Stock Price Prediction

Using Polynomial Regression

By: Divyansh Tiwari

Index

Introduction

A correct prediction of stocks can lead to huge profits for the seller and the broker. Frequently, it is brought out that prediction is chaotic rather than random, which means it can be predicted by carefully analyzing the history of respective stock market.

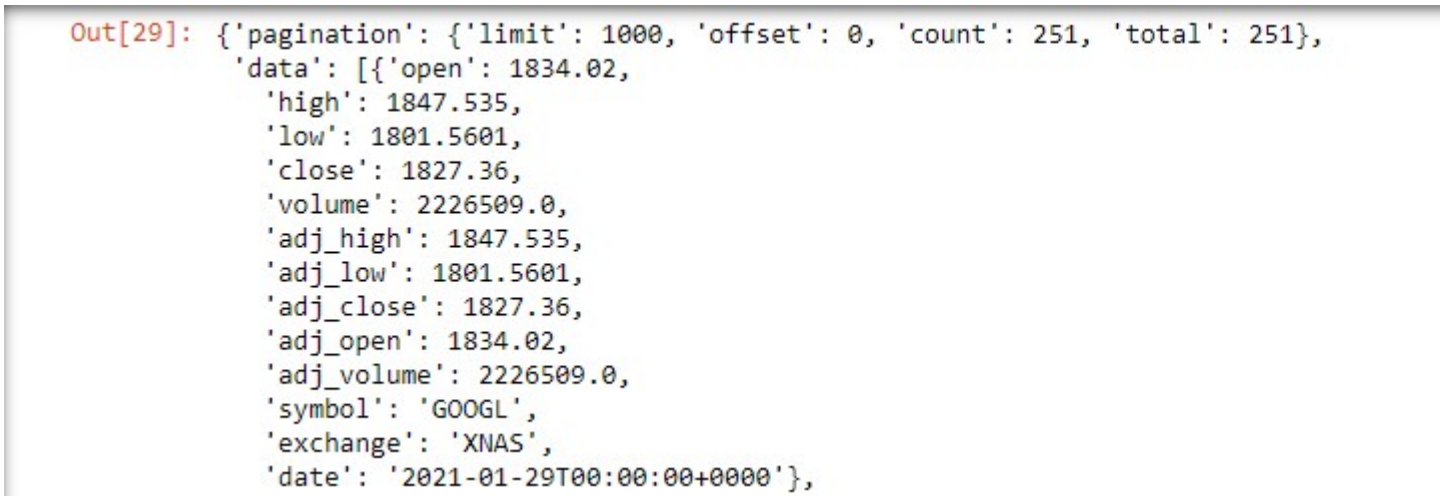
Machine learning is an efficient way to represent such processes. It predicts a market value close to the tangible value, thereby increasing the accuracy. Introduction of machine learning to the area of stock prediction has appealed to many researches because of its efficient and accurate measurements.

Data Analysis

Data Gathering

The vital part of machine learning is the dataset used. The dataset should be as concrete as possible because a little change in the data can perpetuate massive changes in the outcome. In this project, supervised machine learning is employed on a dataset obtained from marketstack.com.

An API call was made to marketstack.com and data of symbol = GOOGL from date 01-01-2018 to date 31-01-2021 was requested. Only 251 entries were available in the returned database that was from 31-01-2020 to 31-01-2020. This dataset comprises of following thirteen variables: open, close, low, high, symbol, date, adj_high,sdj_low, adj_open, adj_close, exchange and volume.



```
Out[29]: {'pagination': {'limit': 1000, 'offset': 0, 'count': 251, 'total': 251},
          'data': [{'open': 1834.02,
            'high': 1847.535,
            'low': 1801.5601,
            'close': 1827.36,
            'volume': 2226509.0,
            'adj_high': 1847.535,
            'adj_low': 1801.5601,
            'adj_close': 1827.36,
            'adj_open': 1834.02,
            'adj_volume': 2226509.0,
            'symbol': 'GOOGL',
            'exchange': 'XNAS',
            'date': '2021-01-29T00:00:00+0000'},
```

Data Preprocessing

Using the modules of python, only valid and usable data was selected from the JSON file and data-frame was created using 'Pandas'.

Out[30]:

