TIME SERIES AND LOGISTIC REGRESSION ANALYSIS

MSC in Data Analytics, January 2024

Vishnunath Nharekkat
School of computing
National College of Ireland
Dublin, Ireland
x22234217@student.ncirl.ie

***Abstract*— This evaluates a monthly time series dataset of average cocoa prices from October 1995 to March 2024. Initially, the raw data is visually analyzed to detect trends, seasonality, and irregular components. Following that, several time series models are calculated and discussed, including basic models such as Moving Averages, Exponential Smoothing approaches, and ARIMA/SARIMA models. Forecasts for the next six months (October 2023 to March 2024) are made and compared to actual data using data from September 2023 as a training set. The study closes by selecting the best forecasting model based on accuracy and applicability for the cocoa price dataset.**

In part B section, we examine a dataset of credit card transactions to better understand and detect fraudulent behaviors. With the increase in digital transactions, the danger of fraud has become a major issue for both customers and financial institutions. Our dataset includes 283,726 transactions, including both fraud and non-fraud events. We hope to discover patterns and features related to fraudulent transactions using exploratory data analysis and logistic regression modeling. We want to construct a reliable fraud detection model by considering data such as transaction timing, quantity, and extra parameters. Our findings underscore the relevance of data-driven methods to fraud prevention and financial transaction security.

PART A: TIME SERIES ANALYSIS

I. INTRODUCTION

The cocoa industry is essential to the world economy because cocoa beans are used to make chocolate and other confectionery items. As a result, cocoa pricing is of great concern to stakeholders throughout the supply chain, from farmers and producers to manufacturers and consumers. Understanding the fluctuations in cocoa prices is critical for making educated decisions, managing risks, and projecting future trends.

In our study, we performed a thorough investigation of a dataset of monthly average cocoa prices. This dataset covers more than three decades, including observations from October 1995 to March 2024. The collection contains a wealth of historical pricing information, including a complete record of cocoa price variations throughout time.

The fundamental goal of this research is to create and analyze time series models that can accurately represent the underlying patterns and volatility in cocoa prices across time. To reach this purpose, we take a systematic methodology that begins with a preliminary evaluation of the raw time series data. This assessment entails displaying the data to determine any discernible trends, seasonal patterns, or irregular components that may present.

Following the preliminary assessment, we will estimate and analyze appropriate time series models from several categories. These include simple time series models like Moving Averages, Exponential Smoothing, and Autoregressive Integrated Moving Averages (ARIMA). Each model is thoroughly examined using relevant diagnostic tests and checks to assure its validity and reliability.

To evaluate the models' predicting performance, we used data up to September 2023 as a training set to generate forecasts for the next six months (October 2023 to March 2024). These estimates are then compared to real cocoa prices for the same time period to determine the accuracy and reliability of each model.

By the end of this study, we aim to develop an ideal forecasting model that produces the most accurate and trustworthy cocoa price forecasts. This model can help cocoa industry stakeholders make educated decisions, manage risks, and prepare for the future more effectively.

II. EXPLORATORY DATA ANALYSIS (EDA)

II.1 Dataset Description

The dataset CocaPrice.csv contains the price of Cocoa from October 1995 to March 2024. The dataset contains historical pricing information, including a complete record of cocoa price variations throughout the time.

This dataset has 354 rows and 2 columns containing both numerical values. The details of the dataset are mentioned below:

- **Date:** integer encoding of the census region.
- **Price:** IT standard encoding for US states.

