Time Series Analysis Project: Paypal Stock Price

Karla Aniela Cepeda Zapata
Department of Informatic and Mathematics
Dundalk Institute of Technology
Co. Louth, Ireland
d00242569@student.dkit.ie

I. Introduction

Time series are everywhere. It is simply a series of data points order by time, and a statistical approach to deal with trend analysis. Examples of time series are: tracking daily temperature, tracking changes in some application performance, real-time data from medical devices, and so on. In this assessment, it is required to visualize, perform a diagnostic analysis, and carried out informed predictions on a specific Time Series asset.

The framework of this project was based on the CRISP-DM methodology. Figure 1 shows the whole life cycle of the project.

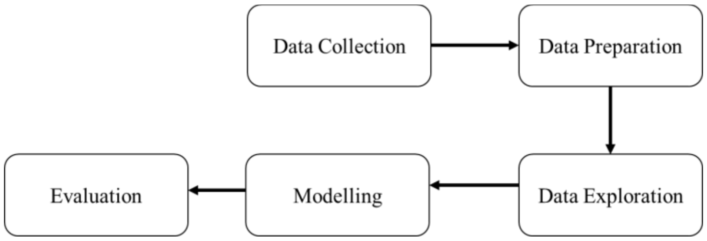


Figure 1.Life cycle of the project.

The technology implemented in this project is enlisted in TABLE I.

Technology
For Data Collection and Preparation
Programming language: Python (IDE Spyder) Libraries: • pandas, numPy • yFinance • sqlalchemy, psycopg2
For storing
PostgreSQL (local instance)
For exploration and analysis
Programming language: Python (IDE Jupyter Notebook) Libraries: • pandas, numpy • matplotlib, plotly, seaborn • statsmodels • sqlalchemy
For repository
Git

TABLE I. Description of final dataset

This paper is structured as follows: (II) DATA DESCRIPTION, (III) DATA COLLECTION AND PREPARATION, (IV) DESCRIPTIVE STATISTICS, (V) RESEARCH QUESTIONS, (VI) RESULTS, and (VII) CONCLUSIONS.

II. Data Description

A. *Description of the dataset*

The area selected to carry out this project is the Stock Market. The ticker chosen was **PYPL**, which belongs to the business PayPal Holdings, Inc. The information was extracted by using the library *yfinance*, which offers a reliable, threaded, and pythonic way to download historical market data from Yahoo! Finance. For more information about this library, access the URL in TABLE II.

Library	URL	To install
yfinance	https://pypi.org/project/yfinance/	pip install yfinance

TABLE II. More information yfinance library.

B. *Dimenstion and types of variables*

To start gathering information, the Ticker module was created by indicating the name of the ticker of interest, in this case, **PYPL**. With this, I could access more information about the business, actions, dividends, splits, and historical market data. I extracted all the historical (max) market data of PayPal, getting the following information shown in TABLE III. This data has 1512 observations and 8 variables, from 2015-07-06 to 2021-04-20.

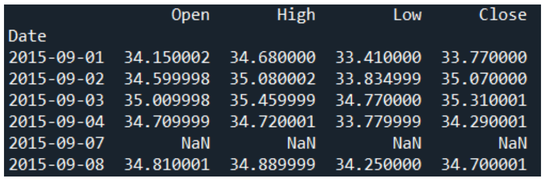
Historical Market Data		
Columns	**Type**	**Description**
Date	Date	Format: dd/mm/YY Frequency: Daily
Open	Decimal	Stock value when market open
High	Decimal	Highest value recorded in a day.
Low	Decimal	Lowest value recorded in a day.
Close	Decimal	Stock value when market close.
Volume	Integer	Number of assets has traded in one day.
Dividends	Boolean	When company gives dividends. This company does not give dividends.
Stock Splits	Boolean	

TABLE III. Description of Data.

III. Data Collection and Preparation

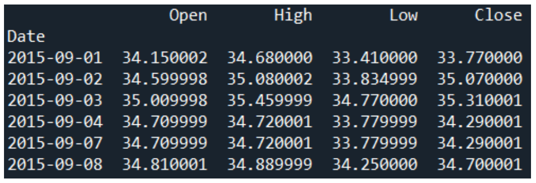
The process to collect and prepare the data was carried out in Python (Spyder IDE). The outcome of this process was stored locally on the database PostgreSQL. As a pre-requirement, the schema of the database had to be prepared beforehand.

In the data preparation stage, after setting the frequency of the data (business days), I found out missing values. After looking into it carefully, these were identified as American holidays. Roughly every year has 11 holidays, which makes sense after I got 53 missing dates (the data collected is made up of 7 years, however, 2015 and 2021 are not completed). To deal with these missing values, I just filled these with the previous value recorded. As an example, the 7th of September of 2015 was Labour Day, therefore the market was closed by that time. Figure 2 shows an example of a missing value, and in Figure 3, how it was filled up with method *ffill* from function *asfreq*.



```
                 Open       High        Low      Close
Date
2015-09-01  34.150002  34.680000  33.410000  33.770000
2015-09-02  34.599998  35.080002  33.834999  35.070000
2015-09-03  35.009998  35.459999  34.770000  35.310001
2015-09-04  34.709999  34.720001  33.779999  34.290001
2015-09-07        NaN        NaN        NaN        NaN
2015-09-08  34.810001  34.889999  34.250000  34.700001
```

Figure 2. Example of missing data due American holiday. In this case, it is Labour Day.



```
                 Open       High        Low      Close
Date
2015-09-01  34.150002  34.680000  33.410000  33.770000
2015-09-02  34.599998  35.080002  33.834999  35.070000
2015-09-03  35.009998  35.459999  34.770000  35.310001
2015-09-04  34.709999  34.720001  33.779999  34.290001
2015-09-07  34.709999  34.720001  33.779999  34.290001
2015-09-08  34.810001  34.889999  34.250000  34.700001
```

Figure 3. Example how missing value was dealt.

As PayPal does not give dividends and does not divide their existing shares into multiple shares, the variables *Dividends* and *Stock Splits* were removed (assuming the company does not have intentions of changing this in a near future). To find out more about the days when the market was closed due to American holidays, please visit the website https://bit.ly/3dzemob.

IV. Descriptive Statistics

The process to visualize and analyse the time series data has been carried out on Python with Jupyter Notebook IDE.

A. *Selection of variable of interest*

My variable of interest, in this case, is the Open price, as the best time to trade is in the first two hours after the market opens. It is important to mention that two hours before the market closes is a good time to trade as well. Figure 4 shows the values of the prices and the volume from 2015 to 2021.

