Predictive Analysis of Stock Returns

Prutha Parmar prp2126, Prajwal Prakash pp2719, Heetika Gada hg2532
Columbia University

Abstract

Stock market is the primary indicator of a country's economic strength and development. The prices of stock market depends on multiple factors. The fluctuations in the stock prices is one of the main reasons for investors to believe that there is a need to predict stock prices or at least get an insight into stock prices in order to make an informed decision. Various kinds of classifiers and machine learning techniques are used for the same. In our project, we will predict and analyze the future stock prices using two kinds of data: one using historical prices and another using real time data. We perform exploratory data analysis, data cleansing, feature engineering and fit various models to get the best prediction accuracy.

Keywords:stock market; machine learning; historical prices; real time data.

1. Introduction

In today's world, predicting stocks has become very important since it can yield significant profits to a company or an investor. There are various factors or features that help predict stocks, but for our problem definition we are considering the historical data set for Part 1 and real time data for Part 2. Additionally, one of the major technical challenges while dealing with such data sets is to get prediction results without the interference of noise. Two major questions that might come to your mind right now is why historical data and real time data analysis? Historical stock are basically a company's stock at a specific point in history. Let's say a company is trading for many years and has a lengthy history of stock prices, which reflects the economic and company changes over time. However, in the case of newer companies it shows what they are priced right now compared to their worth in the first year of trading. Therefore, historical data helps better judge the value of an investment. Furthermore, the better educated you are in historical stock prices, the more you tend to understand the entire stock market investment process. Real time data gives us the updated price or volume information on a tick by tick basis for a stock that trades on the exchange. This is important as it shows how the value of a stock fluctuates. It helps investors/traders who buy and sell securities during the day and have to make quick decisions on price moves. Moreover, real time charts gives updated information on reversion points which essentially tells the trader when the stock is starting to lag or when it is about to break. Hence, we decided to go for historical and real time datasets.

For historical data set, we did pre-processing, applied various models such as Sequential Neural Network, LSTM and LSTM on relative returns and concluded that LSTM on relative returns gave the best prediction accuracy of 94.23%.

For real time data set, we did exploratory data analysis, feature engineering, data preprocessing, cross validation and applied various models like linear regression, Quadratic Discriminant Analysis and KNN regression. We concluded that QDA with line of best fit two gave us the best result i.e. accuracy of 94.48%.

2. Related Works

A lot of research work is going on in this domain of predicting stock prices using historical and real time data. We did go through a few papers. “Using AI to make prediction on Stock Market”[6] does similar kind of exploratory data analysis and they also tend to use linear regression for predicting the real time data. The data set used by them is also provided by Alpha Vantage. Although their approach was quite different than ours. They included technical indicators for predicting accuracy while we don’t.

3. System overview

In this section, we describe the two data sets used in project.

3.1. Data set 1

For our project, we are using two datasets to predict and analyze the stock prices.

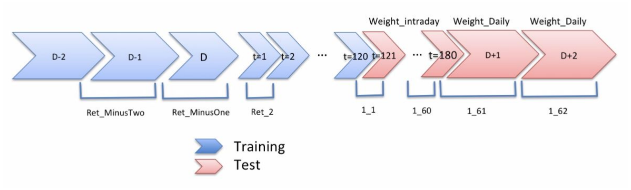
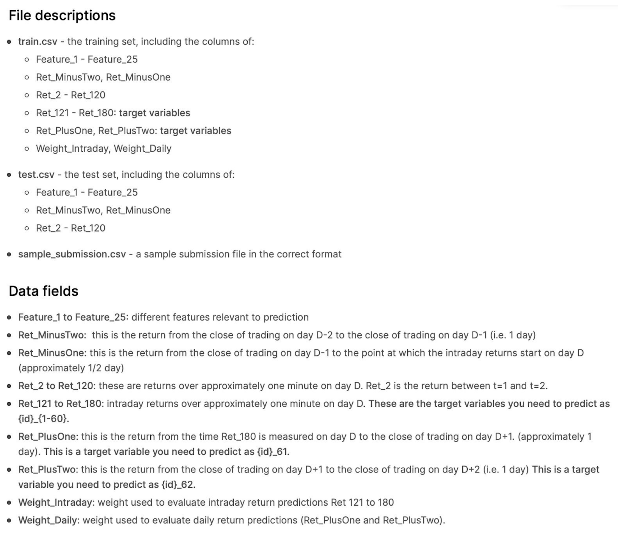


Figure 1: Historical data set for a 5-day window by Winton

In Part 1 of our project, we are predicting the stock prices using the historical data provided by the company Winton as part of “The Winton Stock Market Challenge” on Kaggle. The dataset is designed such that it is a representative of real data and thus should pose a number of challenges such as noise. The data set provided the stock prices for past two days(D-2, D-1) and part of the day D and based on that we were to predict for the rest of the day D, D+1 and D+2. During the day D, there is intraday return data, which are basically the returns at different points in the day. They provided 180 minutes of data, from t=1 to t=180. For the training set they provided with full 180 minutes while in the test set just the first 120 minutes were given. For each 5-day window, 25 features were given, which may or may not be useful for prediction. Each row in the dataset is an arbitrary stock at an arbitrary 5 day time window.



