8/23/2022

HDSC Summer ’22 Premiere Project Forecasting Foreign Exchange Rate

TEAM SELENIUM

INTRODUCTION

Planning a forex trading system requires a meticulous design and extensive testing. A very common indicators to travelers and investors are the exchange rates. Foreign exchange rates of a nation’s currency have a major impact on the lives of its citizens.

Time series is an observation of data items made at an equally spaced time interval. For example, monthly salary data of employees, stock prices, etc. Time-series analysis is used when observations are made repeatedly over 50 or more time periods and these observations are time dependent.

This dataset used for this project was generated on Kaggle. The dataset contains some currencies with their rates in dollar units. We used three models with the dataset and these models’ performances were evaluated using different time series forecasting performance measures by testing how well these models have predicted with respect to the actual observations. We also compared and determined which one of these models will be able to make a better forecast comparing the results from the performance evaluation methods.

PROBLEM STATEMENT

With the hike in foreign exchange rates, 37 of 44 advanced economies have experienced double rates of inflation with consumer prices rising since pre-pandemic times according to Pew Research Centre analysis. Beside price distortion, the fluctuation of the foreign exchange rates erodes savings, discourages investment, inhibit economic growth and in extreme cases causes social and political unrest. Understanding and forecasting the future values of exchange rates is necessary.

AIMS AND OBJECTIVES

To determine how United Kingdom pounds (GBP) changes over time with United States Dollar (USD) using the dataset and predict future exchange rate of GBP using Regression, ARIMA, FBPROPHET for a period of 30 days and comparing it to the original values and determine which one of these models has a better forecast performance.

FLOW PROCESS

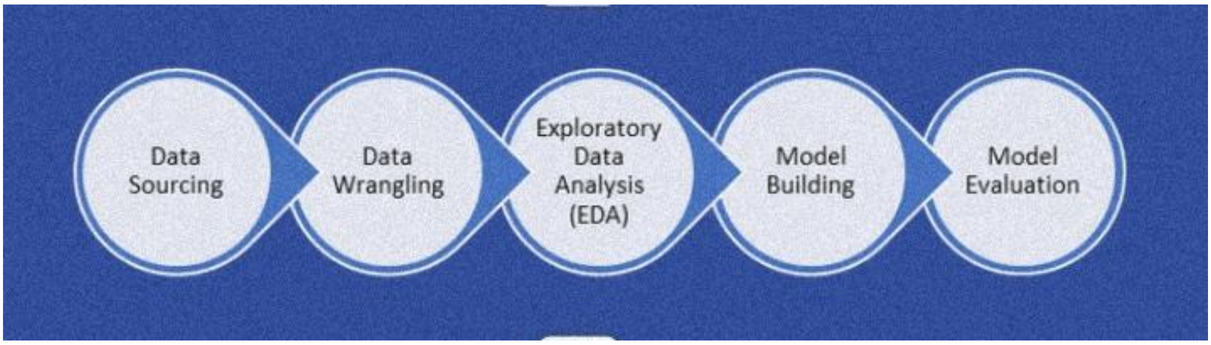


Fig 1. Project flow process

Data Sourcing:

This is the process of gathering data and getting them ready for analysis. Some datasets are already prepared while others require extra data wrangling to make them ready for analysis. For this project we were able to get a foreign exchange dataset from Kaggle which requires little data wrangling.

Jupyter notebook was used for this Project. Python and its libraries were used in carrying out the analysis. Python's Pandas library was used to load the dataset as a comma separated values (.csv) file format which has already downloaded from Kaggle to our local machine.

Data Wrangling:

Data preparation and transformation was carried out to make sure that the observations collected in the dataset are clean and tidy. We must ensure that the data is correct, incorrupt and usable by identifying errors and outliers and correcting them to prevent errors and unintended bias during analysis.

After loading the dataset, we performed a few assessments to check the state and usability of our data. We identified some descriptions and information about the dataset using some Pandas method. Some of the finding from the assessment are as follows.

1. In the original dataset, we discovered that there is a total of 5217 rows and 24 columns using pandas ***.shape()*** method.
2. We identified an unknown column (Unnamed: 0) was created while loading the dataset using pandas ***.info*** method. We dropped that column as it was not useful for our analysis using the .***dropna()*** method.
3. Since our scope of analysis is just for United Kingdom Pound (GBP) exchange rate, we created a new data frame to store the values for this dataset. This includes the exchange rates and the time stamp.

4. There is an unknown variable **‘ND’** in place of the rate values. The possible meaning could be no ‘No Data’, so we have dropped all rows with such values to make the dataset usable.

5. On looking at the datatype for our variables using the **.info()** method, we see that both the variables are of “object” type. In order to perform our analysis, we’ll have to change the datatypes of the two columns. So, we’ll change the exchange rate to numeric using the ***.to_numeric()*** method of pandas and the date column to date time using the ***.to_datetime()*** method.

EXPLORATORY DATA ANALYSIS

Since the scope of analysis is just for United Kingdom Pound (GBP) exchange rate, we created a new data frame to store the values for this dataset. This includes the exchange rate and the time stamp.

Distribution of UK Exchange Rates:

To understand the distribution of UK Pounds Exchange rate over the years we used seaborn displot how this currency is distributed over time. As shown in Fig. 1 below, the frequency of observations in each bin provides insight into the underlying distribution of the UK pounds. We discovered that the UK pounds exchange rate closely looks like gaussian distribution.