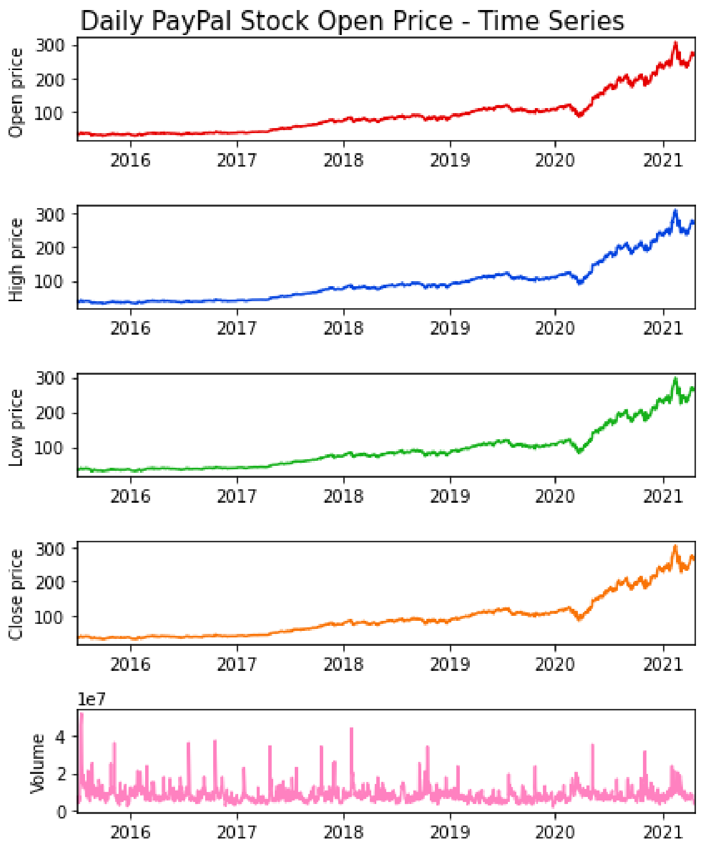
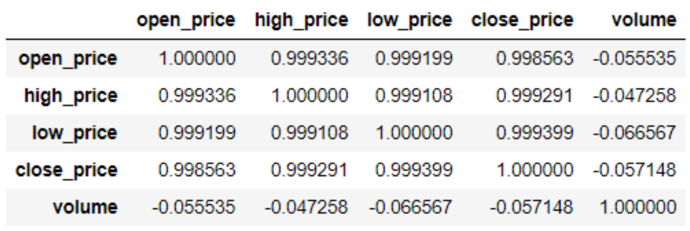


Figure 4. Open, High, Low and Close prices and Volume from 2015 to present.

As shown in Figure 5, the correlation (method: *spearman*) between the different prices is nearly one. This could lead to multicollinearity, therefore I will not to perform a multilinear regression model, but use an ARIMA model to forecast the open price.



	open_price	high_price	low_price	close_price	volume
open_price	1.000000	0.999336	0.999199	0.998563	-0.055535
high_price	0.999336	1.000000	0.999108	0.999291	-0.047258
low_price	0.999199	0.999108	1.000000	0.999399	-0.066567
close_price	0.998563	0.999291	0.999399	1.000000	-0.057148
volume	-0.055535	-0.047258	-0.066567	-0.057148	1.000000

Figure 5. Correlation matrix.

B. Exploratory analysis

I started the exploratory analysis by showing the open price of the PayPal Stock over time. Figure 6 shows how this has been changed from 2015 to 2021, with a frequency of business days (no weekends). In general, this shows an upper trend especially after 2020 due to the current pandemic (lockdowns come along with a growth in online sales). In figure 7 is shown an area plot to emphasize the growth of the trend.

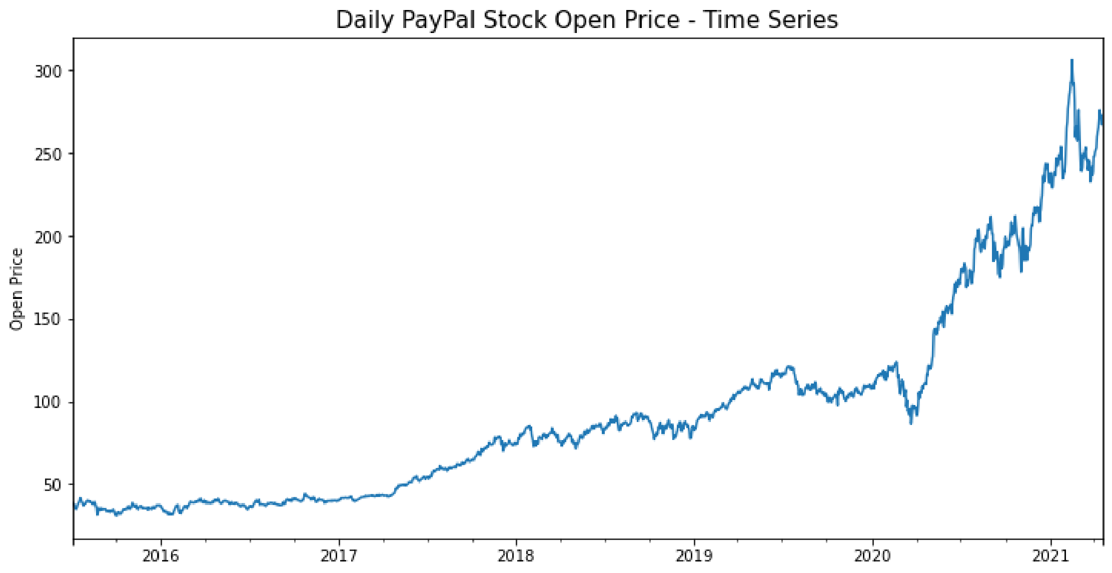


Figure 6. PayPal Open Price from 2015 to 2021.

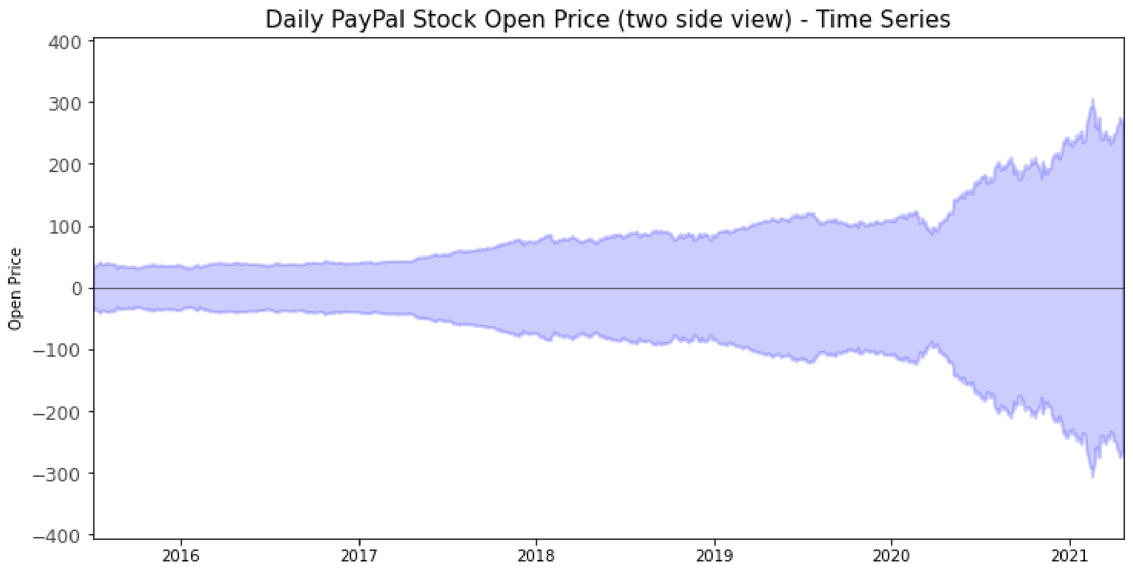


Figure 7. Two-side-view plot to emphasize the growth of the trend.

Figure 8 shows the same plot as Figure 6, but with more annotations and colour per year. The year 2020 is stand out in colour red and highlight the area when the market experienced a major crash in 2020 (from the 19th of February to the 23rd of March) due to the fear of the current pandemic.

