Credit Card Fraud Detection

TABLE OF CONTENTS

S No	Topic	Page No
1.	Chapter-1: Introduction	1-5
2.	1.1 General Introduction	1-2
3.	1.2 Data Collection	2-3
4.	1.3 Phases of Analysis	3-4
5.	1.4 Tools/Platforms	4
6.	Chapter-2: Literature Review	6-8
7.	2.1 Summary of Paper Studies	6-7
8.	2.2 Integrated Summary of the Literature Studied	7-8
9.	Chapter-3: Implementation and Results	9-20
10.	Chapter-4: Implementation and Visualization	21-25
11.	Chapter-5: Conclusion and Future Work	26-28
12.	References	28

LIST OF TABLES

Table No	Title	Page No
1.1	Hardware Specifications	4
1.2	Software Specifications	4
5.1	Summary Table	21

LIST OF FIGURES

Figure No	Title	Page No
1.1	Block Diagram	3
3.1	Transaction Class Distribution	11
3.2	Amount per transaction by Class	12
3.3	Time of transaction vs Amount by class	13
5.1	F1 scores of different models	27

Chapter-1

Introduction

1.1. General Introduction

1.1.1. Description of topic under analysis

Credit Card frauds are a growing concern in today's modern world. It is a serious issue that affects millions of people every year. In order to prevent fraudulent activities, it's essential to have an effective and accurate fraud detection system in place.

However, detecting credit card fraud is challenging due to the vast volume of transactions and the evolving tactics employed by fraudsters. Traditional methods of detecting fraud are no longer enough to keep up with the ever-evolving techniques used by criminals. To combat this, banks and other financial institutions are turning to machine learning since it offers a promising approach to identify such fraudulent transactions by learning patterns and anomalies from historical data. By using python and machine learning algorithms, these organisations can analyse large amounts of data quickly and accurately to identify suspicious activity. It uses various features of credit card transactions, such as transaction amount, merchant location, time of transaction and user behaviour patterns etc.

This allows these financial institutions to take preventive measures to protect their customers from potential financial losses by raising alerts or blocking transactions that are likely to be fraudulent, providing an additional layer of security for both card holders and banks.

1.1.2. Problem Statement

Leverage machine learning algorithms and techniques to build a predictive model that can detect and prevent fraudulent activities.

1.1.3. Intended Operations to be performed

The intended operation performed by a Credit Card Fraud Detection System is to detect whether a particular transaction is fraud or not. The design of a effective Fraud detection system is necessary in order to reduce the losses incurred by the people by frauds. A good fraud detection system should be able to identify the fraud transactions accurately and should make the best decision of the message.

The operations involves data pre-processing, feature engineering and applying different Machine Learning algorithms to find the best model.

1.1.4. End Users

The analysis can be used by banks, financial institutions, people and cyber security officials.

1.2. Data Collection

Data collection is the act of obtaining and analysing information on certain variablesin a predetermined, methodical way so that one may subsequently analyse results and respond to pertinent queries.
A rigorous procedure for gathering data is required because it assures that the information is specified, correct, and that judgements made in the future based on the results are sound. The procedure offers a starting point for comparison and, in certain circumstances, a suggestion of where improvements might be made.

- Inability to provide reliable answers to research questions is one effect of faulty data collection.
- Unable to validate and replicate the study.
- The results of distorted research squander resources and may drive other researchers to pursue futile lines of inquiry. They may also jeopardise judgements, such as those pertaining to public policy, which may result in disproportionate harm.
- As a result, the aim of data collection is to gather reliable information that enables analysis to result in the development of solutions that are believable and persuasive to the given questions.
- Despite the fact that there are many ways to get data, the researcher choose to focus on two key sources in their study:
- Primary source: The term "primary source" refers to the sources used by researchers to get original data when using an empirical methodology like a personal interview.
- Secondary source: The importance of secondary data sources for this type of endeavour cannot be overstated.
- Here, the data gathering was done using secondary source, a website named Kaggle.