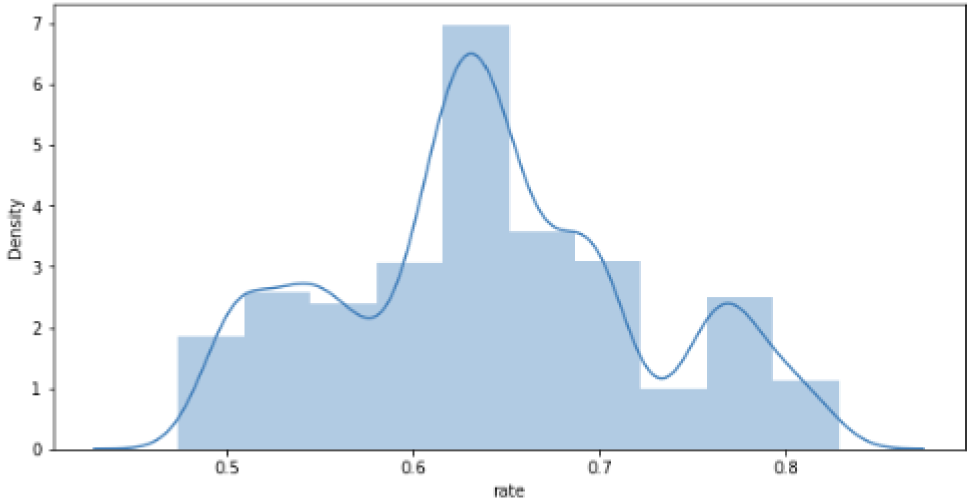


Fig 2: density chart of UK Exchange Rates

Time series line plot

In this plot, time is shown on the x-axis with observed values of data (UK pounds) is shown along the y-axis.

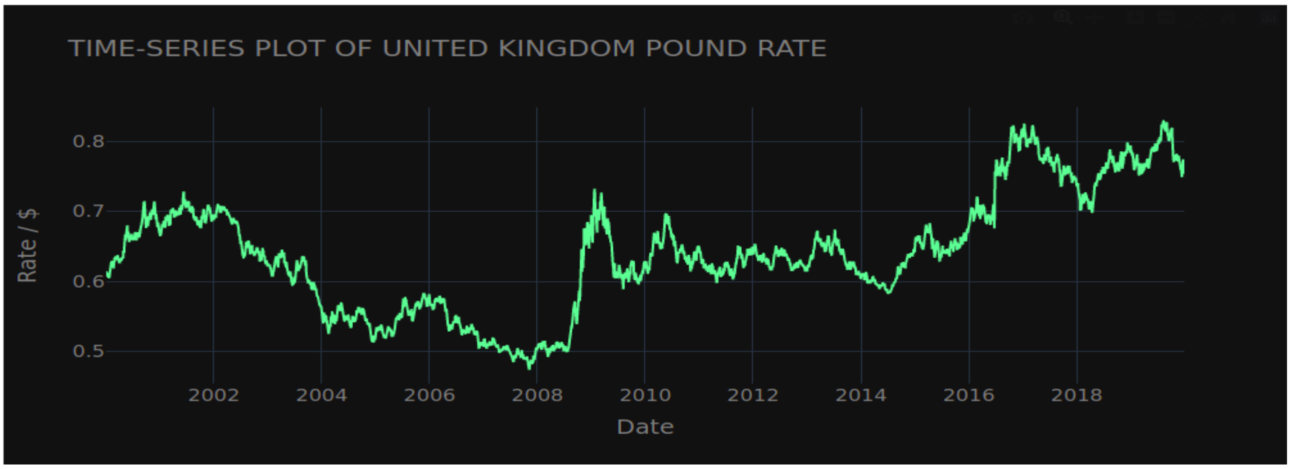
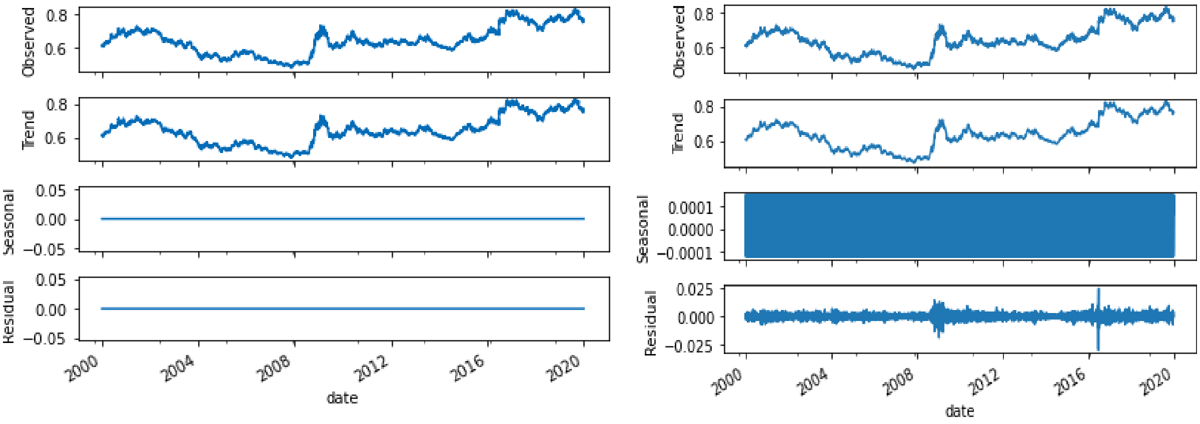


Fig 3: Time Series Plot

On looking the line plot of the data, we can see from the data that the statistical property change with time and the mean is time dependent which means this time series is observed to be non-stationary. Periodic fluctuations were not noticed which indicates that there is no observable seasonality.

We then use the seasonal decompose() method from statsmodel library to break down the time-series into 3 components, namely random residuals, linear trend, and the periodic component (seasonality). We choose additive model for this seasonal decomposition instead of multiplicative model since there is no exponential increase in the amplitudes over time. We also tried the decomposition at two different frequency parameters 1 and 5.



a. **frequency at 1**

b. **frequency at 5**

Fig 4: Plots for Seasonal Decomposition with frequency at 1 and 5 respectively.

From the chart above the following findings were made about the components,

1. We see that there is residual randomness and linear trend in the data.

2. We also noticed that there is no periodic or seasonality component.

After seasonal decomposition we plotted a Lag plot to explore the relationship between each observation and a lag of that observation. A lag plot plots the observation at time t on the x-axis and the observation at the previous time step (t-1) on the y-axis. The values were plotted for a total of 7 lags.