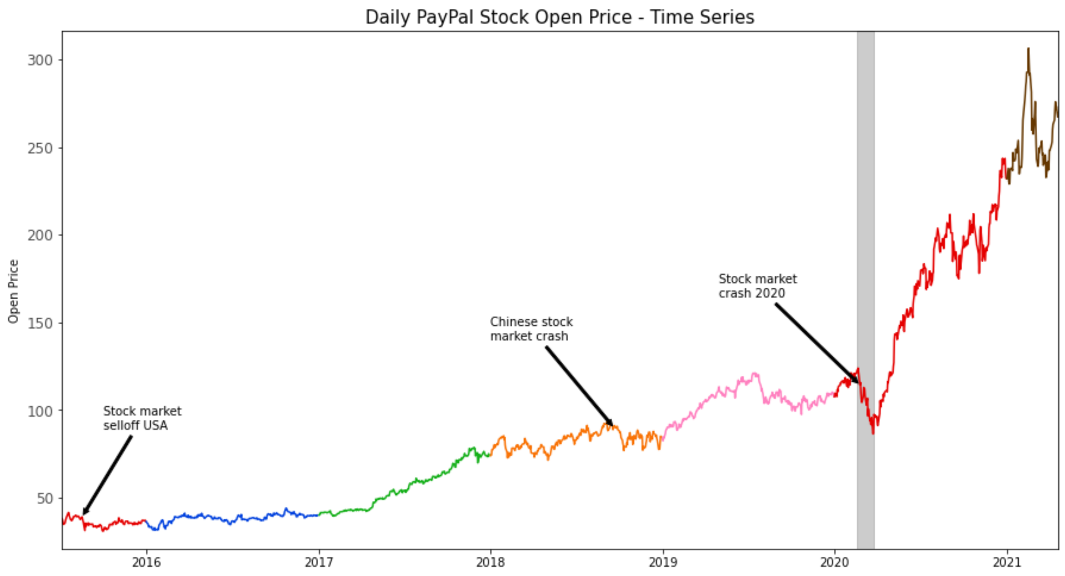


Figure 8. Open Price of PayPal Stock from 2015 to 2021 with annotations.

Figure 9 shows a line plot of the open price per year. The brown line is the year 2020 where is remarkable the effects of the pandemic after the market crash in February 2020. However, later the stock price starting growing significantly when comparing to previous years. In Figure 10 is shown a boxplot per year, where again the year 2020 stands out to have the largest variability.

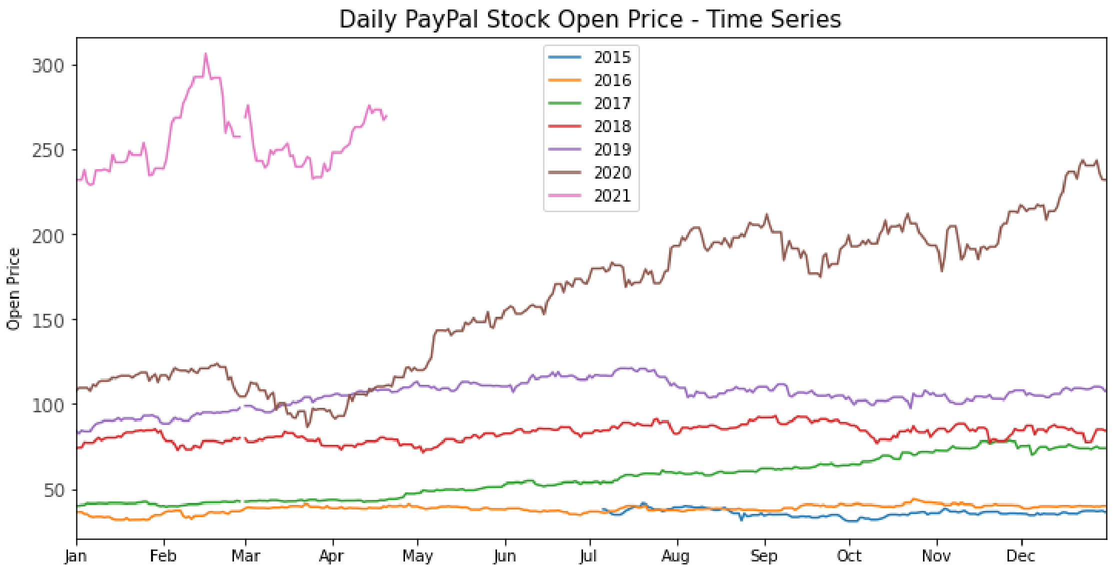


Figure 9. Annual stock price trend.

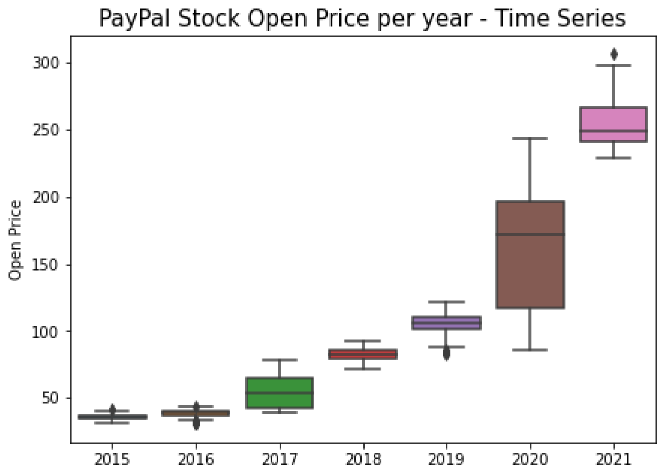


Figure 10. Annual distribution.

V. RESEARCH QUESTIONS

Based on the exploratory analysis, I am interested in answering the following questions:

- Is the open price of PayPal stock stationary or non-stationary? As a way to easily analyse time series, I need to check out if the data is stationary or not stationary. Looking to the decomposed plot would be enough to spot trend, seasonality, and if there is white noise. Therefore, sub-questions:
  - In general, is there a trend in the data?
    - Is it an uptrend?
    - Is it a downtrend?
  - Does it show seasonality?
    - Is it daily, weekly, yearly?
  - Does the variance is constant?
- Is it shown as an Autoregression (AR) or Moving Average (MA) behaviour? To identify this, after transforming the data to a stationary shape, I am going to plot ACF and PACF graphs to identify the behaviour of the data. Once that, I would be able to answer the following question:
  - How is the movement of the spikes in the ACF and PACF?
  - What order of AR or MA process do I need to use?
- *How is the trend of the PayPal stock for the following 6 months?*

The methodology to be implemented to look for the best model is called Box-Jenkins Method. This is compound of three steps:

1. Identify if time series data are **stationary or non-stationary**: before carrying out this method, I am going to split my data into train and test assets. Then, I will be looking at the decomposed plot to confirm if the data is stationary or non-stationary. After this, I will examine the ACF and PACF to spot AR or/and MA process on time series data.
2. Identify the **best model**. For this, I will use *auto_arima* with specific parameters. Then I will model the options observed on ACF and PACF plots, and then compare models to select the best.
3. **Diagnostics**. Inspect the diagnostic plot to check for assumptions: normal distribution, random noise, and not significant lags on residuals. After this, I intent

NOTE: this method is made up of three steps, however, this could lead to more sub-steps not included on the list above.

