score
1st	0.91081549
2nd	0.86965341
3rd	0.91757808
4th	0.91536058
5th	0.8985218

Table 6.3: K-Folds for SVM

F1-Score for SVM is **0.9023858727966685** .

3. **Random Forest** The following table **??** shows the results which are obtained by performing k-fold cross validation on Random Forest :- The average K-Fold cross

No. of Folds(K)	F1 Score
1st	0.99985657
2nd	0.9999338
3rd	0.99987864
4th	0.99991174
5th	0.99987864

Table 6.4: K-Folds for Random Forest

validated F1-Score for Random Forest is **0.9998918785026693** .

Thus from the above table 6.2 to table 6.4 it is clear that Random Forest is having the highest F1-Score among all i.e **0.9998918785026693** .

6.2.4 Acurracy Result Analysis

The table‘6.5 below provides the accuracy level for all algorithms implemented till now.

Classifier	Accuracy(%)
Logistic Regression	99.92
Support Vector Machine	99.94
Random Classifier	99.96

Table 6.5: Accuracy Details

6.2.5 Model Comaprison

The bar graph‘6.4 below shows the different accuracy of all the algorithms implemented.

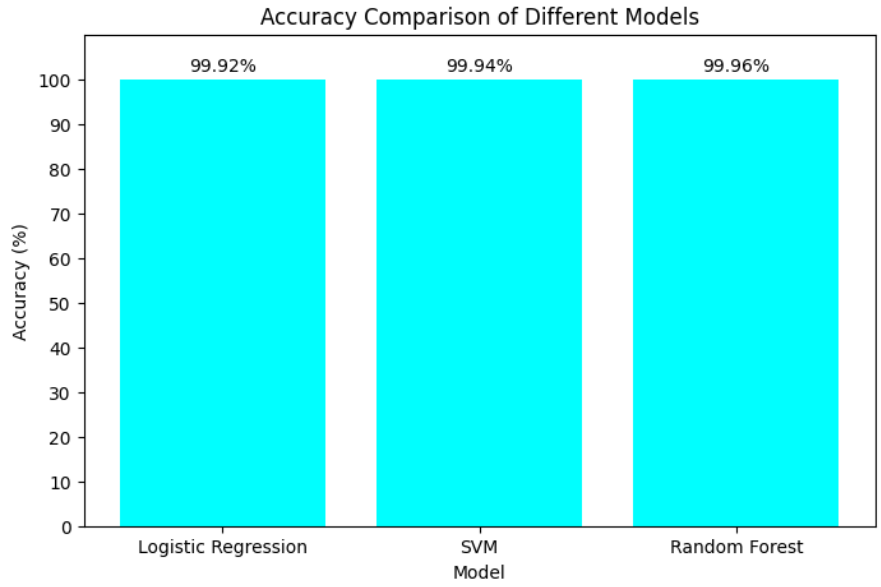

Figure 6.4: Accuracy Barplot for different models

From table 6.5 and figure 6.4, it can be concluded that Random Forest Classifier is the best model as it is having the highest accuracy, So we applied this model in test data for prediction. It will give 99.96% accuracy in the testing data set.

6.2.6 Statistical Significance Test Result Analysis

The p-value obtained after performing a statistical significance test is a crucial metric in hypothesis testing. In our case the P-value obtained after performing **statistical significance test** was **0.0020895802939985045** which is very less than 0.05.

Thus it means that :-

- The p-value is less than the significance level (0.05), suggesting that there is sufficient evidence to reject the null hypothesis.
- This implies that the observed results are unlikely to have occurred by random chance alone.

Chapter 7

CONCLUSION & FUTURE SCOPES

7.1 Consclusion

Without a doubt, credit card fraud is a crime, and cases of fraudulent transactions involving credit cards are on the rise globally. In this report, we attempt to develop models using supervised machine learning algorithms in order to distinguish between fraudulent and non-fraudulent transactions from a dataset obtained from Kaggle. After reading through a number of publications, we decided to employ three algorithms—Logistic Regression, Support Vector Machine, and Random Forest—to create our models. After successfully running the models, various results are achieved. According to the results of the experiments,the Random Forest Classifier had the highest validated F1 Score 0.998918785026693 & highest Accuracy of 99.96%, After the comparison of the three models, Statistical Significance Test was done & the P-value obtained was 0.00208958029399985045 which rejected the null hypothesis & implied that the observed results were unlikely to have occured by chance. Other than that the logistic regression model's accuracy was the lowest among the three.Thus we can conclude that use of supervised Machine Learning algorithms in credit card fraud detection gives fruitful results.