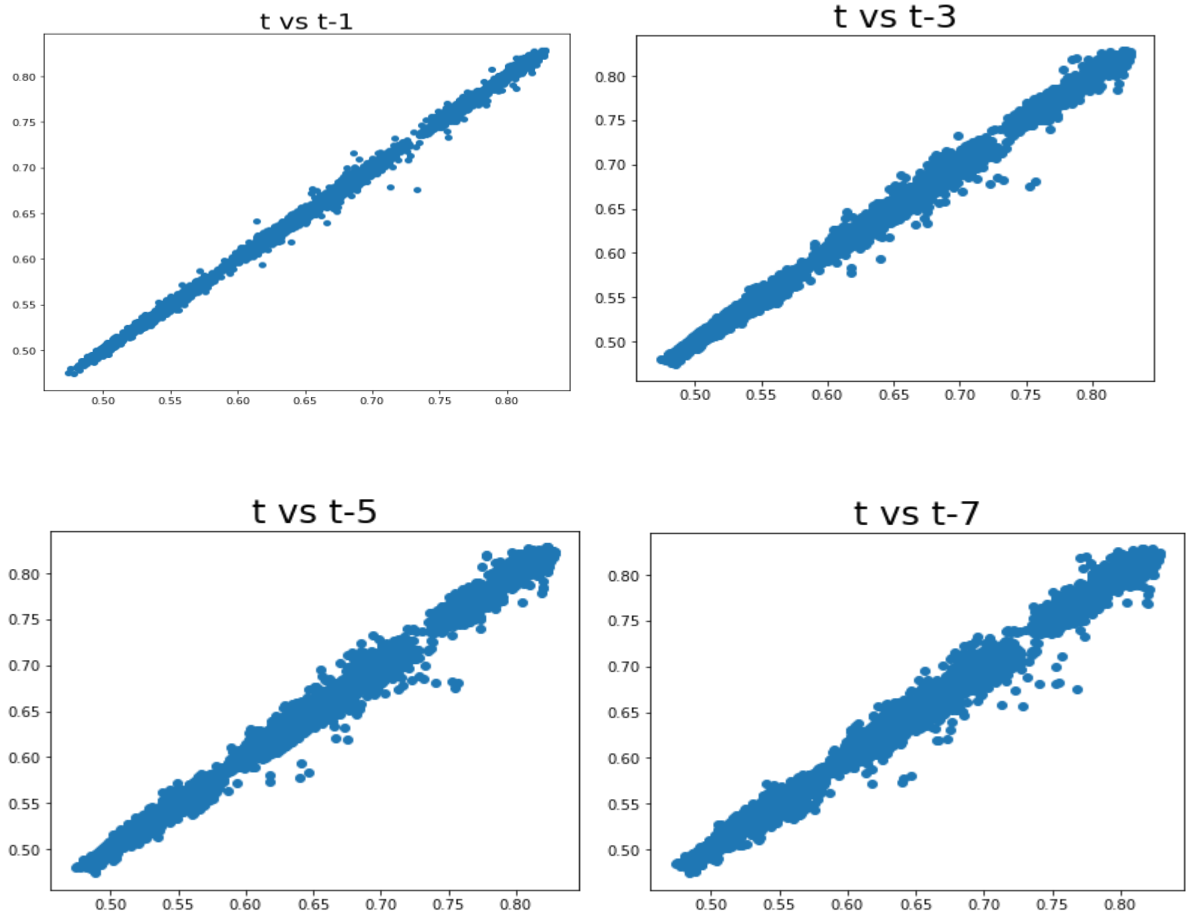


Fig 5: Lag Scatter Plot for selected lags.

The values were plotted for a total of 7 lags. We noticed that the points cluster along a diagonal line from the bottom-left to the top-right of the plot, it suggests a positive correlation relationship between the values at time t and each lag period.

To observe the similarity between observations as a function of the time lag between them, Autocorrelation plot was employed to capture the relationship of an observation with past observations in the same and opposite time interval.

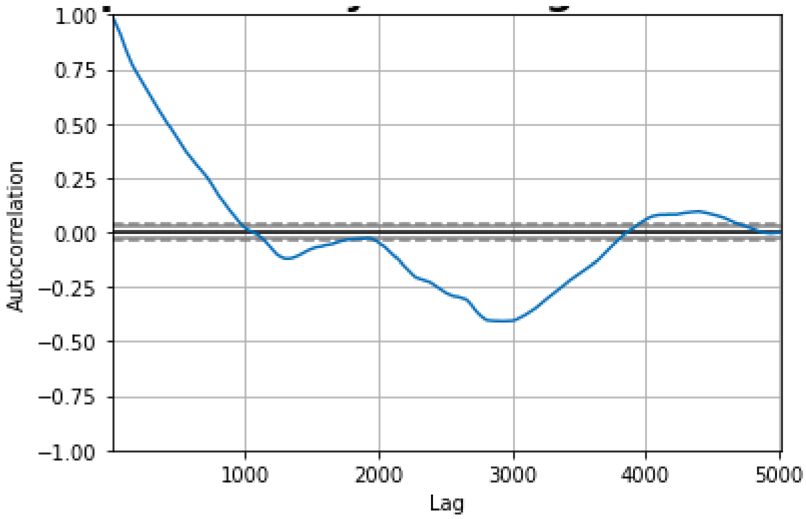
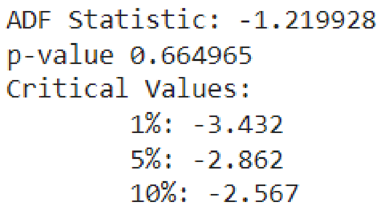


Fig 6: Autocorrelation Plot of GBP data series.

From the autocorrelation plot, it was observed that the initial lag values show a correlation with previous values. However, it starts decreasing as time passes and becomes negative for a while. like a realistic time series in shape and movement. The plot reveals the expected trend across the first two thousand lag observations. For most part of it, it remains close to zero for higher values, which shows no correlation.