VI. Results

A. *Training and Test data*

Before carrying out the Box-Jenkins method, I divided my data into train and test assets. When taking 80% of data for training and 20% for the test process, I realized the test data contained dates after the stock market crash in 2020. I believe I could get a better prediction if I took into account dates after the crash. Therefore, I decided to take 90% of the data for training and 10% for the test process (see Figure 11 to see the distribution of the train and test assets after segmentation). Another reason for this is that Figure 10 shows how the stock price of PayPal has a larger variability in 2020 comparing to previous years.

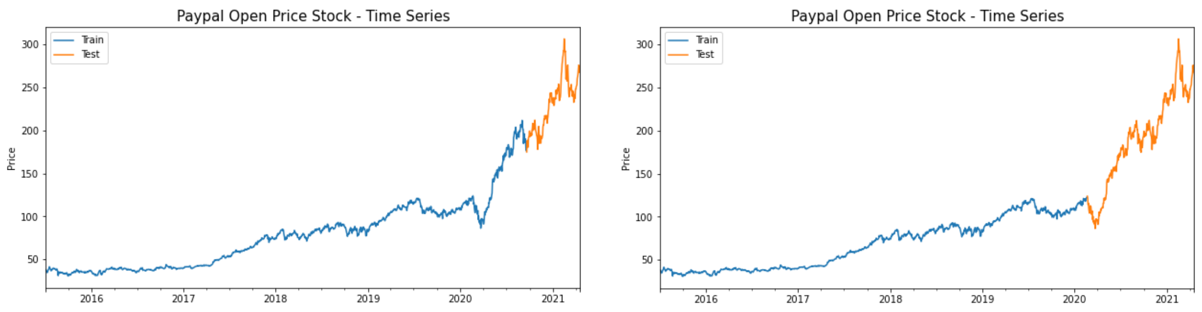


Figure 11. Decomposed plot.

B. *Decompose time series data*

By looking at Figure 6, it is evident that the data is non-stationary as it has an uptrend. However, to look in more detail, I used the decomposed plot to see the trend, seasonal, and residuals components (shown in Figure 12). In the seasonal section, there is a weird form that is hard to understand, hence I decided to plot the seasonal component in a separate image by selecting a smaller period. Now, the seasonal component shown in Figure 13 exhibits a clear weekly seasonality, in this case as the data has a frequency of business days, the week is made up of 5 days: Monday to Friday. Back to Figure 12, the residuals section on the decomposed plot does not show random noise as there is more variability after 2020 (once again) due to the pandemic. In other words, the data is stationary. Consequently, I applied a transformation to adjust for non-stationarity.

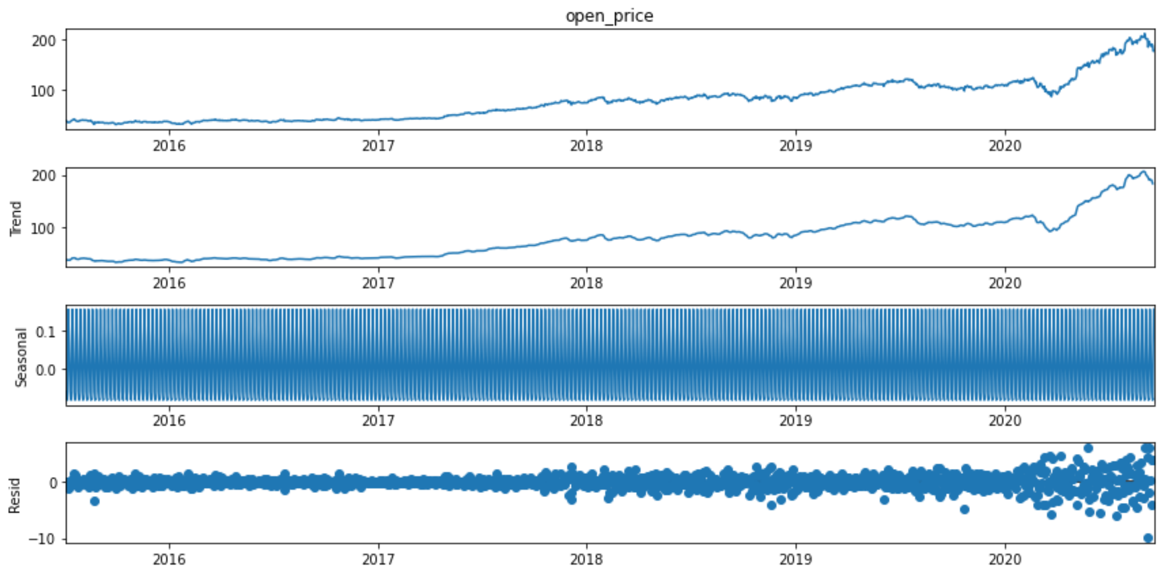


Figure 12. Decomposed plot.

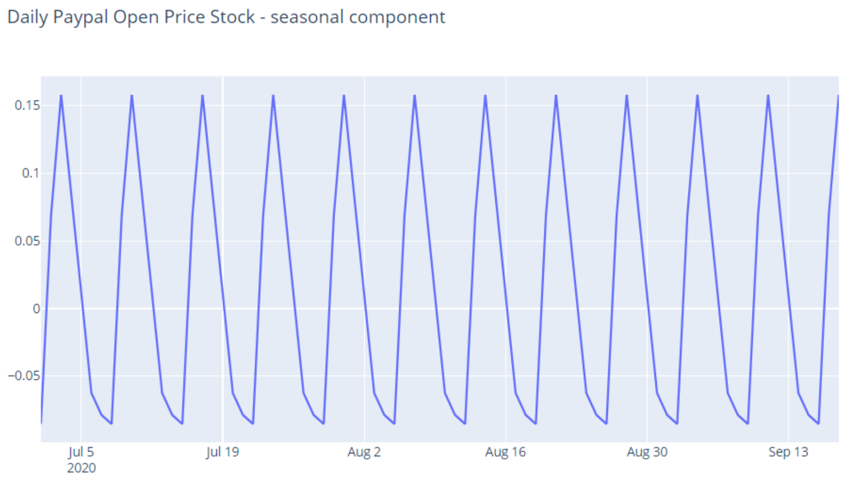
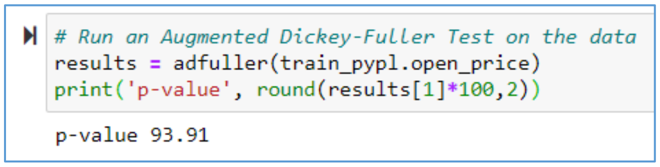


Figure 13. Zoom in on seasonal component. Now it is shown a clear seasonality within the weeks (every 5 days, as frequency of the time series data is business days).

Another way to verify if the data is stationary or not is by conducting a Dicker-Fuller test. The null and alternative hypothesis are:

H_0: Time series is non-stationary
H_1: Time series is stationary.

The code and outcome are shown in Figure 14. The p-value is 0.9391 which is greater than 0.05. Therefore, fail to reject the null hypothesis: The time series is non-stationary.



```
# Run an Augmented Dickey-Fuller Test on the data
results = adfuller(train_pypl.open_price)
print('p-value', round(results[1]*100,2))

p-value 93.91
```

Figure 14. Dicker-Fuller test before transformation process.

C. Transform data to adjust for Non-stationarity

To transform the data into a stationary shape, first I applied the following:

1. Log to deal with the variance.
2. Differencing over 5 periods to deal with seasonality.
3. One differencing.

The data looks more stationary in Figure 15: no trend, no seasonality and looks more like a random walk. The red line that crosses in the middle of the data is the mean value, which is 0.001130.