The CocoaPrice dataset is put into a data frame in a Jupyter notebook for further analysis. Figure 1 provides additional information regarding the dataset's properties.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 354 entries, 0 to 353
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Date    354 non-null    object
 1   Price   354 non-null    float64
dtypes: float64(1), object(1)
memory usage: 5.7+ KB
```

Fig. 1. Information of the dataset

II.2 Data Visualization

First, we have to create a data index for yearly data. The data frame must include historical cocoa price data from October 1995 to March 2024. This procedure indexes price data by date, making it easy to manipulate, analyze, and visualize the yearly cocoa price.

a) Descriptive Summary Variables: Figure 2 depicts the necessary numerical variables and descriptive summary, including count, mean, percentile, maximum and minimum values.

	Price
count	354.000000
mean	1853.000311
std	623.238555
min	874.140000
25%	1396.147500
50%	1805.985000
75%	2168.545000
max	6510.160000

Fig. 2. Descriptive Statistics

It may be seen that the minimum and maximum prices of Cocoa are 874.14 and 6510.16 respectively.

b) Line plot: A line plot is a sort of graph in which data points are represented as dots or markers on a line. It is commonly used to depict trends or patterns across time.

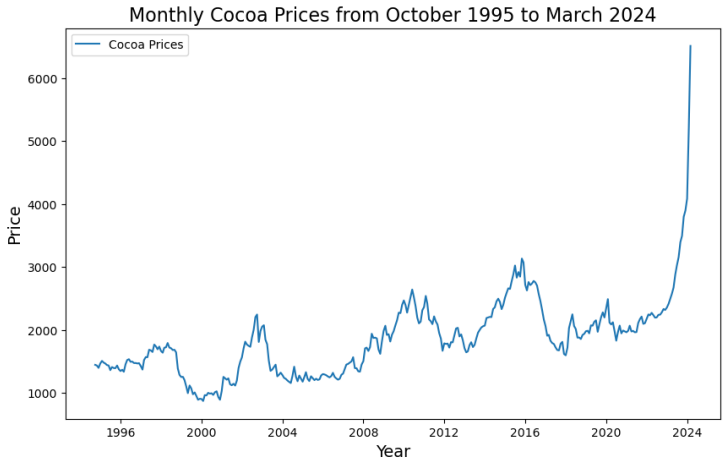

Fig. 3. Line plot of monthly Cocoa price

From the above figure, we can understand the monthly prices of cocoa. After 2020 we can see a sudden hike in the price of cocoa.

c) Histogram: Histogram is a graphical representation of numerical data. Here, Figure 4 shows the histogram of Cocoa price and its frequency. The range of 1100 to 2500 shows a higher frequency compared to others.

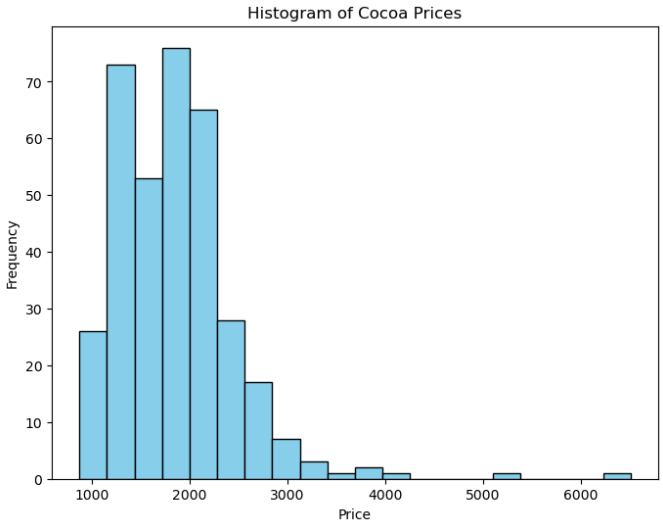

Fig. 4. Histogram of Cocoa price

d) Bar Plot: A Bar plot is a statistical plot that shows the relationship between categorical and numerical variables. Figure 5, shows the relationship between the yearly price variation of Cocoa.

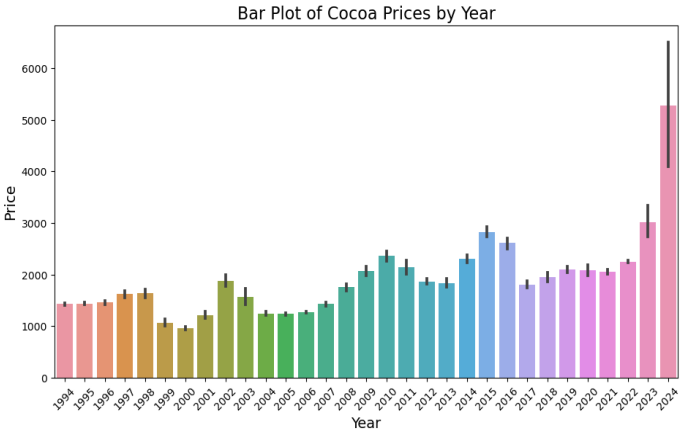

Fig. 5. Bar plot of yearly Cocoa price

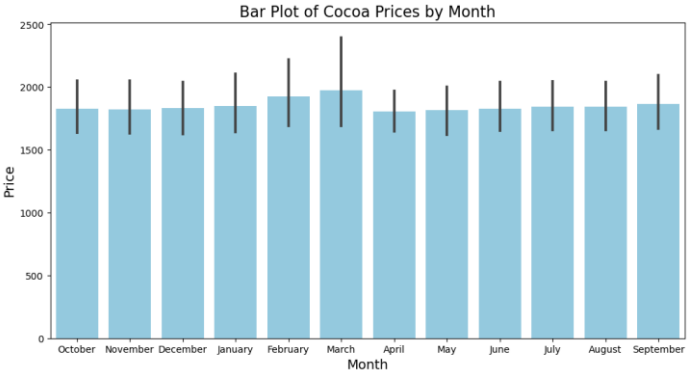

Fig. 6. Monthly prices of Cocoa

Figure 6 shows the monthly prices of Cocoa. From this, we can understand that March month has the highest price in a year.

III. Methodology

Data preparation or data preprocessing is a key step in machine learning procedures. It entails transforming raw data into a form suitable for training machine learning models.

III.1 Handling missing values

To handle missing values, we must determine whether any missing values exist in the dataset. For that, we are using isna().sum() function to check. We don't have any missing values in this dataset.

```
df.isna().sum()
```

```
Price    0