1.3. Phases of Analysis

1.3.1. Block Diagram

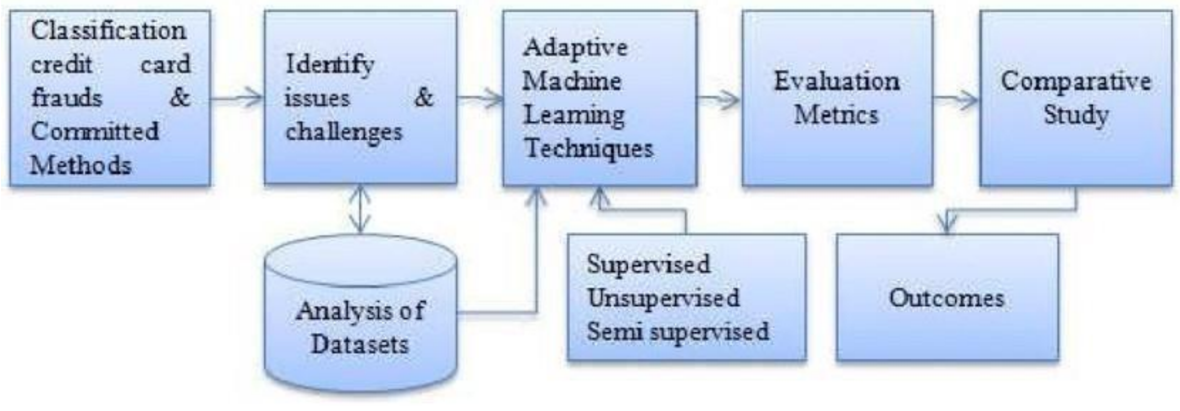

Classification credit card frauds & Committed Methods
Identify issues & challenges
Adaptive Machine Learning Techniques
Evaluation Metrics
Comparative Study
Analysis of Datasets
Supervised Unsupervised Semi supervised
Outcomes

Fig 1.1: Block Diagram

1.3.2. Attributes considered for studying

The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions.

It contains only numerical input variables which are the result of a PCA transformation.

Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Hence there are 29 feature columns and 1 final class column.

1.4. Tools/Platforms

1.4.1. Hardware Specifications

Processor	Core i5
RAM	8.00 GB
Memory	512 GB
System type	64-bit operating system, x64-based processor

Table No 1.1: Hardware Specifications

1.4.2. Software Specifications

OS	Windows 11
Language	Python
Software Development Kit	Jupiter Lab

Table No 1.2: Software Specifications

1.4.3. Packages to be imported

Packages that we imported in this project are:

- NumPy
- Pandas
- Matplotlib
- Seaborn
- Sklearn
- XGBoost

Chapter-2: Literature Review

2.1. Summary of Paper Studies

[1] Enhancing the Credit Card Fraud Detection Using Decision Trees and Adaptive Boosting Techniques

The approach proposed in the study uses machine learning methods, in particular Decision Tree with Adaptive Boosting, to identify fraudulent transactions in the mobile and e-commerce industries. The suggested solution tries to improve the security and integrity of credit card transactions by balancing the data using SMOTE and utilising performance measurements.
The article analyses the findings based on measures for accuracy, precision, and recall to assess the effectiveness of the models. These metrics are frequently used to judge how well fraud detection systems are working.

[2] A Hybrid Convolutional Neural Network and Support Vector Machine-Based Credit Card Fraud Detection Model

In this study, a hybrid CNN-SVM model was suggested as a useful method for identifying credit card transaction fraud. The model obtains promising classification performance metrics by fusing the advantages of CNN and SVM approaches, highlighting its potential for enhancing credit card fraud detection systems.
The experimental findings show how well the hybrid CNN-SVM model performs in identifying credit card fraud. High values of 91.08%, 90.50%, 90.34%, 90.41, and 91.05% for the model's accuracy, precision, recall, F1-score, and AUC (Area Under the Curve) metrics, respectively, are achieved.

[3] Finding Credit Card Fraud from the Imbalanced Data via Deep Neural Networks

This study offers a unique method for detecting credit card fraud in the financial industry, placing emphasis on the value of include both transactional and user information. Based on the experimental findings, HNN-CUHIT outperforms other machine learning models in the identification of fraud as well as solutions for unbalanced classes.
Real-world data from a city bank during the SARS-CoV-2 epidemic in 2020 is used to assess how well HNN-CUHIT performs in detecting credit card fraud. HNN-CUHIT's effectiveness is contrasted with that of other techniques. In the fraud detection experiment, HNN-CUHIT outperforms LR, RF, and CNN with F1-scores

of 0.0360, 0.0284, and 0.0396, respectively, and obtains an F1 -score of 0.0416. HNN-CUHIT outperforms CNN by ROS with an F1-score of 0.0572 in the unbalanced class solution exercise.

[4] Enhanced Credit Card Fraud Detection: A Novel Approach Integrating Bayesian Optimized Random Forest Classifier with Advanced Feature Analysis and Real-time Data Adaptation

This study aids in the creation of trustworthy and effective fraud detection technologies for the banking sector. For fraud analysts and investigators, the suggested Bayesian optimised random forest classifier may be used as a decision support tool. Through an examination of the significance of characteristics in fraud detection, the study also investigates the interpretability of the model. This research improves the proposed approach's openness and interpretability by offering useful insights into the underlying factors that lead to fraud detection.

2.2. Integrated summary of the Literature studied

The cardholder can make purchases and charge them to the connected account using their credit card, which is a compact, thin plastic or fibre card that includes personal information. It's a financial service provided by banks that offers a fixed credit limit for easy cashless transactions. The customer's credit score, credit history, and income are taken into consideration when determining the credit limit. Credit card holders have the choice of paying off their amounts in full by the due date or gradually over a period of time, including with any relevant interest and fees.

With the development of technology, credit cards have gained popularity as a method of payment, with many individuals choosing online services and transactions such as e-commerce, tap-and-pay, and online bill payment.