7.2 Future Scopes

This project allows for the integration of multiple algorithms (*Neural Networks*, *Naive Bayes*, *K–Nearest Neighbour*) as modules, and the results of these can be combined to get a higher accuracy, which makes it more versatile. However, in order to obtain more detailed and

real-time data, official support from banks is required. For that reason, we have developed methods to enhance the explainability interpretability of the model's decisions. By improving the explainability interpretability, we can gain the desired level of accuracy. banks, users, stakeholders. The Random Forest Classifier, which yielded the best accuracy in our research, may be used or incorporated into an application in the future to enable users to independently verify transactions to be fraudulent.

Bibliography

[1] The dataset that is used in the report is taken from *Kaggle, Anonymized credit card transactions*,2017.

[2] Aditya Saini, Swarna Deep Sarkar, Shadab Ahmed, "Credit Card Fraud Detection using Machine Learning and Data Science", *International Journal of Engineering Research & Technology (IJERT)*, Vol. 8 Issue 09, September-2019.

[3] Meera AlEmad, "Credit Card Fraud Detection Using Machine Learning", *Rochester Institute of Technology(RIT) Scholar Works, Dubai*, https://scholarworks.rit.edu/theses July-2022.

[4] Clifton Phua, Vincent Lee, Kate Smith & Ross Gayler “A Comprehensive Survey of Data Mining-based Fraud Detection Research”, *School of Business Systems, Faculty of Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, Australia*.

[5] Emmanuel Ileberi, Yanxia Sun & Zenghui Wang "A machine learning based credit card fraud detection using the GA algorithm for feature selection", *Journal of Big Data*, Article 24, February-2022.

[6] Vaishnavi Nath Dornadula, Geetha S, "Credit Card Fraud Detection using Machine Learning Algorithms", *INTERNATIONAL CONFERENCE ON RECENT TRENDS IN ADVANCED COMPUTING(ICRTAC)*, 2019.

Biographical Sketch

First Student's Name

- Sahil Dey, Melarmath, Agartala pin: 799001
- E-mail: deysagar3001@gmail.com ,Contact no: 9863082229
- Pursuing B.Tech. in Computer Sc. & Engg. branch from N.I.T,Agartala with CPI of 8.25/10.00.
- Completed Class 10 from Umakanta Academy English Medium,Agartala under (T.B.S.E), Tripura with 76% in 2017.
- High School from Shishu Bihar H.S School,Agartala under (T.B.S.E), Tripura with 81% in 2019.

Second Student's Name

- Shubhadeep Nath, Dhaleswar, Agartala pin: 799007
- E-mail: shubhadpn1927@gmail.com, Contact no: 7005190502
- Pursuing B.Tech. in Computer Sc. & Engg. branch from N.I.T,Agartala with CPI of 8.52/10.00.
- Completed Class 10 from Shishu Bihar H.S School,Agartala under (T.B.S.E), Tripura with 87.57% in 2017.
- High School from Shishu Bihar H.S School,Agartala under (T.B.S.E), Tripura with 87.4% in 2019.

Third Student's Name

- Sourajita Chanda,Joynagar,Agartala pin: 799001
- E-mail: sourajitachanda8@gmail.com , Contact no: 7005813692
- Pursuing B.Tech. in Computer Sc. & Engg. branch from N.I.T,Agartala with CPI of 8.65/10.00.
- Completed Class 10 from Shishu Bihar H.S School,Agartala under (T.B.S.E), Tripura with 81.14% in 2017.
- High School from Shishu Bihar H.S School,Agartala under (T.B.S.E), Tripura with 75.8% in 2019.

Fourth Student's Name

- Tithi Majumder,Ker Chowmnani,Agartala pin: 799001
- E-mail: tithimajumder717@gmail.com , Contact no: 8787672956
- Pursuing B.Tech. in Computer Sc. & Engg. branch from N.I.T,Agartala with CPI of 8.66/10.00.
- Completed Class 10 from Shishu Bihar H.S School,Agartala under (T.B.S.E), Tripura with 84% in 2017.
- High School from Shishu Bihar H.S School,Agartala under (T.B.S.E), Tripura with 81% in 2019.