dtype: int64
```

Fig. 7. Details of null-values

III.2 Components of Time series:

A Time series is essentially a series of data points collected over time to help us understand how a specific variable changes over time. Time acts as a framework, allowing us to meaningfully organize these observations, whether by years, months, days, or hours.

When we look at the elements that influence the values within a time series, we can classify them into three main components: Trend, Seasonal Variation, and Residuals.

Trends: It indicates the long-term trend of the data. It allows us to determine if the variable is generally increasing, decreasing, or staying relatively stable over time.

Seasonal Variation: Seasonality refers to patterns that repeat at defined intervals, such as daily, weekly, or yearly.

Residuals: Residual components reflect random oscillations that do not follow a predefined pattern. They occurs as a result of unanticipated occurrences such as rapid market fluctuations or natural calamities.

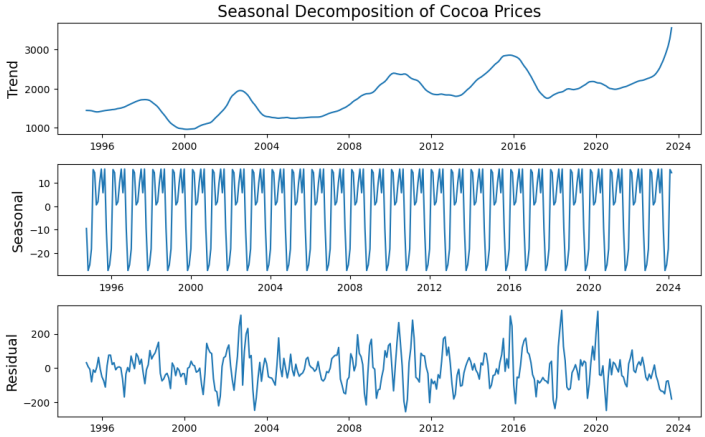

Fig. 8. Seasonal decomposition of Cocoa prices

Figure 8 shows the seasonal decomposition of cocoa prices. We can see three different line plots define the variations of Trends, Seasonal Variations, and Residuals of Cocoa prices.

The trend plot displays the trend component. It shows the long-term pattern of cocoa prices. The Price fluctuates both up and down over time. The second plot depicts the seasonal patterns that occur throughout the year. It helps to discover seasonal patterns in the cocoa price data.

The seasonal decomposition map reveals the underlying components of the cocoa price. It helps us to learn seasonal patterns, long-term trends, and unusual patterns in the data.

IV. Model Evaluation and Result

In this project, we are using three different Simple time series models such as Simple moving average, Weighted moving average, and Naïve. After this, we implemented two exponential smoothing models such as Single exponential smoothing and double exponential smoothing. Finally, we implemented the model ARIMA (Auto Regressive Integrated Moving Average).

IV.1 Simple Moving Average and Weighted Moving Average:

Simple Moving Average (SMA) measures the average of a specified number of data points over a given period. It smooths out fluctuations, helping to identify trends by highlighting the overall direction of the data.

Weighted Moving Average (WMA) assigns various weights to each data point in the calculation. It prioritizes current data, which reflects changing trends more accurately.

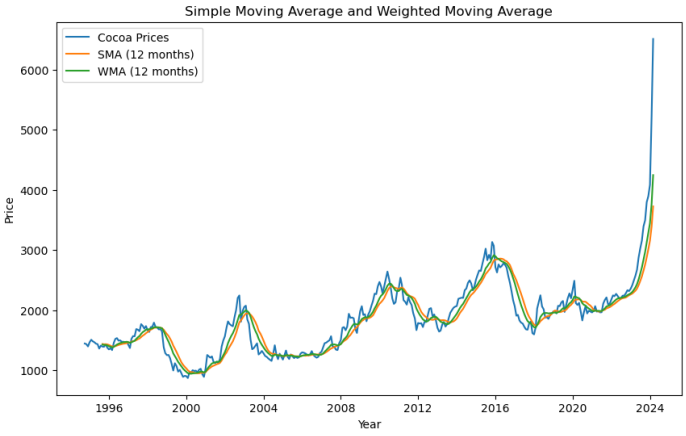

Fig. 9. Graphical visualization of simple and weighted moving average

Figure 9 depicts the graphical visualization of the Simple Moving Average and Weighted Moving Average with actual data.