File descriptions

- **train.csv** - the training set, including the columns of:
  - Feature_1 - Feature_25
  - Ret_MinusTwo, Ret_MinusOne
  - Ret_2 - Ret_120
  - Ret_121 - Ret_180: **target variables**
  - Ret_PlusOne, Ret_PlusTwo: **target variables**
  - Weight_Intraday, Weight_Daily
- **test.csv** - the test set, including the columns of:
  - Feature_1 - Feature_25
  - Ret_MinusTwo, Ret_MinusOne
  - Ret_2 - Ret_120
- **sample_submission.csv** - a sample submission file in the correct format

Data fields

- **Feature_1 to Feature_25**: different features relevant to prediction
- **Ret_MinusTwo**: this is the return from the close of trading on day D-2 to the close of trading on day D-1 (i.e. 1 day)
- **Ret_MinusOne**: this is the return from the close of trading on day D-1 to the point at which the intraday returns start on day D (approximately 1/2 day)
- **Ret_2 to Ret_120**: these are returns over approximately one minute on day D. Ret_2 is the return between t=1 and t=2.
- **Ret_121 to Ret_180**: intraday returns over approximately one minute on day D. **These are the target variables you need to predict as {id}_{1-60}.**
- **Ret_PlusOne**: this is the return from the time Ret_180 is measured on day D to the close of trading on day D+1. (approximately 1 day). **This is a target variable you need to predict as {id}_61.**
- **Ret_PlusTwo**: this is the return from the close of trading on day D+1 to the close of trading on day D+2 (i.e. 1 day) **This is a target variable you need to predict as {id}_62.**
- **Weight_Intraday**: weight used to evaluate intraday return predictions Ret 121 to 180
- **Weight_Daily**: weight used to evaluate daily return predictions (Ret_PlusOne and Ret_PlusTwo).

Figure 2: Training set, testing set and data fields description

After reading the dataset, we perform exploratory data analysis and realize that we need to do pre-processing to replace missing values and normalize the input data. Once that is done we apply various models such as Sequential Neural Network, LSTM and LSTM on relative returns.

3.2. Data Set 2

For Part 2, we are using real time stock prices provided by Alpha Vantage by using it’s free API keys. We are specifically using the TIME_SERIES_DAILY API keys. This API returns daily time series i.e. date, daily open, daily high, daily low, daily close, daily volume of the global equity specified, covering 20+ years of historical data. The most recent data point is the prices and volume information of the current trading day which is updated real time.

After reading the dataset, we perform exploratory data analysis and gain insights in data. We also perform rolling mean and return rates measurement. After the analysis, we do feature engineering by getting high-low percentage and percentage change. We then perform the usual data-preprocessing by replacing the missing values. Thereafter, we get a cross-validation set. Once this is done, we apply various models such as linear regression, QDA and KNN regression. To evaluate our models, we use the scoring method.

3.3. Workflow

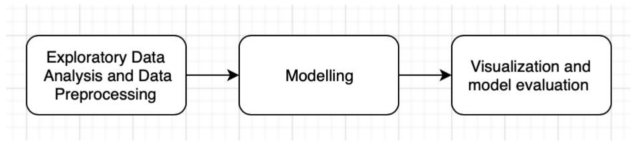

Exploratory Data Analysis and Data Preprocessing
Modelling
Visualization and model evaluation

Figure 3: Workflow for Part 1: Historical data set

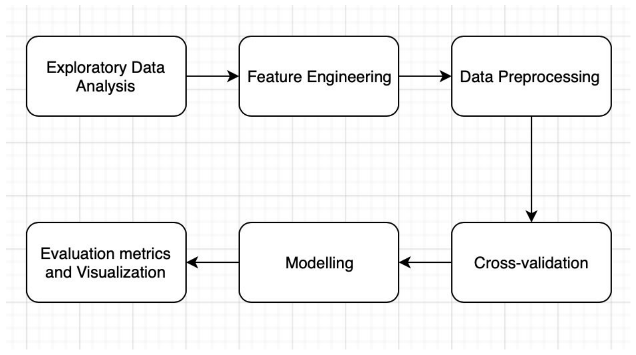


Figure 4: Workflow for Part 2: Real time data set

3.3.1 Software packages used

The packages used for the entire project are listed as follows:

a) Numpy
b) Pandas
c) Scikit-Learn
d) Keras
e) Matplotlib
f) DateTime

4. Algorithm

Since we are using two different data sets, this section is divided into two: Part 1 using historical data set and Part 2 using real time data set.