Credit cards do, however, come with certain difficulties, especially in terms of security and fraud. For banks throughout the world, protecting card payments and fostering customer trust in using credit cards for transactions are top priorities. Credit card fraud is becoming increasingly difficult for financial organisations to detect, according to reports.
The researchers have utilised supervised and unsupervised machine learning algorithms to identify fraud during the past few years. The supervised method employs machine learning techniques to categorise, such as logistic regression, random forests, support vector machines,and neural networks, among others, and labels all nodes as belonging to distinct classes. The unsupervised technique uses node properties, such as K-means, PCA, GAN, etc., to directly classify nodes without labelling any of them. Furthermore, the limitation of the machine learning algorithm influences the prediction accuracy in the real-world data environment for the datasets made publicly available online but particularly altered owing to secrecy and privacy.

Chapter-3

Implementation and Results

3.1. Phase 1: Data Collection and importing libraries

The dataset is extracted from Kaggle. The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions. It contains only numerical input variables which are the result of a PCA transformation. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

PCA Transformation

PCA (Principal Component Analysis) is a dimensionality reduction technique commonly used in machine learning and data analysis. It aims to transform a high-dimensional dataset into a lower-dimensional space while retaining the most important information or patterns in the data.

The main idea behind PCA is to find the directions, called principal components, along which the data varies the most. These principal components are orthogonal to each other and ranked by the amount of variance they explain in the original data.

The resulting transformed data will have reduced dimensions, with each dimension being a linear combination of the original features. The reduced dimensions are ordered in terms of the amount of variance they explain, with the first dimension (principal component) explaining the most variance and subsequent components explaining decreasing amounts.

PCA can be useful for various purposes, such as data visualization, noise reduction, feature extraction, and improving the performance of machine learning algorithms. By reducing the dimensionality of the data, PCA can help alleviate the curse of dimensionality, remove redundant information, and focus on the most relevant aspects of the data.

It's important to note that PCA is an unsupervised technique and does not take into account any class labels or target variables. If you have labeled data and want to perform dimensionality reduction while considering class separability, you may consider other techniques such as Linear Discriminant Analysis (LDA).

The libraries imported are:

- NumPy

  NumPy (Numerical Python) is a Python library that provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. It is one of the fundamental libraries in Python for scientific computing and data analysis.
  It is a powerful library for numerical computing in Python. It provides efficient data structures for handling large arrays, along with a comprehensive set of mathematical functions. NumPy is widely used in various fields, including data analysis, machine learning, image processing, simulation, and more.

- Pandas

  Pandas is a popular open-source Python library used for data manipulation and analysis. It provides data structures and functions that make working with structured data, such as tabular data, more efficient and intuitive. Pandas is built on top of NumPy and is often used in conjunction with other libraries like NumPy, Matplotlib, and scikit-learn. It is widely used in data analysis, data preprocessing, feature engineering, and exploratory data analysis. It is a valuable tool for tasks like data cleaning, data transformation, data wrangling, and data aggregation.
  Pandas is also commonly used in combination with machine learning libraries for data preprocessing and model evaluation.

- Matplotlib

  Matplotlib is a popular Python library for creating static, animated, and interactive visualizations. It provides a wide range of functions and methods for creating various types of plots, charts, and graphs.
  Matplotlib is highly customizable and allows you to create publication-quality visualizations for data analysis, exploration, and presentation. It is widely used in data analysis, scientific research, engineering, and visualization tasks. It is a versatile library that can be used for basic plotting as well as creating complex, publication-quality visualizations.
  Matplotlib works well in conjunction with other libraries such as NumPy, Pandas, and SciPy, and it is often used in combination with data analysis and machine learning workflows.

- Seaborn

  Seaborn is a Python data visualization library that is built on top of Matplotlib. It provides a high-level interface for creating attractive and informative statistical graphics. Seaborn aims to simplify the process of creating visualizations and enhance the visual aesthetics by providing default styles and color palettes. It is widely used in the data science community for its ability to create visually appealing and informative statistical visualizations with minimal effort. It is particularly useful for exploratory data analysis, data visualization in research and presentations, and communicating insights from data. By combining the power of Seaborn with other data analysis libraries like Pandas and NumPy, you can create compelling visualizations that help in understanding and communicating patterns and relationships in your data.

- Sklearn

  Scikit-learn, also known as sklearn, is a popular machine learning library in Python. It provides a wide range of tools and algorithms for machine learning tasks, including classification, regression, clustering, dimensionality reduction, and model selection. Sklearn is built on top of other scientific computing libraries such as NumPy, SciPy, and Matplotlib, and it is designed to be easy to use and integrate with other Python libraries. It is widely used in academia and industry for machine learning tasks due to its simplicity, scalability, and comprehensive set of tools. Whether you are a beginner in machine learning or an experienced practitioner, sklearn provides a solid foundation for building and deploying machine learning models in Python.