```
Metrics for Simple Moving Average (SMA):
MAE: 180.18494639376217
RMSE: 289.8563203974798
MSE: 84016.68647436648

Metrics for Weighted Moving Average (WMA):
MAE: 135.55885252661565
RMSE: 222.88515088810465
MSE: 49677.79048641317
```

Fig. 10. Metric calculations of Simple Moving Average and Weighted Moving Average

Figure 10 shows the Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Square Error (MSE) of the Simple Moving Average and Weighted Moving Average. By comparing this Weighted Moving Average model performs better than the Simple Moving Average. The Weighted Moving Average has less error compared to Simple Moving Average.

IV.2 Naïve Model:

The Naïve model essentially predicts future values of a time series using the most recent observation. It assumes no change from the previous known values, making it simple but sometimes insufficient for good forecasting.

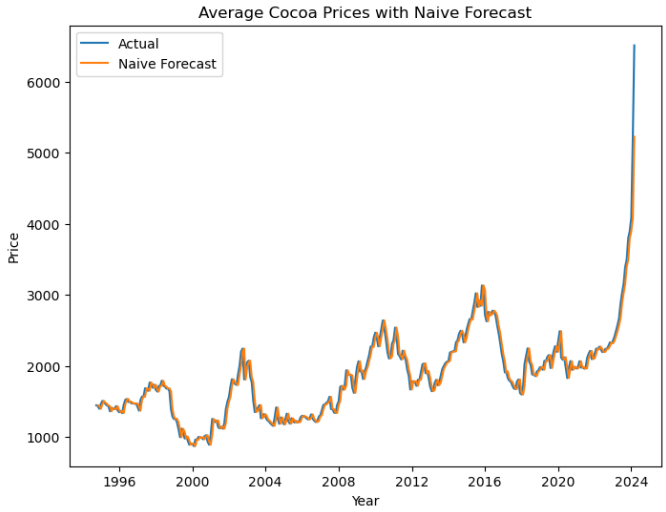

Fig. 11. Graphical visualization of Naïve Forecast

Figure 11 shows the graphical representation of the Naive forecast model. We can understand that it has very few errors compared to the previous two simple time series models. Most of the lines of the Naïve forecast are very close to the actual values of the data.

```
Naive Method:
MAE: 81.46195467422098
MSE: 18374.45655552408
RMSE: 135.55241257729085
```

Fig. 12. Graphical visualization of Naïve Forecast

Figure 12 illustrates the metric calculation of the Naïve forecast model. Here compared to the Simple moving average and Weighted Moving Average, the Naïve method has low error. This simple time series model is performing better than other models.

IV.3 Single Exponential Smoothing and Double Exponential Smoothing:

Single Exponential Smoothing estimates future values of a time series by illustrating current data. It is based on a weighted average of previous observations, with decreasing weights as observations become older.

Double exponential smoothing predicts future values of a time series while considering trend and seasonality. It makes forecasts based on weighted averages of previous observations and past trend estimates, responding to changing patterns over time.

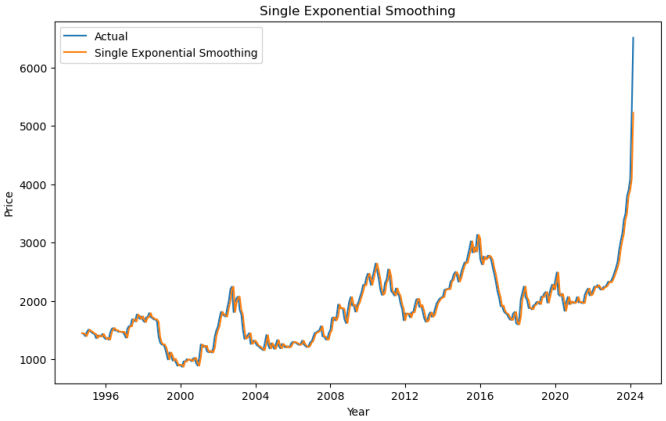

Fig. 13. Graphical visualization of Single Exponential Smoothing

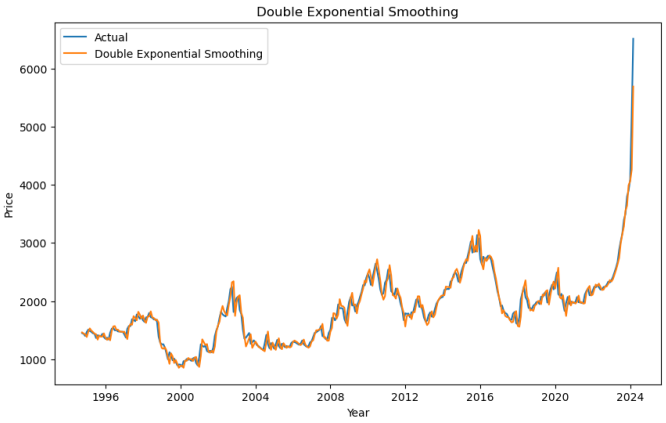

Fig. 14. Graphical visualization of Double Exponential Smoothing

Figures 13 and 14 shows the graphical visualization of Single Exponential Smoothing and Double Exponential Smoothing. By comparing these two graphs we can understand that Single exponential smoothing performs better than Double Exponential Smoothing. Double exponential Smoothing shows more error with actual values.