Since we know by the observation from our dataset that it is non-stationary, we'll need some empirical proof. We'll now run Augmented Dickey Fuller Test (ADF Test) which is a test for stationarity. The null hypothesis of the Augmented Dickey-Fuller test is that the time series is non-stationary if the test statistic value i.e., p-value > 0.05 threshold. So, we can reject this hypothesis by getting p-value that is approximately zero i.e., p-value < 0.05 threshold confirms that time series is stationary which is less than 5% level of significance.

After running the ADF test, we can see that the time series is non-stationary as the p-value obtained is greater than the 0.05 threshold as shown in figure 8 below.



```
ADF Statistic: -1.219928
p-value 0.664965
Critical Values:
	1%: -3.432
	5%: -2.862
	10%: -2.567
```

Fig 7: ADF Result for stationarity test.

Autocorrelation function (ACF) and Partial Autocorrelation Function (PACF) are important functions in analysing a time series. Both plots are drawn as bar charts showing the 95% and 99% confidence intervals as horizontal lines. Bars that cross these confidence intervals are therefore more significant and worth noting. They produce plots that are essential in determining the values of **p**, **q** and **d** for Autoregressive (AR) and Moving Average (MA) models. An ACF measures and plots the average correlation between data points in time series and previous values of the series measured for different lag lengths. A PACF is similar to an ACF except that each partial correlation controls for any relationship between observations of a lesser lag length. Since we plan using models that require these values the ACF and PACF plots are shown in figure 9 below.

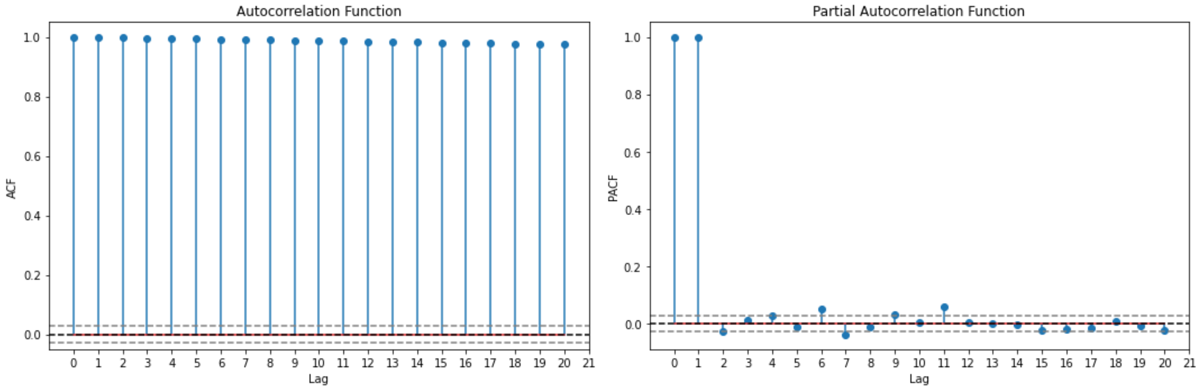


Fig 8: ACF and PACF plots of GBP data series.

The value for an ACF and a PACF at the first lag are the same because both measure the correlation between data points at time **t** with data points at time **t-1**. However, at the second lag, the ACF measures the correlation between data points at time **t** with data points at time **t-2**, while the PACF measures the same correlation but after controlling for the correlation between data points at time **t** with those at time **t-1**.

We noted that our Test Statistic value is greater than our critical value at 5%. Therefore, we can say that our time series is non-stationary. We can also see from our ACF plot that there is high correlation between time series at time t and various other lag values. There is also no sign of any kind of seasonality. We then went on to difference our series once to see if this removes non-stationarity.

In order to transform our dataset and remove the time-dependency, **differencing** method was performed on the series. We performed differencing by taking the first order differences of the observations. We replace each observation with the difference between it and the previous value. Below is the autocorrelation chart after differencing.

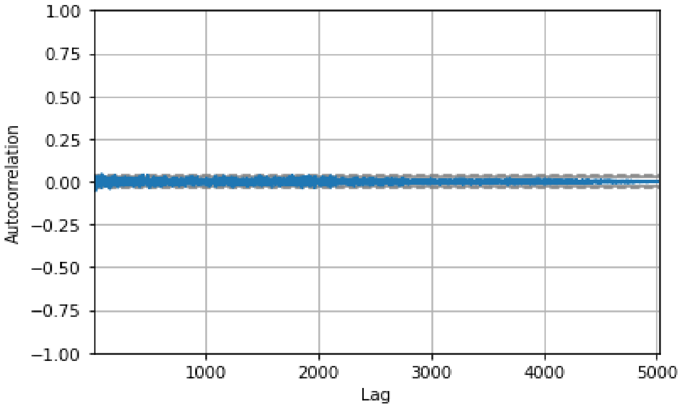


Fig 9: Autocorrelation Plot of GBP data series after differencing.

From the auto-correlation plot, we observe that all correlations are close to zero and they lie below the 95% and 99% confidence levels showing stationarity after differencing. We also noted that that the time series was looking stationary with a mean 0 and a small standard deviation. This shows us that the data has become stationary now.

ACF and PACF plots were produced again as shown below.