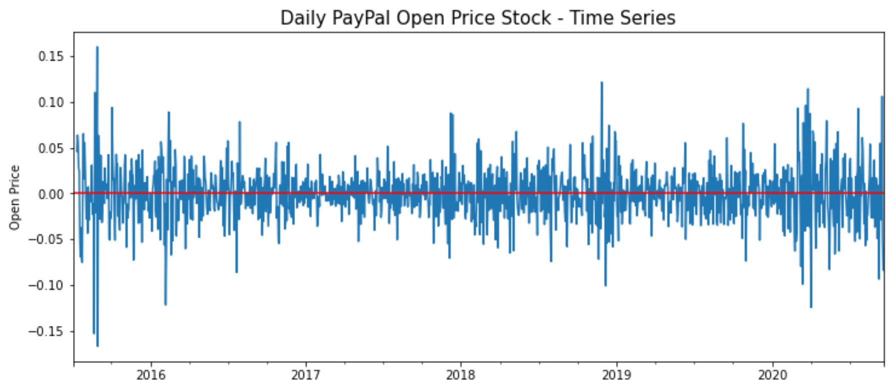
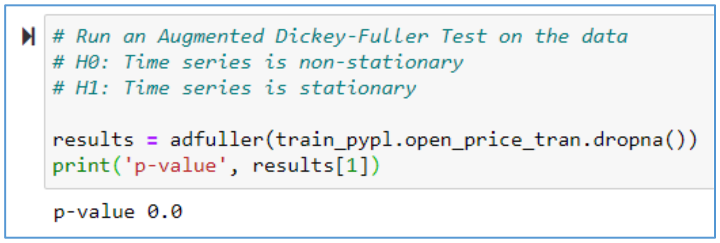


Figure 15. Data after transformation.

Finally, once again I performed the Dicker-Fuller test to confirm whether the time series data is stationary. This time the p-value is less than 0.05 (see Figure 16). Therefore, the null hypothesis is rejected: Time series object is stationary.



```
# Run an Augmented Dickey-Fuller Test on the data
# H0: Time series is non-stationary
# H1: Time series is stationary

results = adfuller(train_pypl.open_price_tran.dropna())
print('p-value', results[1])

p-value 0.0
```

Figure 16. Dicker-Fuller test after transformation process.

D. *Diagnosing the ACF and PACF Plots*

Once the data was stationary, the next step is plotting the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) to identify whether there is an AR and/or MA process. The plots are shown in Figure 17. My interpretation: the ACF shows a sharp cut off that means there is a MA process with order 1.

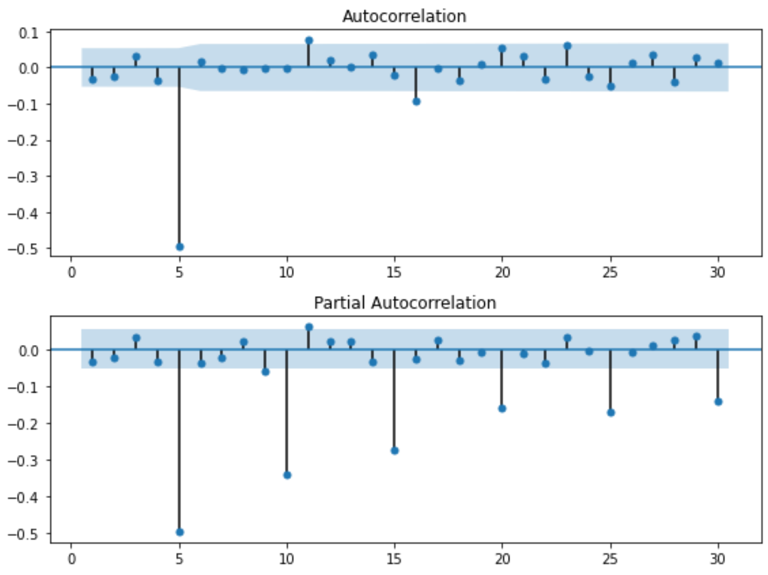


Figure 17. ACF and PACF plots on transformed data.

The possible options spotted on the ACF plot for the modelling process are:

- ARIMA(0,1,0)(0,1,1)[5]
- ARIMA(0,0,0)(0,1,1)[5]

However, when I performed the *auto_arima* function (step E, please look at the results in Figures 20 and 21) I got the model ARIMA(1,1,0) with no seasonality. This grabbed my attention and decided to plot the ACF and PACF with data transformed as follows:

1. Apply log to deal with the variance.
2. Apply one differencing.
3. No seasonality differencing applied

Then, I plotted it and it looked stationary (see Figure 18). From the Dickey-Fuller Test (see Figure 19), the p-value is < 0.05, therefore the null hypothesis is rejected: Time series is stationary.

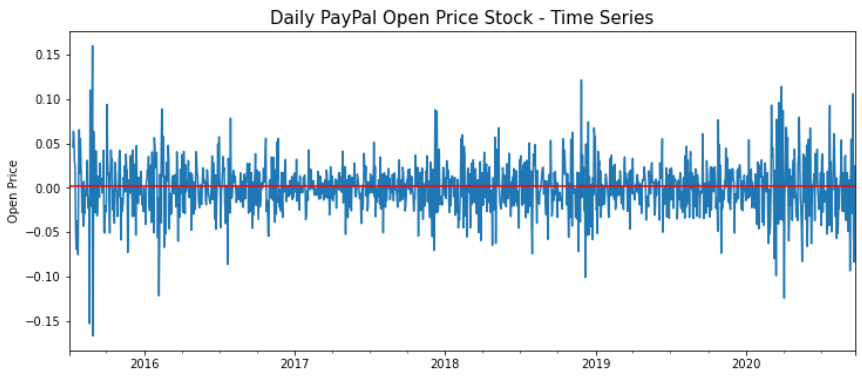
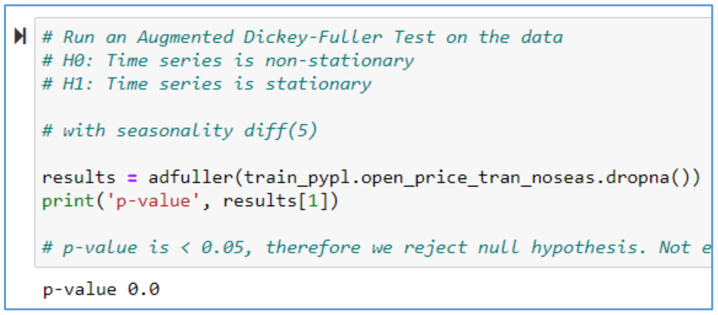


Figure 18. Open Price Stock after transformation.



```
# Run an Augmented Dickey-Fuller Test on the data
# H0: Time series is non-stationary
# H1: Time series is stationary

# with seasonality diff(5)

results = adfuller(train_pypl.open_price_tran_noseas.dropna())
print('p-value', results[1])

# p-value is < 0.05, therefore we reject null hypothesis. Not e
```

```
p-value 0.0
```

Figure 19. Dicker-Fuller test after transformation process.

Then, when plotting the ACF and PACF with the transformed data (no seasonal differencing), the plots have a mixed behaviour of AR or MA, which is shown in figure 20 (this is the reason the auto_arima function suggest AR(1) and one differencing).