```
Metrics for Single Exponential Smoothing (SES):
MAE: 2007.653320577542
RMSE: 2265.201469937214
MSE: 5131137.699405714
```

Fig. 15. Metrics of Single Exponential Smoothing

```
Metrics for Double Exponential Smoothing (DES):
MAE: 4521.182084728901
RMSE: 4550.524227697948
MSE: 20707270.746866006
```

Fig. 16. Metrics of Double Exponential Smoothing

Figures 15 and 16 show the metrics calculation of Single Exponential Smoothing and Double Exponential Smoothing. By comparing the metrics Single Exponential Smoothing has minimum error. So we can understand that in exponential smoothing Single Exponential Smoothing performs better than Double Exponential Smoothing.

IV.3 ARIMA (Auto Regressive Integrated Moving Average):

ARIMA (Auto Regressive Integrated Moving Average) is a statistical technique for evaluating and forecasting time series. It uses autoregression, differencing, and moving average approaches to model and predict values based on previous data, making it suitable for a variety of forecasting jobs.

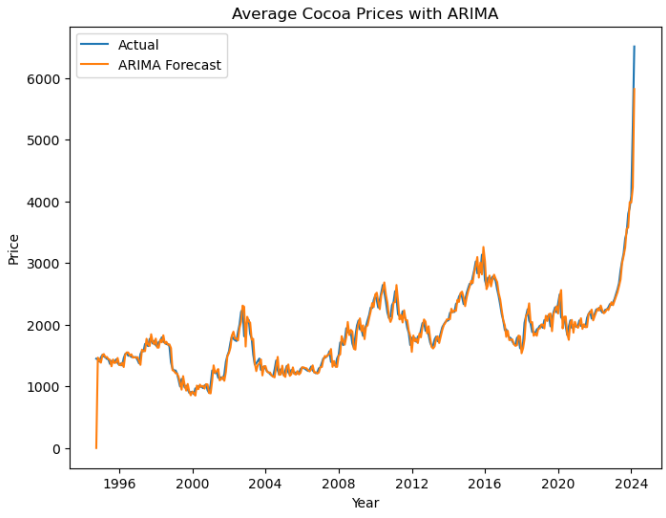

Fig. 17. Graphical visualization of ARIMA

Figure 17 illustrates the graphical representation of ARIMA. Here by visualizing the graph we can understand that has some errors compare to actual data. Let's discuss it by calculating metrics.

```
Metrics for ARIMA:
MAE: 3641.4577089028207
RMSE: 3681.7953549422805
MSE: 13555617.035674552
```

Fig. 18. Metrics of ARIMA

Figure 18 illustrates the metric calculation of the ARIMA model. By evaluating the metrics we can the MSE has high value and it cannot provide good results. A decrease in error helps the model to perform well.

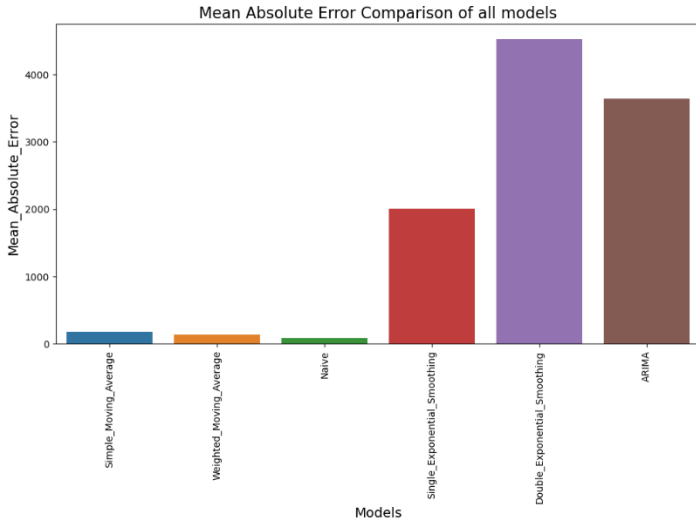

Fig. 19. Comparison of models

By comparing the models Simple Moving Average, Weighted Moving Average, Naïve, Single Exponential Smoothing, Double Exponential Smoothing, and ARIMA. The Naïve Model is performing excellently compared to other models. It shows very little Mean Absolute Error. So, we can suggest Naïve as a good model for predicting.

V. Conclusion

Following a thorough examination of several time series models, including simple moving average, weighted moving average, Naive, single exponential smoothing, double exponential smoothing, and ARIMA, it is clear that Naive outperforms the others in terms of forecast accuracy metrics such as RMSE, MSE, and MAE. Despite its simplicity, naive forecasting exhibits strong predictive power for cocoa price data from October 1995 to September 2023, accurately capturing underlying patterns and trends.

Simple moving averages and weighted moving averages, while simple to design, struggle to respond to the dynamic nature of cocoa price variations, yielding less accurate forecasts than Naive. Single and double exponential smoothing approaches, while capable of capturing trends and seasonality, have slightly larger predicting errors than Naive.

ARIMA, a more advanced model, produces competitive results but necessitates careful parameter adjustment and assumption validation, making it more difficult to implement than Naive.

Finally, for this particular time series dataset, Naive emerges as the best forecasting model due to its simplicity, ease of implementation, and greater predictive performance. However, for accurate forecasting in the future, it is critical to examine the specific qualities of the data as well as prospective changes in underlying trends.

PART B: LOGISTIC REGRESSION

VI. Introduction

Credit card transactions are now a normal aspect of our daily lives in the digital age. However, the convenience of Internet shopping comes with the risk of fraudulent activity. Credit card fraud happens when unlawful purchases are performed with someone else's credit card information, resulting in financial loss and annoyance for both individuals and organizations.

To address this issue, data analytics techniques such as logistic regression can be used to detect fraudulent transactions and avoid potential losses. Logistic regression is a statistical method for estimating the likelihood of a binary outcome, such as whether a transaction is fraudulent or not.

In this work, we analyze a dataset including anonymized data from over 280,000 credit card transactions that were classed as fraudulent or non-fraudulent. Our goal is to create a logistic regression model that can accurately predict the possibility of fraud based on several transaction characteristics.

We hope to gain a better understanding of the elements that contribute to credit card theft by doing exploratory data analysis, building models, and evaluation. We can help financial institutions and businesses discover and prevent fraudulent actions, thereby protecting consumers' interests and maintaining trust in electronic payment systems.

VII. Exploratory Data Analysis (EDA)

VII.1 Dataset Description

The fraud.csv dataset contains credit card transaction data that has been cleaned as well as partially standardized. It contains 31 variables, which include the following:

- **Time:** It is the time of transaction measured in seconds over two days.
- **Amount:** It is the transaction amount in Euro.
- **Class:** It is the target variable that determines whether the transaction is fraudulent or not. The values of the Class column are in binary format. Binary 1 denotes fraud and 0 represents non-fraud.
- **V1 to V28:** This is a transformed feature derived from the original dataset.

The fraud dataset is loaded into a data frame in a Jupyter notebook for further processing. More information about the features in the dataset is shown in Figure 20.