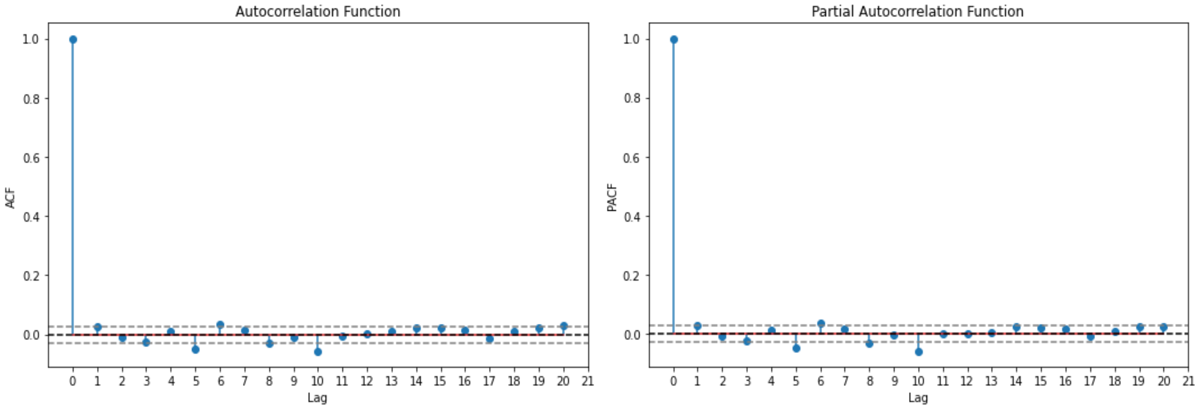
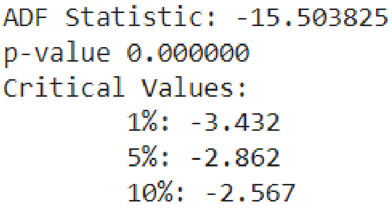


Fig 10: ACF and PACF plots of GBP data series after differencing.

From these plots, we can see that the differenced time series has a mean close to 0 and a very small variance. Also, This is also a sign of stationarity.

We then re-ran the test Augmented Dickey-Fuller (ADF) test to check for stationarity and confirmed that time series is stationary as the p-value of the test is nearly zero and it is less than 5% level of significance.



```
ADF Statistic: -15.503825
p-value 0.000000
Critical Values:
        1%: -3.432
        5%: -2.862
        10%: -2.567
```

Fig 11: ADF Result for stationarity test after differencing.

From here we carried on to modelling.

MODELLING

In order to build or models, we have done all the prerequisites steps that are required. We have performed extensive EDA, and also made our time series stationary. We'll now be looking to build 3 models, namely Linear Regression, ARIMA, and Prophet models.

Linear Regression

Linear regression algorithm learns how to make a weighted sum from its input feature. The baseline model for this analysis is linear regression. It uses basic statistics to project future values for a target variable. The basic concept of regression in Time Series work on creating new column with values starting from 1 till end depending upon the number of datapoints available in our dataset, and then regressing our dataset over that generated column.

The dataset was divided into two during the analysis. The first 70% was used for training and the last 30% was used for testing for this model.

Arima Model

When it comes to time series predictions, ARIMA is commonly used. ARIMA stands for Auto-Regressive Integrated Moving Average and the equation is used to predict forecasted values. This model is based on the ideology that the information in the past values of the time series can alone be used to predict the future values.There a three terms used in delineating Arima model which are the **p**, **q** and **d**. Identification of a time series is the process of finding integer, usually very small (e.g., 0, 1, or 2), values of *p*, *d*, and *q* that model the patterns in the data. The middle element, *d*, is investigated before *p* and *q.* The goal is to determine if the process is stationary and, if not, to make it stationary by differencing, where d is the number of differencing before determining the values of *p* and *q.* The auto-regressive components represent the memory of the process for preceding observations. The value of *p* is the number of auto-regressive components in an ARIMA (*p*, *d, q*) model. The value *q* indicates the number of moving average components in an ARIMA (*p, d, q*). When *q* is zero, there are no moving average components. When *q* is 1, there is a relationship between the current score and the random shock at lag 1.

We know, the value of p is obtained from the PACF plot, d is the number of times we have differenced our time-series (which is 1 in this case), and q is obtained from the ACF plot.

On plotting our ACF and PACF in Fig. 7 plots with the differenced data (also called as lollipop charts), we can see that the p and q values turn out to be 10 for both. We identify the p and q values by identifying where the lollipops are out of the band of error, which shows the significant values. Any value inside the band is treated as insignificant.

We built the Arima models, and obtained the Mean Absolute Percentage Error (MAPE) for the model.

Facebook Prophet

The Facebook prophet is a model developed by Facebook which make time Serie prediction easy. The prophet model requires a data frame with a y and ds columns on which we want to experiment. The model function takes data and then predict a period that has been specified too this process is repeated until an end point is reached.

We called the Prophet model and passed our dataframe into it with the Time Serie as **ds** and UK exchange rate values as **y**.

RESULTS

After using the three models we performed performance evaluation, we carried out performance evaluation of the models using 4 evaluation metrics which are Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Squared Error (MSE). The following is a summary of the result from our analysis.

1. We observe that there is a considerate gap between the actual and forecasted values. The mean absolute percentage error for the linear regression model is 16.82% which will be our baseline percentage error.Also MAE is 0.126,RMSE is 0.146 and MSE is 0.021