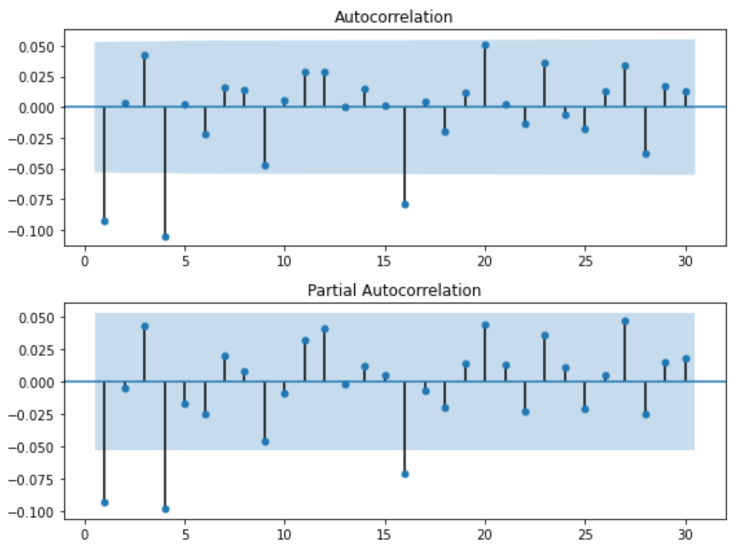


Figure 20. ACF and PACF plots on transformed data.

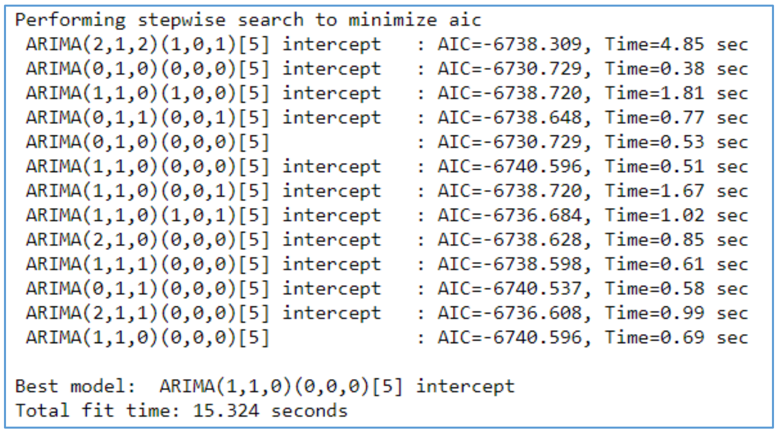
The final list of suggested models is:

- ARIMA(0,1,1)
- ARIMA(1,1,1)
- ARIMA(0,0,0)(0,1,1)[5]
- ARIMA(0,1,0)(0,1,1)[5]

NOTE: The ARIMA(1,1,0) model is not included in the list as this was spotted from the auto_arima function.

E. Modelling

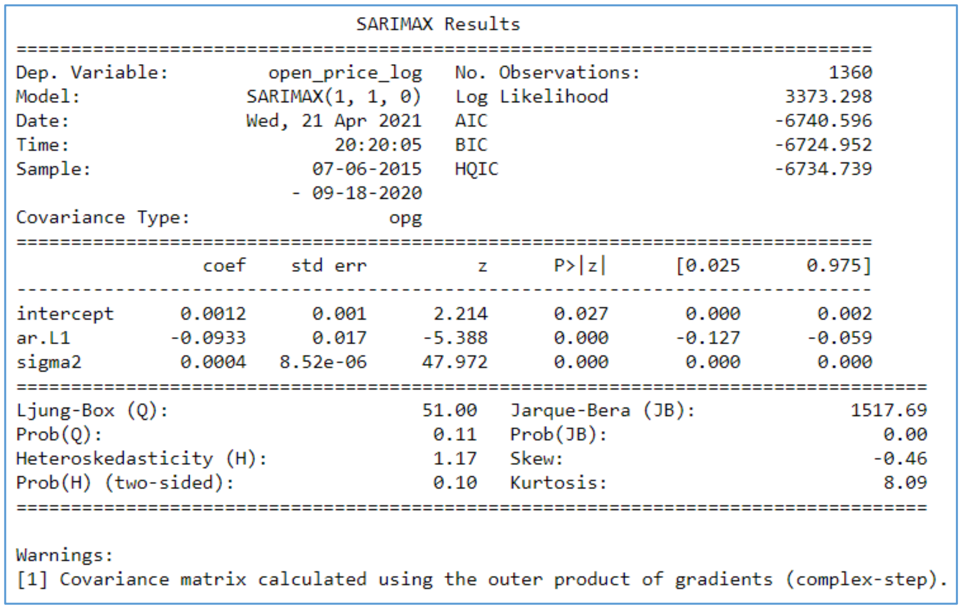
Before selecting the best model from the options in step D, I performed the *auto_arima* function to get the model with the lowest AIC values. It is important to point out the variable of interest was transformed beforehand as the variance was not constant (leading to an assumption violation when diagnostic residuals from the model). Therefore, from this point, the modelling process used log transformation on *open_price* stock. The results from the auto_arima function are shown in Figure 21 where the best model defined by the process was ARIMA(1,1,0)(0,0,0)[5] with AIC -6734.727 and BIC -6719.085. Figure 22 is shown a summary of the results.



```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(1,0,1)[5] intercept   : AIC=-6738.309, Time=4.85 sec
 ARIMA(0,1,0)(0,0,0)[5] intercept   : AIC=-6730.729, Time=0.38 sec
 ARIMA(1,1,0)(1,0,0)[5] intercept   : AIC=-6738.720, Time=1.81 sec
 ARIMA(0,1,1)(0,0,1)[5] intercept   : AIC=-6738.648, Time=0.77 sec
 ARIMA(0,1,0)(0,0,0)[5]             : AIC=-6730.729, Time=0.53 sec
 ARIMA(1,1,0)(0,0,0)[5] intercept   : AIC=-6740.596, Time=0.51 sec
 ARIMA(1,1,0)(0,0,1)[5] intercept   : AIC=-6738.720, Time=1.67 sec
 ARIMA(1,1,0)(1,0,1)[5] intercept   : AIC=-6736.684, Time=1.02 sec
 ARIMA(2,1,0)(0,0,0)[5] intercept   : AIC=-6738.628, Time=0.85 sec
 ARIMA(1,1,1)(0,0,0)[5] intercept   : AIC=-6738.598, Time=0.61 sec
 ARIMA(0,1,1)(0,0,0)[5] intercept   : AIC=-6740.537, Time=0.58 sec
 ARIMA(2,1,1)(0,0,0)[5] intercept   : AIC=-6736.608, Time=0.99 sec
 ARIMA(1,1,0)(0,0,0)[5]             : AIC=-6740.596, Time=0.69 sec

Best model:  ARIMA(1,1,0)(0,0,0)[5] intercept
Total fit time: 15.324 seconds
```