	open	high	low	close	volume	adj_high	adj_low	adj_close	adj_open	adj_volume	symbol	exchange	date
0	1834.02	1847.535	1801.5601	1827.36	2226509.0	1847.535	1801.5601	1827.36	1834.02	2226509.0	GOOGL	XNAS	2021-01-29T00:00:00+0000
1	1831.00	1887.990	1831.0000	1853.20	2763905.0	1887.990	1831.0000	1853.20	1831.00	2763905.0	GOOGL	XNAS	2021-01-28T00:00:00+0000
2	1874.91	1880.470	1797.2800	1818.94	4125631.0	1880.470	1797.2800	1818.94	1874.91	4125631.0	GOOGL	XNAS	2021-01-27T00:00:00+0000
3	1885.99	1915.750	1876.1300	1907.95	1573078.0	1915.750	1876.1300	1907.95	1885.99	1573078.0	GOOGL	XNAS	2021-01-26T00:00:00+0000
4	1912.74	1921.820	1859.1600	1894.28	2529346.0	1921.820	1859.1600	1894.28	1912.74	2529346.0	GOOGL	XNAS	2021-01-25T00:00:00+0000
...	...	...	...	...	...	...	...	...	...	...	...	...	...
246	1467.38	1485.240	1465.4100	1479.11	1418049.0	1485.240	1465.4100	1479.11	1467.38	1418049.0	GOOGL	XNAS	2020-02-07T00:00:00+0000
247	1451.98	1481.560	1450.4800	1475.97	1891079.0	1481.560	1450.4800	1475.97	1451.98	1891079.0	GOOGL	XNAS	2020-02-06T00:00:00+0000
248	1463.61	1464.580	1429.6700	1446.05	1818793.0	1464.580	1429.6700	1446.05	1463.61	1818793.0	GOOGL	XNAS	2020-02-05T00:00:00+0000
249	1454.49	1467.340	1422.0300	1445.41	4793967.0	1467.340	1422.0300	1445.41	1454.49	4793967.0	GOOGL	XNAS	2020-02-04T00:00:00+0000
250	1461.65	1486.300	1456.6100	1482.60	3608760.0	1486.300	1456.6100	1482.60	1461.65	3608760.0	GOOGL	XNAS	2020-02-03T00:00:00+0000

251 rows × 13 columns

Data Filtering

Next Data was filtered and all the un-useful columns were removed. Only the columns open close and volume were kept. And DataFrame was inverted to keep it in the increasing order of dates. Then a column 'day' was added to keep the day count.

Out[79]:

	day	open	close	volume
0	0	1461.65	1482.60	3608760.0
1	1	1454.49	1445.41	4793967.0
2	2	1463.61	1446.05	1818793.0
3	3	1451.98	1475.97	1891079.0
4	4	1467.38	1479.11	1418049.0

Data Cleaning

To fit the model, one needs to get rid of the null values. Hence, first we need to identify the columns with null, none or NaN values. A count of the null values from each column was taken.

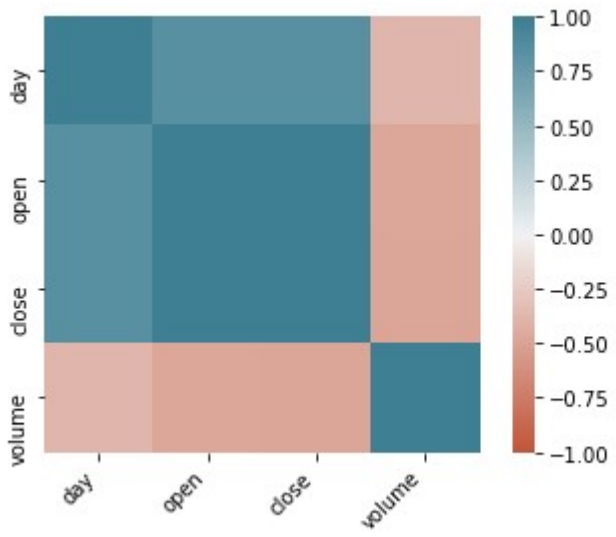
Out[32]:
```
open      0
close     0
volume    0
dtype: int64
```

We don't have any null values in our Dataset. Hence we proceed.

Data Visualization:

Heat Map: Correlation:

Correlation helps to find how strong the dependency of 2 variables is. And hence we plot a heatmap for observing the correlation between the columns



Green means positive, red means negative. The stronger the color, the larger the correlation magnitude. The heatmap allows us to observe which columns have a strong relation between them. Hence, we observe a good relationship between Day and Closing price.

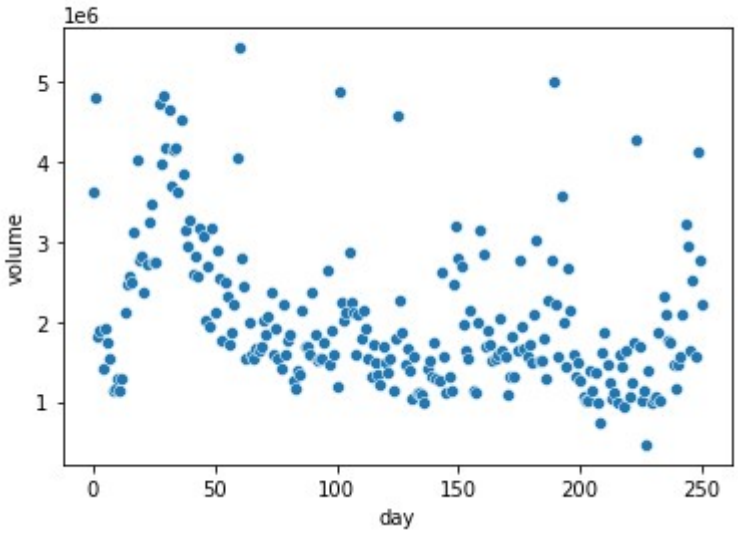
Next we plot some graphs to observe the relationship between different columns:

Day vs Closing-

Scatter Plots:

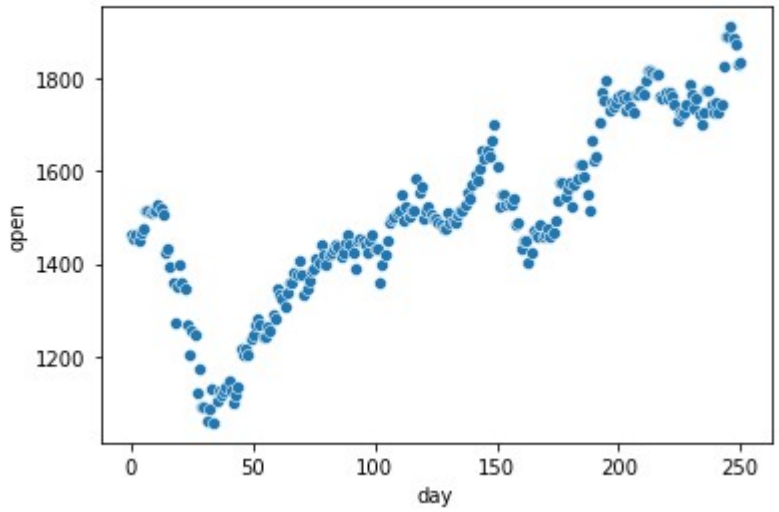
1. Day vs Volume

```
Out[92]: <AxesSubplot:xlabel='day', ylabel='volume'>
```



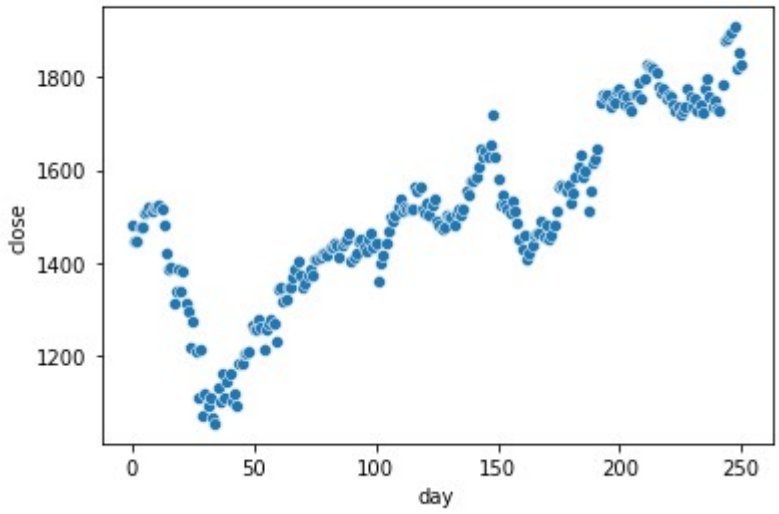
2. Day vs Opening

```
Out[94]: <AxesSubplot:xlabel='day', ylabel='open'>
```



3. Day vs Closing

```
Out[93]: <AxesSubplot:xlabel='day', ylabel='close'>
```



Model Development - Polynomial Regression

We used Polynomial regression – to fit a polynomial on our dataset which can help us predict further values by putting values in our polynomial. We split the training and testing dataset in 80-20 ratio. And then fit a 4 degree polynomial on our dataset.

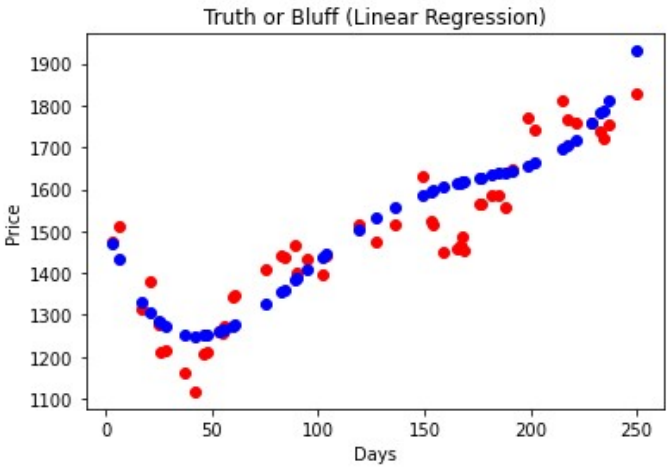
Observation and Conclusion

Results:

```
Mean absolute error = 63.45
Mean squared error = 5874.24
Median absolute error = 64.34
Explain variance score = 0.84
R2 score = 0.83
```

Best possible score is 1.0 and it can be negative (because the model can be arbitrarily worse). A constant model that always predicts the expected value of y, disregarding the input features, would get a R^2 score of 0.0. We get an R2_score of 0.83 which it pretty good.

Plotting Real vs Predicted Values we get:



Here Red are the real values, and blue being the predicted values.

Link: https://nbviewer.jupyter.org/github/DivyT-03/StockPricePred/blob/main/Stock%20Price%20Prediction.ipynb