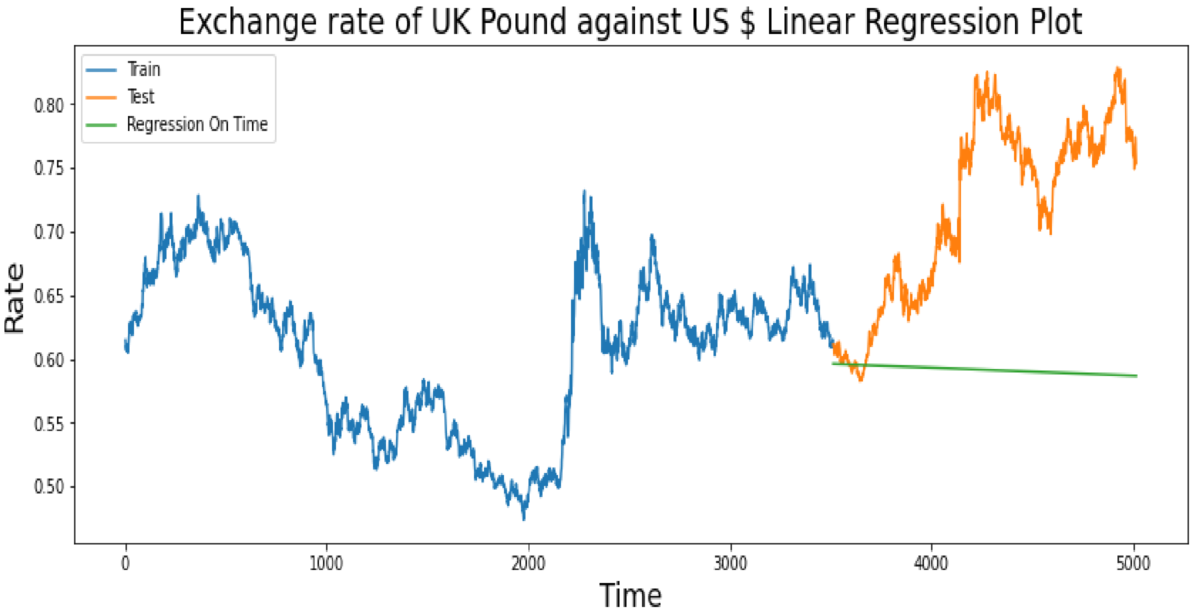


Fig 12: Result Chart for the linear regression model

2. On running the ARIMA model, and forecasting values at daily basis, we observe that there is a little gap between the actual and forecasted values. we discovered that the MAPE obtained in this case was 1.194%.Also MAE is 0.021,RMSE is 0.0117and MSE is 0.00013

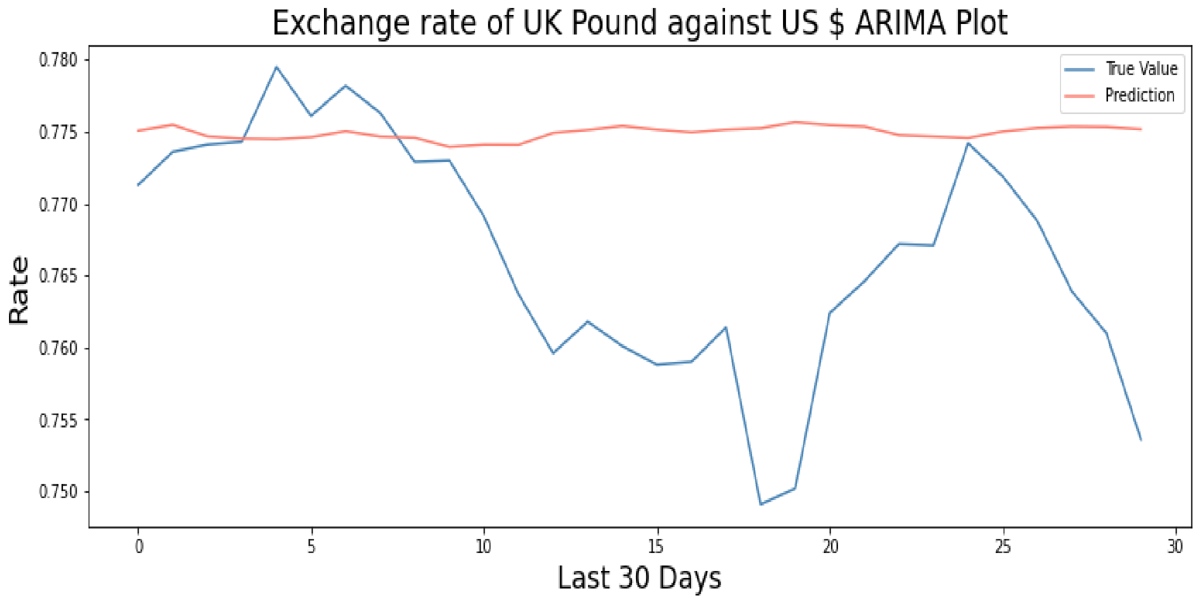


Fig 13: Result Chart for the ARIMA model

3. Then we discovered that MAPE for the prophet model is 3.302% which also show a small gap between the actual and predicted values but not as small as the ARIMA model. Also MAE is 0.0255, RMSE is 0.03 and MSE is 0.000194