- XG Boost

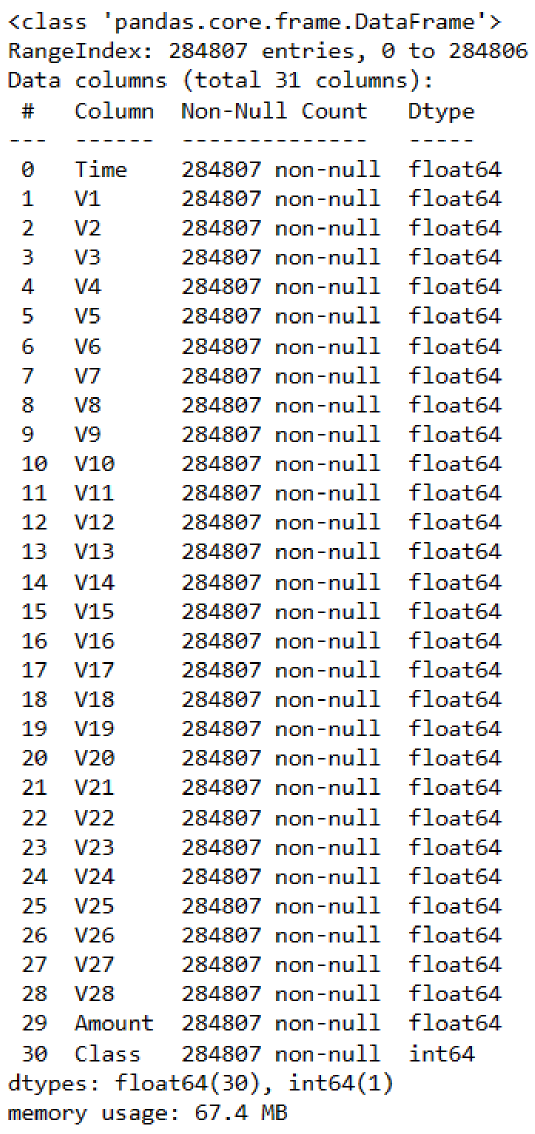
  XGBoost, short for Extreme Gradient Boosting, is a powerful machine learning algorithm that has gained significant popularity in recent years. It is an optimized implementation of the gradient boosting framework and is known for its high performance and ability to handle large and complex datasets. XGBoost has achieved remarkable success in various machine learning competitions and is widely used in industry applications. Its speed, scalability, and accuracy make it a popular choice for both research and practical purposes. If you have a dataset with complex patterns or want to improve the performance of your machine learning models, XGBoost is definitely worth considering.

3.2. Phase 2: Data Pre-Processing

Data pre-processing is an important stage in the examination of research papers. It entails cleaning and translating raw data into an analysis-ready format. This procedure entails deleting unnecessary data, dealing with missing values, dealing with outliers, and normalizing the data.

Data Exploration

The basic information of the data is:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64  
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

As per the count per column, there are no null values.

Now we check the Transaction details.

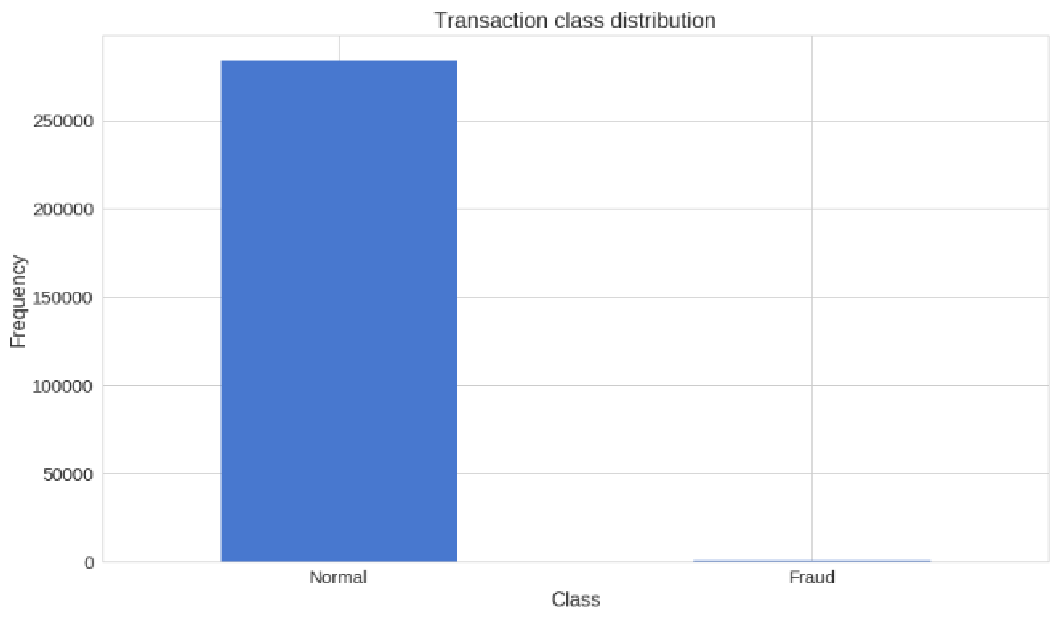


Fig 3.1: Transaction Class Distribution

This shows that we have a highly imbalanced dataset. Normal transactions overwhelm the fraudulent ones by a large margin.
How different are the amount of money used in different transaction classes:For fraud transactions:

```
count      492.000000
mean       122.211321
std        256.683288
min          0.000000
25%          1.000000
50%          9.250000
75%        105.890000
max       2125.870000
Name: Amount, dtype: float64
```