```
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Time    283726 non-null  float64
 1   V1      283726 non-null  float64
 2   V2      283726 non-null  float64
 3   V3      283726 non-null  float64
 4   V4      283726 non-null  float64
 5   V5      283726 non-null  float64
 6   V6      283726 non-null  float64
 7   V7      283726 non-null  float64
 8   V8      283726 non-null  float64
 9   V9      283726 non-null  float64
 10  V10     283726 non-null  float64
 11  V11     283726 non-null  float64
 12  V12     283726 non-null  float64
 13  V13     283726 non-null  float64
 14  V14     283726 non-null  float64
 15  V15     283726 non-null  float64
 16  V16     283726 non-null  float64
 17  V17     283726 non-null  float64
 18  V18     283726 non-null  float64
 19  V19     283726 non-null  float64
 20  V20     283726 non-null  float64
 21  V21     283726 non-null  float64
 22  V22     283726 non-null  float64
 23  V23     283726 non-null  float64
 24  V24     283726 non-null  float64
 25  V25     283726 non-null  float64
 26  V26     283726 non-null  float64
 27  V27     283726 non-null  float64
 28  V28     283726 non-null  float64
 29  Amount  283726 non-null  float64
 30  Class   283726 non-null  int64
dtypes: float64(30), int64(1)
memory usage: 67.1 MB
```

Fig. 20. Information of the dataset

VII.2 Data Visualization

a) Descriptive Summary Variables: The important numerical variables and descriptive summaries including count, mean, percentile, maximum, and minimum values are shown in Figure. 21.

	count	mean	std	min	25%	50%	75%	max
Time	283726.0	94811.077600	47481.047891	0.000000	54204.750000	84692.500000	139298.000000	172792.000000
V1	283726.0	0.003038	1.000000	-28.956239	-0.470195	0.010464	0.675590	1.260214
V2	283726.0	-0.002511	1.000000	-44.158375	-0.364559	0.038835	0.485991	13.395087
V3	283726.0	0.001069	1.000000	-32.031662	-0.589708	0.119285	0.680700	6.219043
V4	283726.0	-0.002098	1.000000	-4.018693	-0.601148	-0.015732	0.523020	11.932920
V5	283726.0	0.001327	1.000000	-82.601759	-0.500963	-0.038829	0.444600	25.273389
V6	283726.0	-0.000856	1.000000	-19.641043	-0.577381	-0.206593	0.297908	55.034118
V7	283726.0	0.001467	1.000000	-35.479777	-0.450049	0.033282	0.464682	98.226798
V8	283726.0	-0.000725	1.000000	-62.097827	-0.177115	0.018572	0.276241	16.968859
V9	283726.0	-0.001457	1.000000	-12.263038	-0.588065	-0.048011	0.544026	14.235602
V10	283726.0	-0.001338	1.000000	-22.842897	-0.497561	-0.086618	0.421419	22.059619
V11	283726.0	0.000198	1.000000	-4.709314	-0.747653	-0.031712	0.725989	11.798052
V12	283726.0	-0.000719	1.000000	-18.783749	-0.408373	0.139817	0.620280	7.890413
V13	283726.0	0.000606	1.000000	-5.818474	-0.650837	-0.012986	0.666223	7.159605
V14	283726.0	0.000265	1.000000	-20.178556	-0.447097	0.052728	0.517042	11.055030
V15	283726.0	0.001140	1.000000	-4.917451	-0.635540	0.053885	0.710579	9.703578
V16	283726.0	0.001330	1.000000	-16.172501	-0.534350	0.076822	0.599192	19.818226
V17	283726.0	0.000202	1.000000	-29.866565	-0.574390	-0.078180	0.473553	10.983319
V18	283726.0	0.001809	1.000000	-11.343445	-0.594731	-0.002558	0.599438	6.020067
V19	283726.0	-0.000325	1.000000	-8.868598	-0.560980	0.004140	0.563708	6.874992
V20	283726.0	0.000243	1.000000	-70.777709	-0.274641	-0.080980	0.173000	51.197027
V21	283726.0	-0.000512	1.000000	-48.114286	-0.315378	-0.040669	0.257206	37.577686
V22	283726.0	-0.000021	1.000000	-15.089554	-0.749016	0.009212	0.729066	14.496009
V23	283726.0	0.000318	1.000000	-71.841533	-0.259264	-0.017891	0.236889	36.120452
V24	283726.0	0.000354	1.000000	-4.683788	-0.585267	0.067724	0.726088	7.569926
V25	283726.0	-0.000446	1.000000	-19.752486	-0.609119	0.031231	0.672781	14.426891
V26	283726.0	0.000310	1.000000	-5.403038	-0.677858	-0.108228	0.498413	7.296596
V27	283726.0	0.004455	1.000000	-57.020918	-0.178501	0.003736	0.230473	79.880447
V28	283726.0	0.001668	1.000000	-47.039123	-0.161018	0.034411	0.238627	103.186167
Amount	283726.0	88.472687	250.399437	0.000000	5.600000	22.000000	77.510000	25691.160000
Class	283726.0	0.001667	0.040796	0.000000	0.000000	0.000000	0.000000	1.000000

Fig. 21. Table of dataset statistics

The first column indicates the number of values in each variable. The mean and standard deviation values indicates the range of variation for each value. Min and max provide the least and maximum values. Figure 2 displays descriptive statistics values for each variable, indicating their range of variation. The first quartile is 25%, the median is 50% and the third quartile is 75%. These three metrics will provide distribution information for the dataset.

b) Countplot: Countplot is a type of plot used in data visualization that displays the count of observation in each category of a categorical variable.

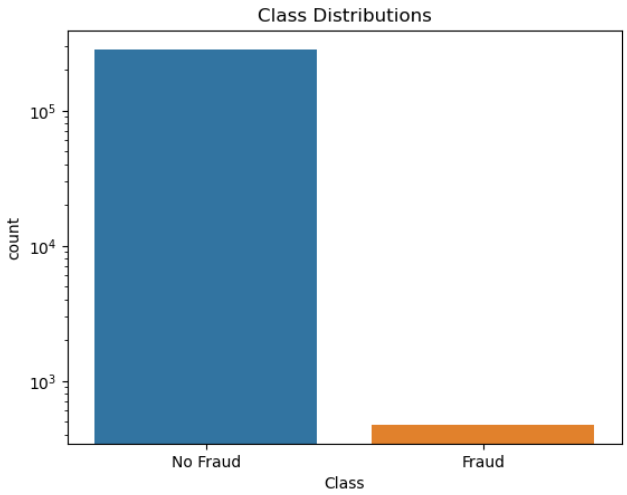

Fig. 22. Count plot of Class (Target variable)

Figure 22 depicts the visualization of the target variable (Class). From this we can understand that, the data is imbalanced. To solve this problem, we are implementing some methods in the project. Will discuss it later.

VIII. Data Preparation

Data preparation, often known as preprocessing, is a key step in machine learning procedures. It entails transforming raw data into a format suitable for training machine learning models.

VIII.1 Handling missing values and duplicates

For handling missing values we are using isna().sum() function to verify is there are any missing values in the dataset.