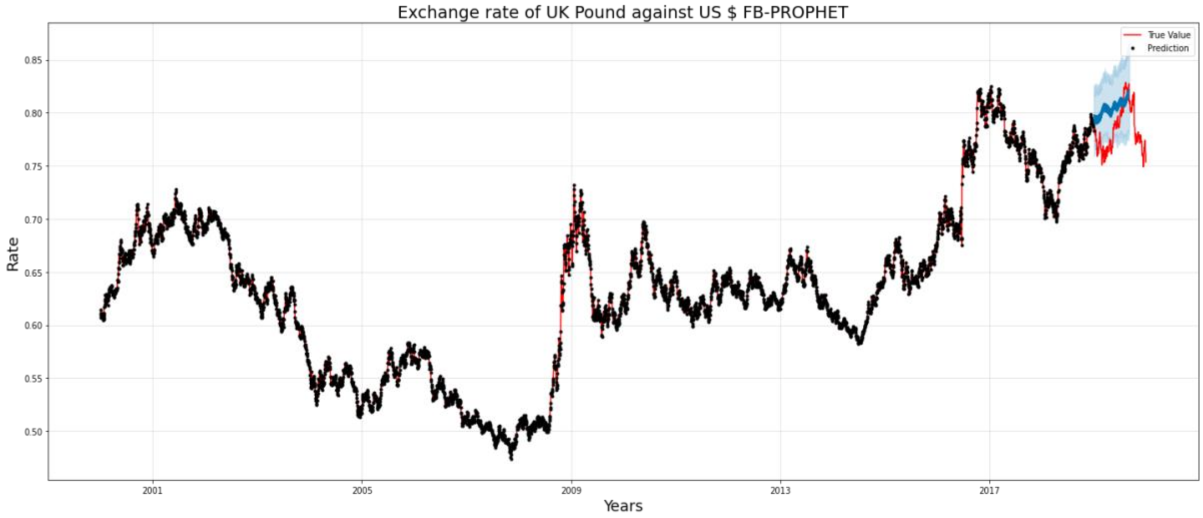
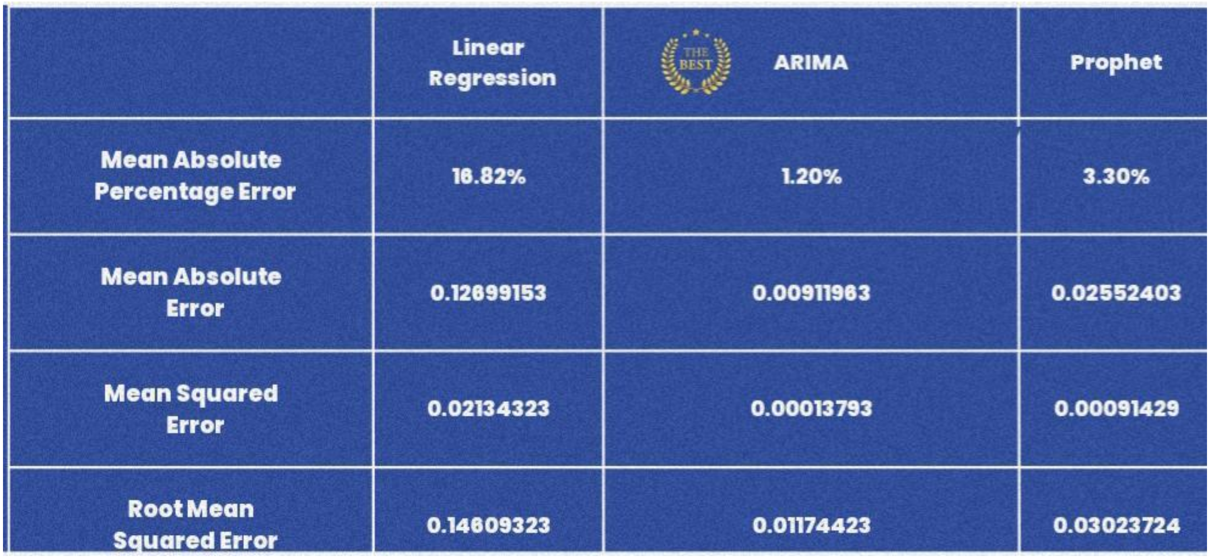


Fig 14: Result Chart for the FBprophet model

Performance evaluation results table



	Linear Regression	THE BEST ARIMA	Prophet
Mean Absolute Percentage Error	16.82%	1.20%	3.30%
Mean Absolute Error	0.12699153	0.00911963	0.02552403
Mean Squared Error	0.02134323	0.00013793	0.00091429
Root Mean Squared Error	0.14609323	0.01174423	0.03023724

Table 1 performance evaluation results table.

CONCLUSIONS

The impact Foreign Exchange market has on people's finances and the general economy can never be over emphasized. To be on the profit end, the need to get correct insights and make accurate predictions of Foreign Exchange values is extremely important.

From the result obtained using the three models, we discovered that ARIMA model has the lowest of all the performance evaluation metrics used, which means it has the highest accuracy. According to our Analysis. ARIMA Model predicted a 30days exchange rate and thus, the UK currency is forecasted to depreciate as against USD.

LIMITATIONS

The dataset contains observation with 'ND' as values, we had to drop this observation which might have affected our result.
Foreign Exchange market is a peculiar market. Our prediction is based on the data we have gathered, so we can't guarantee that this prediction will always be precise, as there might be other conditions that may affect the Foreign Exchange rates in the future such as recession, war, natural phenomenon, etc., that can affect the stability of the market.

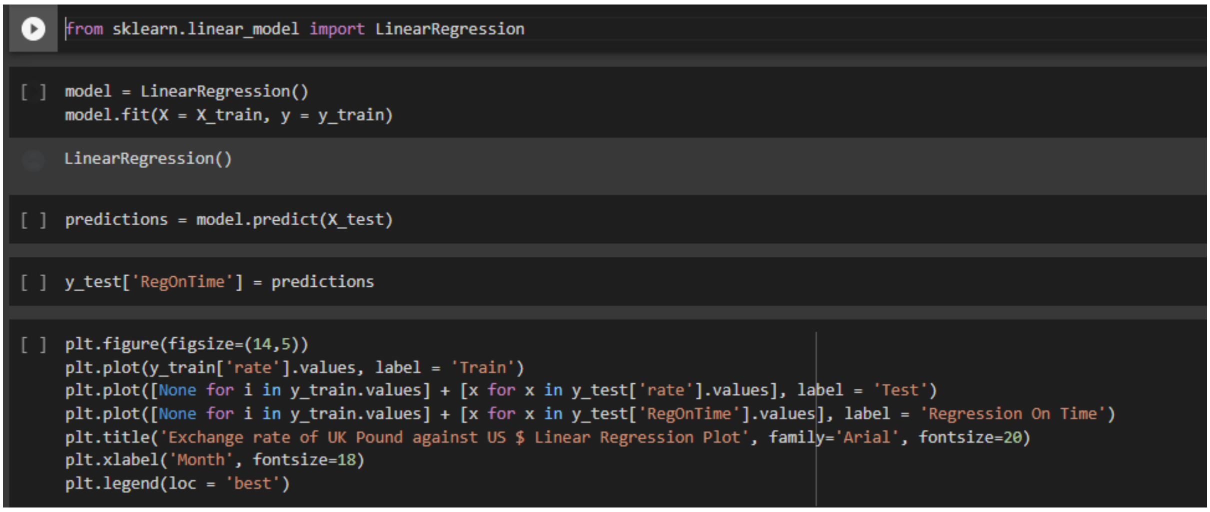
Click this link to access the project's Jupyter notebook.