Figure 21. Outcomes from *auto_arima* function.



```
                               SARIMAX Results
==============================================================================
Dep. Variable:          open_price_log   No. Observations:                 1360
Model:                SARIMAX(1, 1, 0)   Log Likelihood                3373.298
Date:                 Wed, 21 Apr 2021   AIC                          -6740.596
Time:                         20:20:05   BIC                          -6724.952
Sample:                     07-06-2015   HQIC                         -6734.739
                          - 09-18-2020
Covariance Type:                   opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      0.0012      0.001      2.214      0.027       0.000       0.002
ar.L1         -0.0933      0.017     -5.388      0.000      -0.127      -0.059
sigma2         0.0004   8.52e-06     47.972      0.000       0.000       0.000
===================================================================================
Ljung-Box (Q):                       51.00   Jarque-Bera (JB):              1517.69
Prob(Q):                              0.11   Prob(JB):                         0.00
Heteroskedasticity (H):               1.17   Skew:                            -0.46
Prob(H) (two-sided):                  0.10   Kurtosis:                         8.09
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

Figure 22. Summary results from ARIMA(1,1,0)(0,0,0)_5.

The p-values on the coefficients (excluding intercept) are significant as all of them have p-value < 0. The equations given is the following:

$$y_t = 0.0012 - 0.0933y_{t-1} + \varepsilon_t$$

where ε_t is white noise with a variance of 0.0004 and standard deviation 0.02.

Then, I interpreted the Ljung-Box Test for an autocorrelation pattern in the residuals. The hypotheses are:

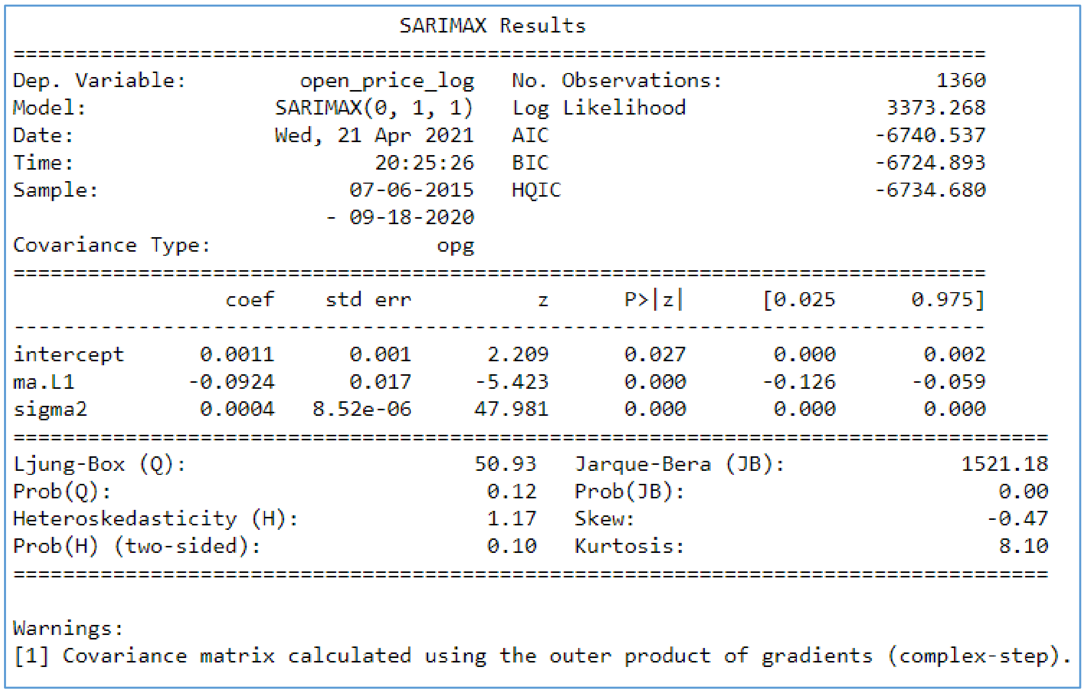
H_0: The data are independently distributed.
H_1: The data are not independently distributed.

In this case, the p-value shown in the summary of the results (Figure 20) is 0.11 > 0.05, therefore the Null Hypothesis is not rejected: The data are independently distributed.

TABLE IV shows a summary of the results from the models suggested. The best model is ARIMA(0,1,1) as it has the lowest AIC and BIC values. Therefore, Figure 23 is shown the whole summary result.

Summary of results			
Model	**AIC**	**BIC**	**Notes**
ARIMA(0,1,1)	-6740.537	-6724.893	
ARIMA(1,1,1)	-6738.598	-6717.740	Coefficient are not significant as p-value is greater than 0.05.
ARIMA(0,1,0)(0,1,1)[5]	-6677.522	-6661.890	
ARIMA(0,0,0)(0,1,1)[5]	-4779.107	-4763.472	

TABLE IV. Summary of results from suggested models.


```
                               SARIMAX Results
==============================================================================
Dep. Variable:         open_price_log   No. Observations:                 1360
Model:               SARIMAX(0, 1, 1)   Log Likelihood                3373.268
Date:                Wed, 21 Apr 2021   AIC                          -6740.537
Time:                        20:25:26   BIC                          -6724.893
Sample:                    07-06-2015   HQIC                         -6734.680
                         - 09-18-2020
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      0.0011      0.001      2.209      0.027       0.000       0.002
ma.L1         -0.0924      0.017     -5.423      0.000      -0.126      -0.059
sigma2         0.0004   8.52e-06     47.981      0.000       0.000       0.000
===================================================================================
Ljung-Box (Q):                       50.93   Jarque-Bera (JB):              1521.18
Prob(Q):                              0.12   Prob(JB):                         0.00
Heteroskedasticity (H):               1.17   Skew:                            -0.47
Prob(H) (two-sided):                  0.10   Kurtosis:                         8.10
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

Figure 23. Summary results from ARIMA(0,1,1).

From the model ARIMA(0,1,1), the p-values of the coefficients (excluding intercept) are significant as all of them have p-value < 0. The equations given are the following:

$$y_t = 0.0011 - 0.0924\varepsilon_{t-1} + \varepsilon_t$$

where ε_t is white noise with a variance of 0.0004 and standard deviation 0.02.

Looking at the Ljung-Box test, the p-value is 0.12 that is greater than 0.05, therefore the Null Hypothesis is not rejected: The data are independently distributed.

So far, there are two good models: ARIMA(1,1,0) generated by *auto_arima* function, and ARIMA(1,1,0) spotted from the ACF and PACF plots. In the following step, I checked residuals by plotting the diagnostic plot.

F. *Diagnostic*

To evaluate the accuracy of the models, I looked into the diagnostic plot (Figures 24 and 25). This had to be done to make sure the following assumptions:

- **Random variance**. The plot on the top-right side from both diagnostic plots shown random variance. This assumption is approved for both models.
- **Normal distribution**. From both diagnostic plots, the histogram and Q-Q plot shown the residuals have a normal distribution. This assumption is approved for both models.
- **No significant lags in correlogram** (the most critical one). (the most critical one). From both diagnostic plots, the correlogram showed a significant lag at 4, however, it looks random. Therefore, this assumption is approved for both models.

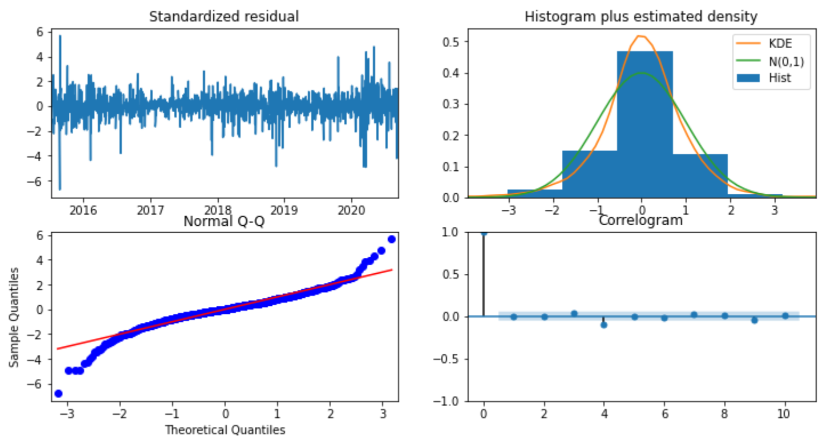


Figure 24. Diagnostic plot for ARIMA(1,1,0).