For normal transactions:

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

The following figure shows the graphical representation of the same:

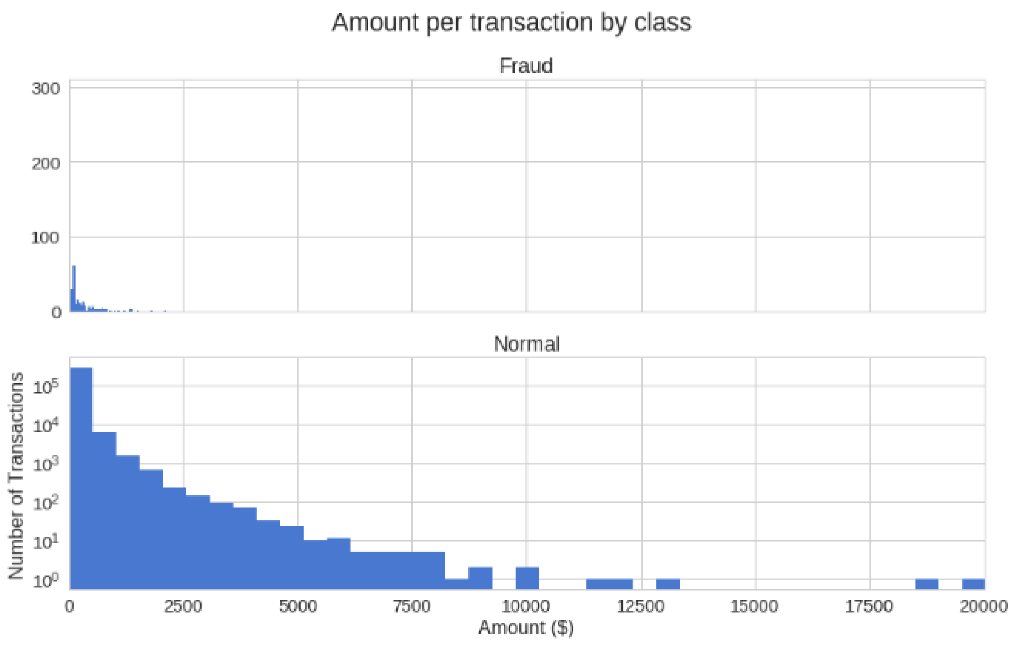


Fig 3.2: Amount per transaction by class

To check if time of transactions really matter

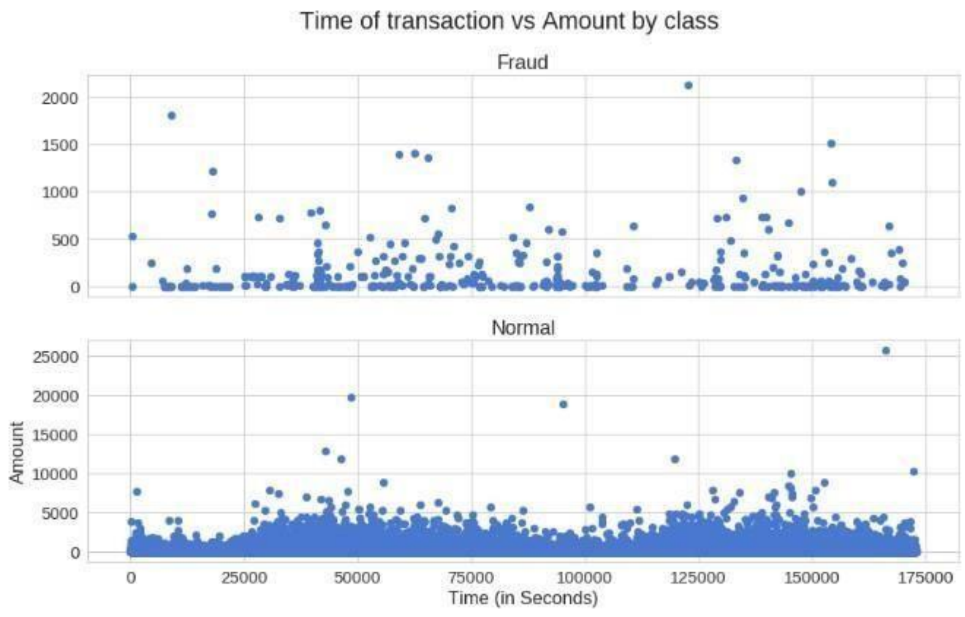


Fig 3.3: Time of transaction vs Amount by class

Since the data was imbalanced, use of standard scaler can fix it.

```
sc = StandardScaler()

amount = data['Amount'].values

data['Amount'] = sc.fit_transform(amount.reshape(-1, 1))
```

3.3. Phase 3: Model Building

Splitting the dataset into train and test and define dependent and independent variable.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 1)
```

After splitting the dataset, machine learning models are to be applied.

1. Decision Tree

Decision Tree is of type supervised learning algorithm, graphically represented for getting all possible solutions to a problem based on given conditions. It used CART algorithm which stands for Classification and Regression Tree algorithm.