```
df.isna().sum()
```

```
Time      0
V1        0
V2        0
V3        0
V4        0
V5        0
V6        0
V7        0
V8        0
V9        0
V10       0
V11       0
V12       0
V13       0
V14       0
V15       0
V16       0
V17       0
V18       0
V19       0
V20       0
V21       0
V22       0
V23       0
V24       0
V25       0
V26       0
V27       0
V28       0
Amount    0
Class     0
dtype: int64
```

Fig. 23. Details of missing values

From the above figure, we can understand that there is no missing values in the data.

After handling missing values, we check any duplicate data found in the data using duplicated().sum(). As a result of using the function, we got zero duplicate data.

VIII.2 Feature engineering

Multicollinearity occurs when two or more predictor variables in a regression model are substantially associated with one another. In other words, multicollinearity develops when there are linear correlations between the independent variables.

VIF (Variance Inflation Factor) is a measure used to assess the severity of multicollinearity in a regression model. It estimates how much the estimated regression coefficient variance is increased as a result of model multicollinearity.

Here we are going to use VIF to find out the multicollinearity of the data.

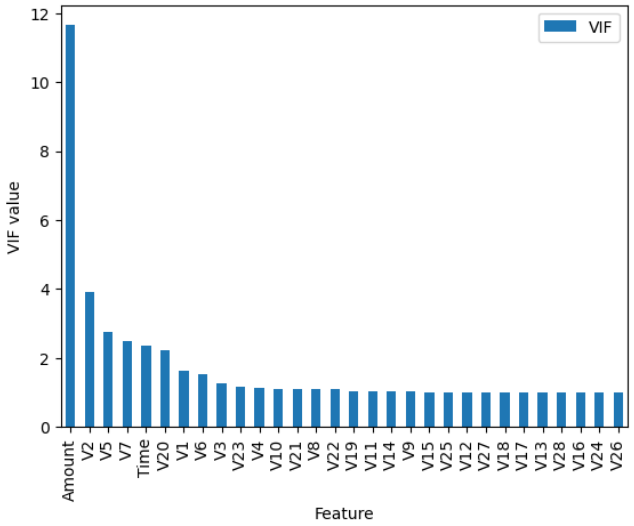

Fig. 24. Multicollinearity of variables

From the above figure, we can understand that 'Amount' has high multicollinearity. For a good model, VIF should be less than 5. so we will have to transform our data.

We cannot use log transformation as there are a lot of negative values in every row of our data. So we are using the Yeo-Johnson method to do transformation.

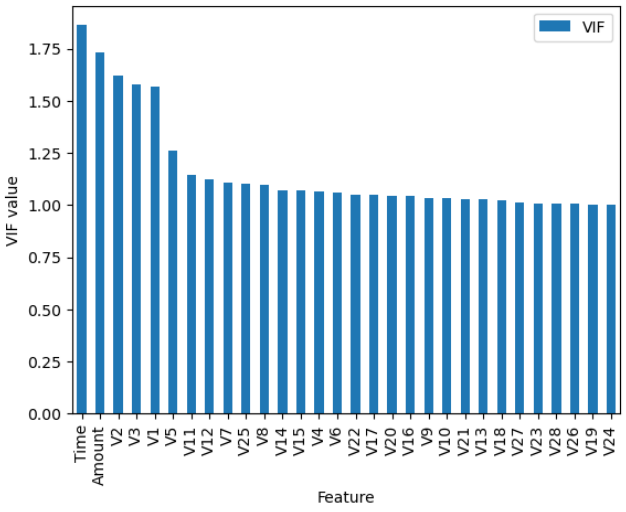

Fig. 25 Multicollinearity of data after transformation

The above figure shows the visualization of multicollinearity after using transformation.

VIII.3 Data Normalization

Data normalization is a preprocessing technique that rescales numerical features in a dataset to a common scale.

In this data, we have already done the normalization. So it is suitable for data splitting.

VIII.4 Data Splitting

The train_test_split function is used in this method to divide the data into training and testing sets. We split 70% of the data for training and 30% for testing.

IX. Model Evaluation and Result

IX.1 Model 1

In Model 1 we are using basic Logistic regression to generate the model. The confusion matrixes display of model 1 is shown below:

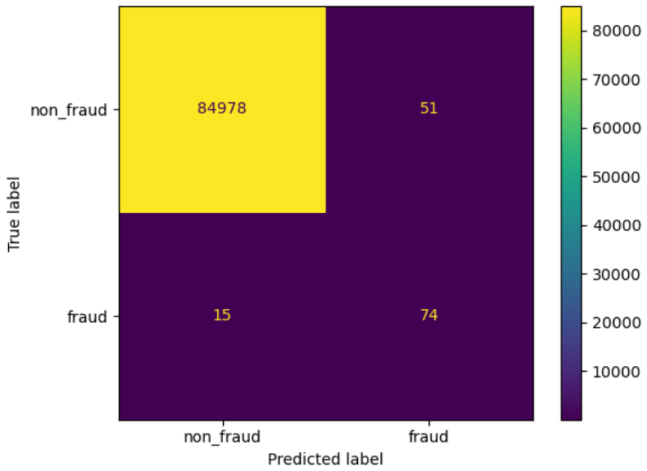

Fig. 26 Confusion matrix of model 1

In this model, we are finding accuracy, precision, recall, and f2_score. The result of this model is mentioned below:

Metric	Score
Accuracy	0.999225
Precision	0.831461
Recall	0.592
F2 Score	0.628183

Fig. 27 Metrics of Model 1

The below graph shows the Precision recall curve of model 1.

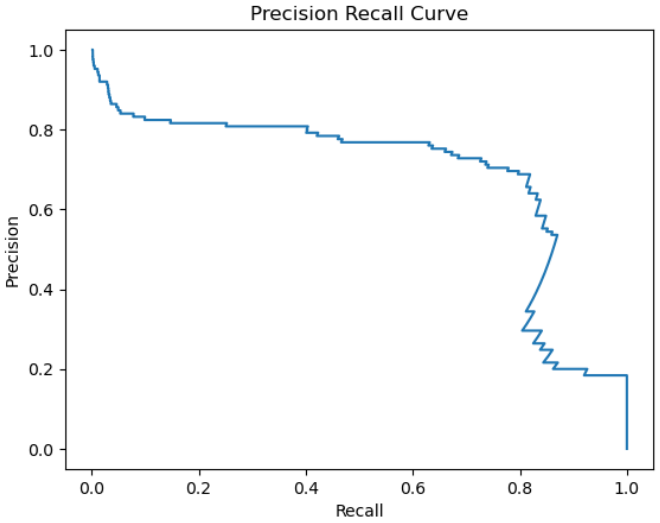

Fig. 28 Precision recall curve of Model1

We got Recall as 0.592 and Precision as 0.83. From this, we can understand the value of precision is high compared to recall. It means that the model classifies a lot of fraudulent transactions as non-fraud.

IX.2 Model 2

In Model 2 we are balancing the data. From the model 1, we get low precision and recall because of unbalanced data. So, in this model, we are using the SMOTE function from imblearn to balance the data. Then we are using the logistic regression model to find the metrics. The confusion matrixes display of model 2 is shown below:

Fig. 29 Confusion matrixes of Model 2

Metric	Score
Accuracy	0.947273
Precision	0.969998
Recall	0.923172
F2 Score	0.932172

Fig. 30. Metrics of Model 2

By evaluating the result, we got a better Precision and Recall value. The below figure shows the Precision-Recall curve of model 2.

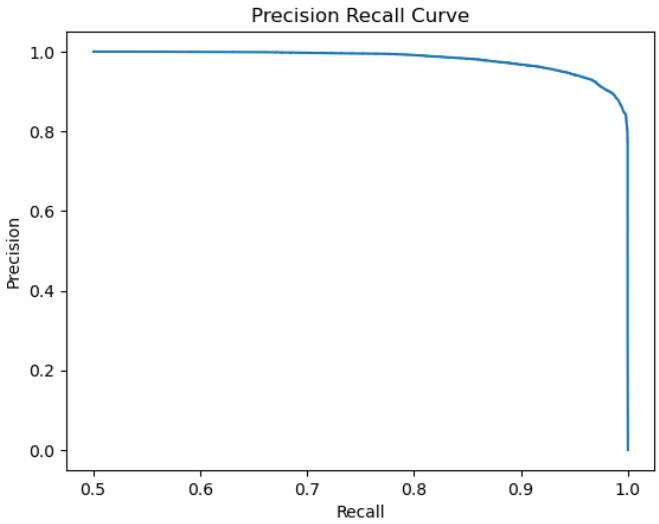

Fig. 31 Precision-Recall curve of model 2

IX.3 Model 3

In Model 3 we are using a basic Random Forest Classifier to generate the model. The confusion matrixes display of model 3 is shown below:

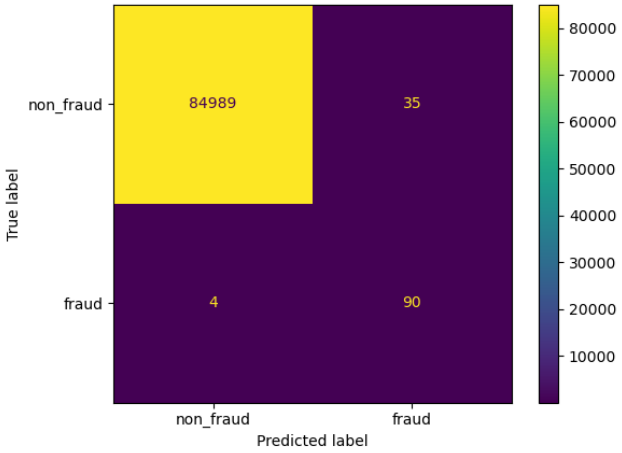

Fig. 32 Confusion matrixes of Model 3

In this model, we are finding accuracy, precision, recall, and f2_score. The result of this model is mentioned below:

Metric	Score
Accuracy	0.999542
Precision	0.957447
Recall	0.72
F2 Score	0.757576

Fig. 33 Metrics of Model 3

The below graph shows the Precision recall curve of model 3.

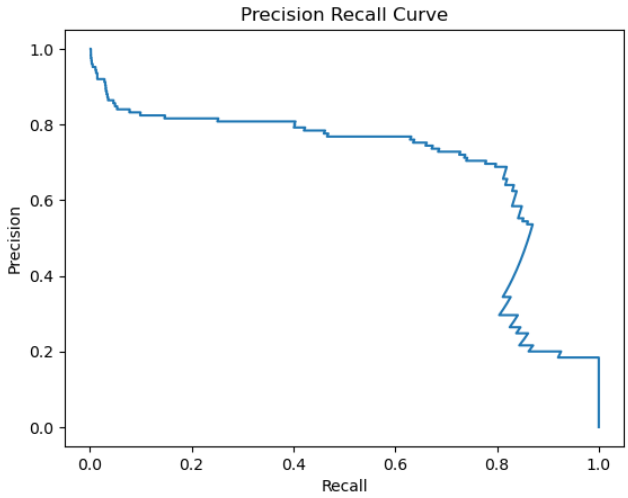

Fig. 34 Precision recall curve of Model 3

We got Recall as 0.72 and Precision as 0.95. From this, we can understand the value of precision is high compare to recall. It means that the model classifies a lot of fraudulent transactions as non-fraud.

IX.4 Model 4

In Model 4 we are balancing the data. From the model 1 and 3, we get low precision and recall because of unbalanced data. So, in this model, we are using the SMOTE function from imblearn to balance the data. Then we are using the Random Forest Classifier model to find the metrics. The confusion matrixes display of model 4 is shown below:

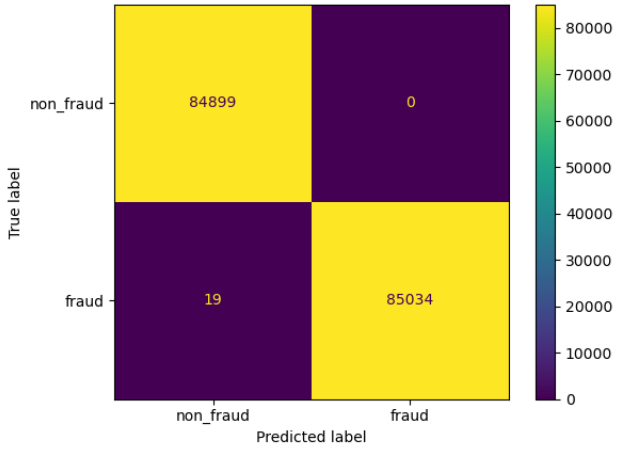

Fig. 35 Confusion matrixes of Model 4

Metric	Score
Accuracy	0.999888
Precision	1
Recall	0.999777
F2 Score	0.999821

Fig. 36. Metrics of Model 4

By evaluating the result we got a better Precision and Recall value. The below figure shows the Precision-Recall curve of model 4.

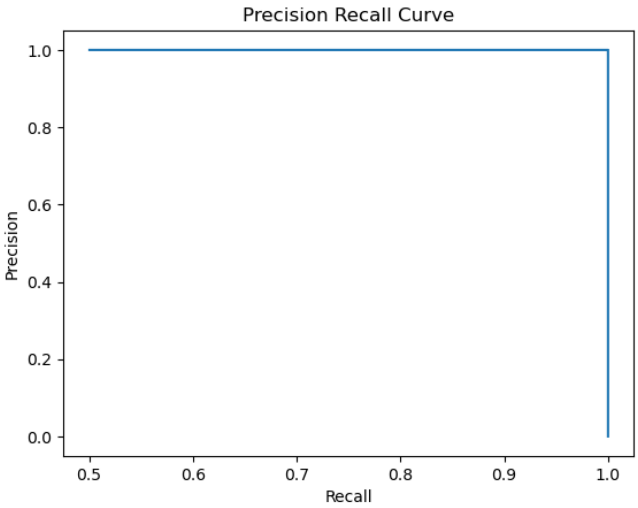

Fig. 37 Precision-Recall Curve of Model 4

X. Conclusion

In conclusion, our study solves the issue of balanced data in credit card fraud detection by applying a variety of models and methodologies. We explored four models: Basic logistic regression, Logistic regression with SMOTE oversampling, Basic Random Forest Classifier, and Random Forest Classifier with SMOTE oversampling. Our evaluation metrics included Precision, Recall, Accuracy, and F2-score providing a good picture of model performance.

After evaluation, logistic regression with SMOTE and random forest with SMOTE consistently outperformed their simple counterparts. These models showed enhanced capabilities in correctly recognizing fraudulent transactions, which is critical for minimizing financial losses and preserving client security. The use of SMOTE for data balance greatly improved the model's capacity to generalize to the minority class, resulting in improved overall performance.

XI. Reference

[1] https://www.analyticsvidhya.com/blog/2022/05/a-comprehensive-guide-to-time-series-analysis-and-forecasting/

[2] https://www.wisdomgeek.com/development/machine-learning/sarima-forecast-seasonal-data-using-python/,

[3] https://thesai.org/Downloads/Volume11No12/Paper_65-Fraud_Detection_in_Credit_Cards.pdf

[4] https://www.icco.org/,