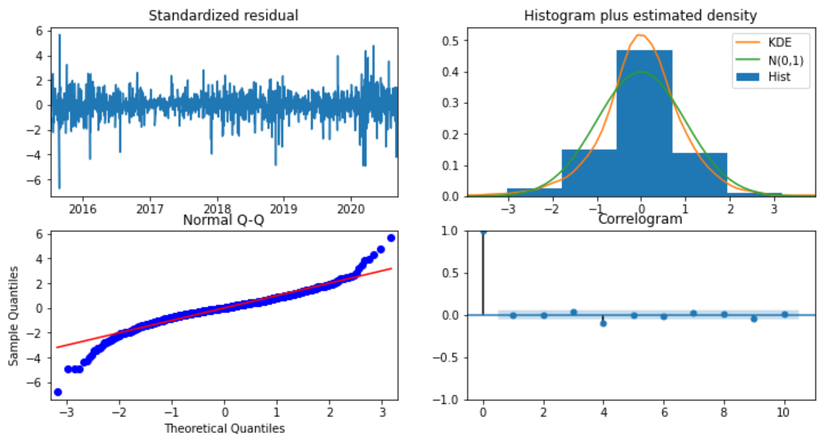


Figure 25. Diagnostic plot for ARIMA(0,1,1).

Finally, to select the best model, as interested in the accuracy of the forecasts from the model, I calculated the RMSE, MAE, and MAPE for the forecasts from both of the models on the test data. The results are shown in TABLE V. It seems both models have similar RMSE, MAE, and MAPE values (delta is very small). Therefore, the selection of the model was based on the lowest AIC and BIC values. The **ARIMA(1,1,0)** model shown the lowest values; therefore, this model was selected.

Accuracy of models when forecasting					
Model	**RMSE**	**MAE**	**MAPE**	**ACI**	**BIC**
ARIMA(1,1,0)	0.1956	0.1724	3.1347	-6740.596	-6724.952
ARIMA(0,1,1)	0.1954	0.1721	3.1303	-6740.537	-6724.893

TABLE V. Outcomes to check accuracy.

G. *Forecasting*

The first thing to do before forecasting is looking the difference between the fitted values and the real values from the train data. In Figure 26 it is shown the fitted values and the real values in the train time series data. It seems the model has fitted the values appropriately.

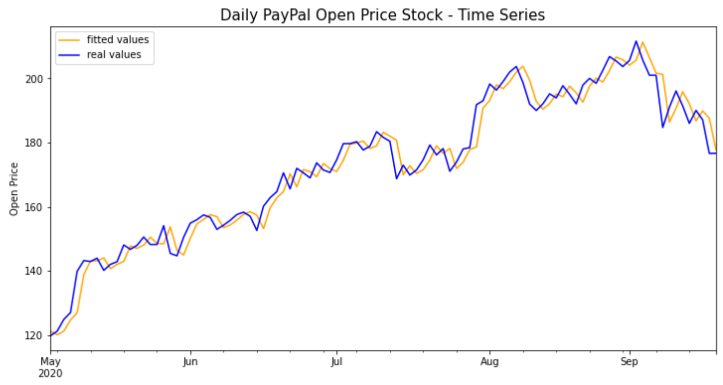


Figure 26. Predictions from 2020-05 to max date on training data.

Now, it is time to forecast the future values from the following months on the training dataset and compare the test data. As shown in Figure 27, the forecasted trend showed an increase in the open price stock of PayPal, which is accurate to the test data. Additionally, it does not forecast the remarkable increase but predicted really good the upper trend.

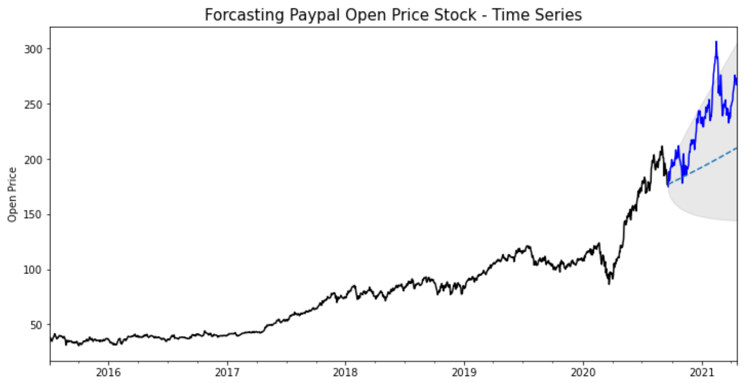


Figure 27. Predictions from 2020-05 to max date on training data.

Lastly, I was interested on looking into the trend of PayPal price stock for the following six months. As the frequency of the time series is business days (5 days per week, as market is closed on weekends), I used a period of 120 days equal to one month. In Figure 28 it is shown an upper trend for the following month.

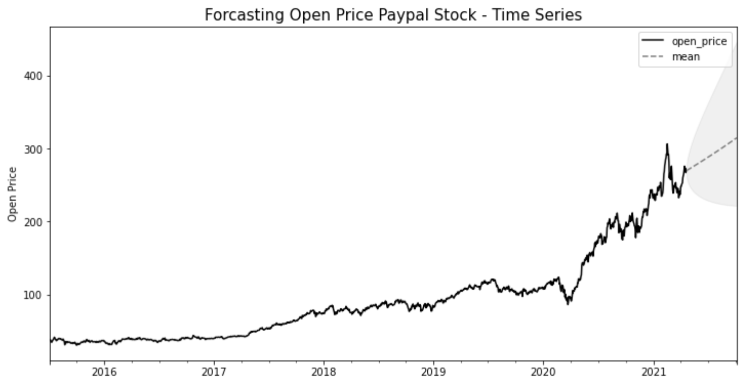


Figure 28. Forecasting for the following 6 months.

VII. Conclusions

Time series analysis is an important technic for forecasting, and being used in many areas such as finance. From the exploratory data, it was interesting to look at how the stock price has an important growth after the crash market in 2020. As a result, when looking at the residuals in the decomposed plot, the variance was not random. Additionally, due to the amount of data the seasonal component had a weird shape (like a blue bar), however, when zooming up into a short period it was clear there was a weekly seasonality behaviour.

An interesting finding on the ACF and PACF plots (with no seasonal differencing) is that there were two possible models: ARIMA(1,1,0) and ARIMA(0,1,1). When comparing these (AIC, BIC, and errors when forecasting) there was a small difference, showing the ARIMA(1,1,0) to be the better. The ARIMA(1,1,0) forecasted an upper trend for the following 6 months, suggesting a good option to invest in PayPal stock. There are different technics to spot when to buy stocks, such as looking at the combination of moving average of 20, 50, and 200 days, however, this topic is out of the scope of this project.

What should have I done differently? I should have plotted the ACF and PACF without seasonality differencing as the auto_arima function suggested as the best model ARIMA(1,1,0). I had to go back to transform the data with no seasonality (i.e., log and one differencing) step and carry out again all the analysis on ACF and PACF.

For future work, I want to look into how to apply other technics such as ICA to predict market stock prices.