```
DT = DecisionTreeClassifier(max_depth = 4, criterion = 'entropy')
DT.fit(X_train, y_train)
dt_yhat = DT.predict(X_test)
```

2. K-Nearest Neighbours

K Nearest Neighbours also known as lazy learner algorithm is a supervised Learning algorithms used for both classification and regression. Instead of instantly learning the dataset, it first stores the dataset and then at the time of classification performs action on the given dataset.

KNN is an Instance-Based Learning that compares new instances with instances stored in memory at the time of training of dataset. A new instance is classified by measuring its distances with the instances retrieved from memory, defined in terms of standard Euclidean Geometry, that is, distance between points in n- dimensional space.

The accuracy of this model depends upon two factors:

The value of 'K'

The number of selected features.

```
n = 7

KNN = KNeighborsClassifier(n_neighbors = n)

KNN.fit(X_train, y_train)
```

```
knn_yhat = KNN.predict(X_test)
```

3. Logistic Regression

Logistic Regression is a supervised machine learning algorithm used for binary classification problem. A logistic function is used to describe the likelihood of the binary result, mapping any real-valued input to a value between 0 and 1. The decision boundary is set at a probability threshold of 0.5, and the procedure uses maximum likelihood estimation to estimate the logistic function’s parameters. A straightforward and understandable approach that can handle both category and numerical information is logistic regression. It has been extensively utilized in a variety of industries, including marketing, banking, and healthcare, as a baseline model.

```
lr = LogisticRegression()

lr.fit(X_train, y_train)

lr_yhat = lr.predict(X_test)
```

4. Support Vector Machines

Machine provides a simple geometrical interpretation in a high-dimensional feature space that is nonlinearly related to input space. SVMs provide a learning technique for Pattern Recognition, Regression Estimation. The solution provided by SVM is theoretically elegant, computationally efficient and very effective in many large practical problems

```
        svm = SVC()
        svm.fit(X_train, y_train)
        svm_yhat = svm.predict(X_test)
```

5. Random Forest

Random Forest grows many classification trees. To classify a new object from an input vector, the input vector is added down each of the trees in the forest. One of the reason for choosing this algorithm is because it runs efficiently on large databases, also learning is very fast in this algorithm. Random Forest is a kind of supervised machine learning algorithm used for both Classification and Regression. Its builds multiple decision trees and merges them together to get a more accurate and stable prediction.

```
rf = RandomForestClassifier(max_depth = 4)

rf.fit(X_train, y_train)

rf_yhat = rf.predict(X_test)
```

6. XG Boost

Extreme Gradient Boosting, or XGBoost, is a well-known machine learning method that excels at a variety of tasks, especially in gradient boosting frameworks. Gradient boosting machines are ensemble learning techniques that integrate a number of weak prediction models (usually decision trees) to produce a stronger model. It is an optimized implementation of these techniques.

```
xgb = XGBClassifier(max_depth = 4)

xgb.fit(X_train, y_train)

xgb_yhat = xgb.predict(X_test)
```

Chapter-4

Implementation and Visualisations

4.1. Confusion Matrix

A confusion matrix is a widely used tool in machine learning to visualize the performance of a classification model. It presents a tabular representation of the predictions made by the model compared to the actual labels of the data.

The confusion matrix consists of four key components:

True Positives (TP): The number of instances correctly predicted as positive by the model.
True Negatives (TN): The number of instances correctly predicted as negative by the model.
False Positives (FP): The number of instances incorrectly predicted as positive by the model (also known as Type I errors).
False Negatives (FN): The number of instances incorrectly predicted as negative by the model (also known as Type II errors).
The confusion matrix helps in evaluating the model's performance by providing insights into the types and quantities of prediction errors made.

Using the confusion matrix, several evaluation metrics can be derived, such as:

Accuracy: The proportion of correctly classified instances out of the total number of instances.
Accuracy = (TP + TN) / (TP + TN + FP + FN)

Precision: The proportion of true positive predictions out of all positive predictions made by the model.
Precision = TP / (TP + FP)

Recall (Sensitivity or True Positive Rate): The proportion of true positive predictions out of all actual positive instances.
Recall = TP / (TP + FN)

F1 Score: The harmonic mean of precision and recall, providing a balanced measure of a model's performance.
F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

The confusion matrix allows for a comprehensive analysis of a model's performance, particularly in understanding the trade-offs between different evaluation metrics. It provides insights into the model's ability to correctly classify instances across different classes.

It is important to refer to authoritative sources or research papers for a more comprehensive understanding of the confusion matrix and its significance in machine learning.

4.2. Accuracy

One of the often employed measures to assess the effectiveness of machine learning models is accuracy. It calculates the percentage of accurate predictions the model produced using a certain dataset. The number of accurate forecasts divided by the total number of predictions, multiplied by 100, is used to calculate the accuracy of a model.

While accuracy is an easy-to-understand statistic, it may not necessarily give a whole picture of a model's performance, particularly in unbalanced datasets or when the cost of various types of mistakes varies. The distribution of classes or the relative significance of various sorts of mistakes are not taken into consideration by accuracy.

Decision Tree:

```
Accuracy score of the Decision Tree model is 0.9991583957281328
```

K-Nearest Neighbours:

```
Accuracy score of the K-Nearest Neighbors model is 0.999288989494457
```

Logistic Regression:

```
Accuracy score of the Logistic Regression model is 0.9989552498694062
```

Support Vector Machine:

```
Accuracy score of the Support Vector Machines model is 0.99931801033141
8
```

Random Forest:

```
Accuracy score of the Random Forest model is 0.9991874165650937
```

XG Boost:

```
Accuracy score of the XGBoost model is 0.999506645771664
```

4.3. F1 Score

The F1 score is a widely used metric in machine learning to assess the performance of models. It is particularly valuable when dealing with imbalanced datasets, where the class distribution is uneven.

The F1 score combines precision and recall into a single value, providing a balanced evaluation of a model's performance. Precision measures the proportion of correctly predicted positive instances out of all instances predicted as positive, while recall measures the proportion of correctly predicted positive instances out of all actual positive instances.

The F1 score is calculated using the harmonic mean of precision and recall. It ranges from 0 to 1, with 1 indicating perfect precision and recall. A higher F1 score signifies better performance in terms of both avoiding false positives and false negatives.

<u>Decision Tree:</u>

```
F1 score of the Decision Tree model is 0.7521367521367521
```

<u>K-Nearest Neighbours:</u>

```
F1 score of the K-Nearest Neighbors model is 0.7949790794979079
```

<u>Logistic Regression:</u>

```
F1 score of the Logistic Regression model is 0.6666666666666666
```

<u>Support Vector Machine:</u>

```
F1 score of the Support Vector Machines model is 0.7813953488372093
```

<u>Random Forest:</u>

```
F1 score of the Random Forest model is 0.7454545454545454
```

<u>XG Boost:</u>