TEAM MEMBERS

1. Abdulhamid Ibrahim
2. Ikhuohon-Eboreime Rhoda
3. Abdullahi Isa
4. Vishal Garg
5. Ayobami Olanrewaju
6. Blessing Ojo
7. Chimuanya Reina Samuel
8. Chizoba Chukwuemekw
9. Heba Allah Hashim
10. Hennry Mbom
11. John SSolomo
12. Joshua Obikunle
13. Joy Abiodun
14. Martha Adekoya-Cole
15. Mba Chinenye Juliet
16. Miracle Uche
17. Musa Badaru
18. Nelson Onaiwu
19. Nonso Ebele-Muolokwu
20. Olajide kuti
21. Saheed Oseni
22. Amarakro Prebor
23. Veeraraghavan Vijayarajan

APPENDICES

Appendix 1. Linear Regression modelling code snippet



```
from sklearn.linear_model import LinearRegression
```

```
model = LinearRegression()
model.fit(X = X_train, y = y_train)
```

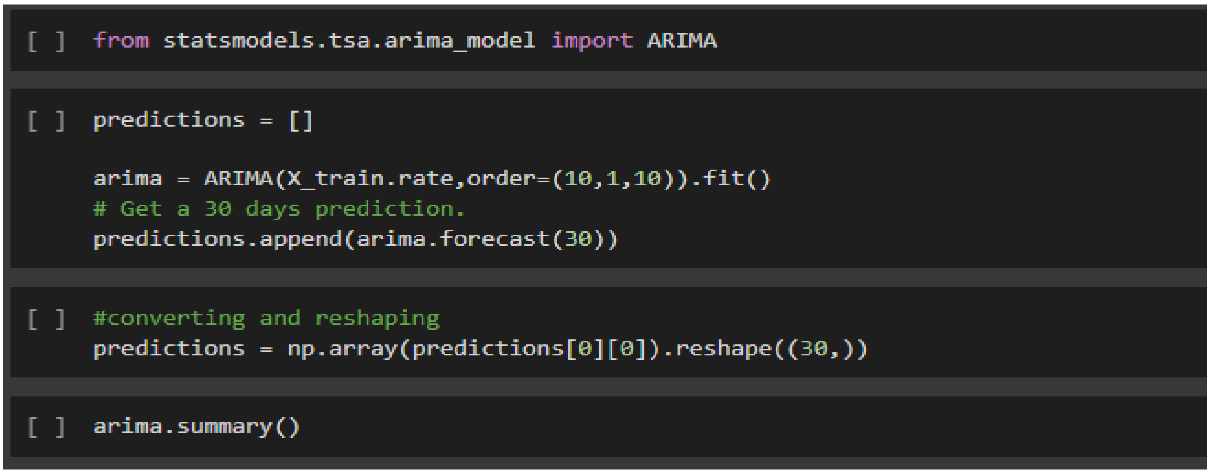
```
LinearRegression()
```

```
predictions = model.predict(X_test)
```

```
y_test['RegOnTime'] = predictions
```

```
plt.figure(figsize=(14,5))
plt.plot(y_train['rate'].values, label = 'Train')
plt.plot([None for i in y_train.values] + [x for x in y_test['rate'].values], label = 'Test')
plt.plot([None for i in y_train.values] + [x for x in y_test['RegOnTime'].values], label = 'Regression On Time')
plt.title('Exchange rate of UK Pound against US $ Linear Regression Plot', family='Arial', fontsize=20)
plt.xlabel('Month', fontsize=18)
plt.legend(loc = 'best')
```

Appendix 2. ARIMA modelling code snippet



```
from statsmodels.tsa.arima_model import ARIMA
```

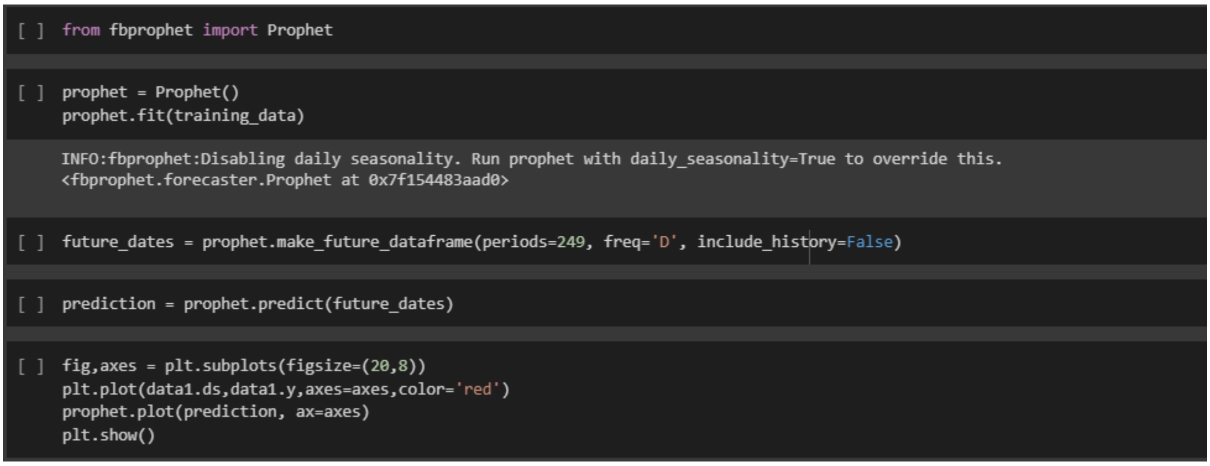
```
predictions = []

arima = ARIMA(X_train.rate,order=(10,1,10)).fit()
# Get a 30 days prediction.
predictions.append(arima.forecast(30))
```

```
#converting and reshaping
predictions = np.array(predictions[0][0]).reshape((30,))
```

```
arima.summary()
```

Appendix 3. FBprophet modelling code snippet



```
from fbprophet import Prophet
```

```
prophet = Prophet()
prophet.fit(training_data)
```

```
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
<fbprophet.forecaster.Prophet at 0x7f154483aad0>
```

```
future_dates = prophet.make_future_dataframe(periods=249, freq='D', include_history=False)
```

```
prediction = prophet.predict(future_dates)
```

```
fig,axes = plt.subplots(figsize=(20,8))
plt.plot(data1.ds,data1.y,axes=axes,color='red')
prophet.plot(prediction, ax=axes)
plt.show()
```