```
F1 score of the XGBoost model is 0.8495575221238937
```

4.4. Precision

Precision is a fundamental metric used in machine learning to evaluate the performance of models, particularly in binary classification tasks. It measures the proportion of correctly predicted positive instances out of all instances predicted as positive.

Precision focuses on the accuracy of positive predictions and quantifies how well a model avoids false positives. It is calculated by dividing the number of true positive predictions by the sum of true positive and false positive predictions.

Precision = True Positives / (True Positives + False Positives)

The precision score ranges from 0 to 1, with 1 representing perfect precision where all positive predictions are correct and no false positives occur. A higher precision score indicates better performance in terms of minimizing false positives.

4.5. Recall

Recall is a key metric in machine learning used to evaluate the performance of models, particularly in binary classification tasks. It measures the proportion of correctly predicted positive instances out of all actual positive instances.

Recall, also known as sensitivity or true positive rate, quantifies how well a model avoids false negatives. It is calculated by dividing the number of true positive predictions by the sum of true positive predictions and false negative predictions.

Recall = True Positives / (True Positives + False Negatives)

The recall score ranges from 0 to 1, with 1 indicating perfect recall where all actual positive instances are correctly identified and no false negatives occur. A higher recall score signifies better performance in terms of minimizing false negatives.

Chapter-5

Conclusion and Future Work

5.1. Scope of Improvement

Credit card fraud detection is an ongoing challenge for financial institutions and businesses. While significant advancements have been made in this field, there is always room for improvement. Here are some areas where the scope of improvement exists in credit card fraud detection projects:

Data quality and variety: Improving the data's quality and diversity can increase the accuracy of fraud detection models. A more complete and reliable model can be produced by incorporating a wide range of transaction kinds, user behaviours, and geographic information.
Real-time detection: Integrating real-time fraud detection tools enables quick response when questionable behaviour is found. By utilising tools like stream processing and machine learning algorithms, fraud may be identified and stopped almost instantly, reducing potential losses.
Utilising cutting-edge machine learning approaches like deep learning, reinforcement learning, or ensemble methods can improve the precision and effectiveness of fraud detection models. These methods can find intricate patterns and anomalies in data that conventional approaches can miss.
By focusing on areas like these, the scope for improvement in credit card fraud detection projects can be broadened, leading to more accurate and effective systems for detecting and preventing fraudulent activities.

5.2. Summary

1. Loaded few modules and initial cleaning of data
2. Data Pre-Processing
3. Balancing of data and visualization
4. Implemented few models and visualized the results.

S.No	ML Model	Accuracy	F1 Score
1	Decision Tree	0.9991583957281328	0.7521367521367521
2	K-Nearest Neighbour	0.999288989494457	0.7949790794979079
3	Logistic Regression	0.9989552498694062	0.6666666666666666
4	Support Vector Machine	0.999318010331418	0.7813953488372093
5	Random Forest	0.9991874165650937	0.7454545454545454
6	XG Boost	0.999506645771664	0.8495575221238937

Table 5.1: Summary Table

The following graph shows the comparison of different models based on their F1 score

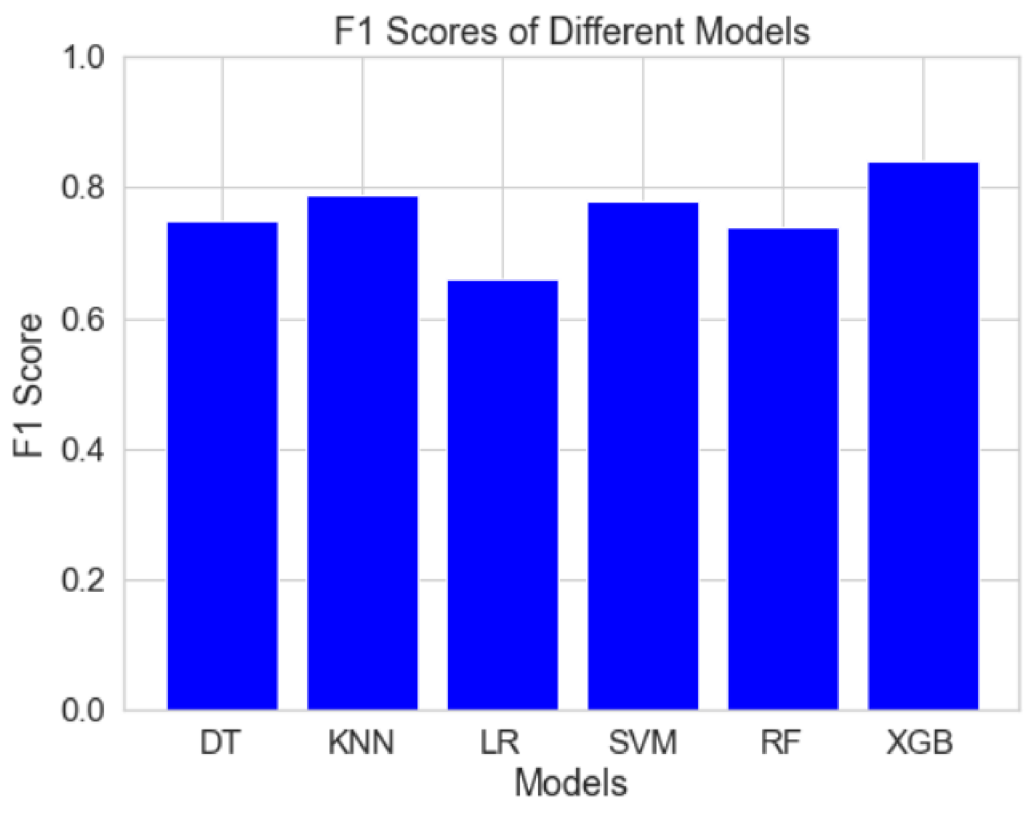


Fig 5.1: F1 scores of different models

Hence we can say the XG Boost model is the best model amongst all.

5.3. Conclusion

Credit card fraud is a significant issue that affects millions of people worldwide. Traditional fraud detection methods are not sufficient to keep up with evolving fraud tactics. Machine learning offers a promising approach to detect and prevent credit card fraud. Python and machine learning algorithms can analyze large amounts of data to identify suspicious activity. The goal is to build an effective fraud detection system that can accurately identify fraudulent transactions.

Overall, the project aimed to address the challenge of credit card fraud detection using machine learning algorithms and techniques. By analyzing transaction data and identifying patterns and anomalies, the goal is to build a predictive model that can accurately detect and prevent fraudulent activities, thereby protecting customers and financial institutions from potential losses.

References

Dal Pozzolo, A., Caelen, O., Le Borgne, Y. A., Waterschoot, S., Bontempi, G. (2015). Learned lessons in credit card fraud detection from a practitioner perspective. Expert Systems with Applications, 42(10), 5008-5020.

Bhattacharyya, S., Bhattacharya, S., & Tharakunnel, K. (2011). A hybrid credit card fraud detection model using neural networks. Decision Support Systems, 50(3), 602-614.

Fawcett, T., Provost, F. (1997). Adaptive Fraud Detection. Data Mining and Knowledge Discovery, 1(3), 291-316.

Phua, C., Lee, V., Smith-Miles, K., Gayler, R. (2005). A comprehensive survey of data mining-based fraud detection research. Arxiv preprint cs/0506066.

Zhang, X., Zhou, Y., Wang, L., & Guan, L. (2018). Credit card fraud detection using deep learning based on auto-encoder and restricted Boltzmann machine. Neurocomputing, 321, 321-332.

Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., & Bontempi, G. (2017). Credit card fraud detection: A realistic modeling and a novel learning strategy. IEEE Transactions on Neural Networks and Learning Systems, 29(8), 3784-3797.

Bhattacharya, S., & Islam, M. (2018). Credit card fraud detection using machine learning: A systematic review. Journal of Big Data, 5(1), 1-37.