4.1.Part 1: Using historical data set provided by Winton

4.1.1. Methods/Implementation

a) Data Pre-processing

For data pre-processing, the first thing we do is replace the missing values. We replace the missing data with the mean of the data of each column. Then, we split the data into two: training dataset and testing dataset. Finally, we use MinMaxScaler that subtracts the minimum value in the feature and then divides by range. The range is the difference between the original maximum and minimum. We use this function to normalize the input data i,e. It preserves the shape of the original distribution without changing the information embedded in the original data. It doesn't really affect the outliers in terms of importance.

b) Models

Keras Regression

Keras regression was giving us 100% prediction accuracy i.e. zero loss which means it was overfitting the data. This is not desired. Hence, this model was rejected.

Sequential Neural Network

Since the Keras Regression failed, we moved to the next model: Keras Sequential Model. To build the model, we used three fully connected dense layers. Additionally, we used ReLU activation function and Adam optimizer. After fitting the model, we plotted and calculated the loss. The mean square error came out to be 9.47 and mean absolute error came out to be 5.86. We thought that the performance can be further improved by avoiding the problem of vanishing gradient/short-term memory and hence, went for the next model which is LSTM.

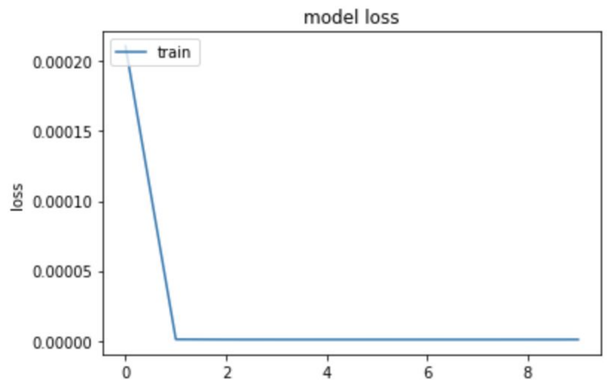
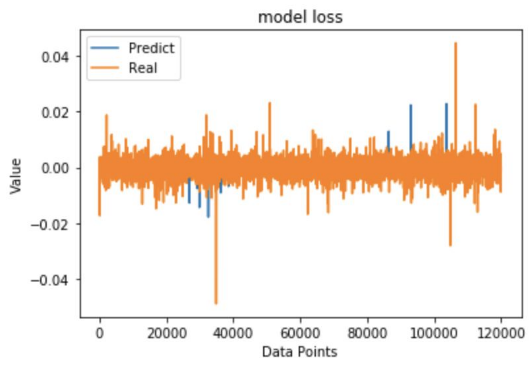


Figure 5: Model loss plot(train) for Sequential neural network

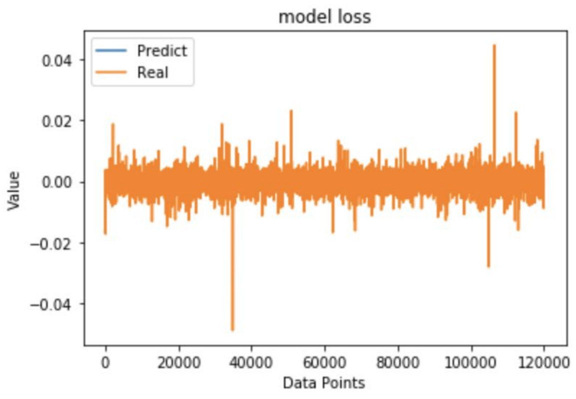


```
array([[-4.0185423e-06],
       [-4.0185423e-06],
       [-4.0185423e-06],
       ...,
       [-4.0185423e-06],
       [-4.0185423e-06],
       [-4.0185423e-06]], dtype=float32)
```

Figure 6: Model loss plot for Sequential neural network

LSTM

LSTM or long short term memory is an artificial recurrent neural network architecture that was developed to deal with the exploding and vanishing gradient problem that can be encountered when training traditional RNNs. A typical LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. In LSTM, the cell remembers the values over arbitrary time periods and the three gates regulate the information flow into and out of the cell. LSTM networks are well-suited for time series data since there can be lags of unknown duration between important events in a time series. Thus, we build a LSTM model with various dropout and dense layer. After fitting the model, we plotted and calculated the loss. The MSE was 9.47 and the MAE came out to be 5.86.



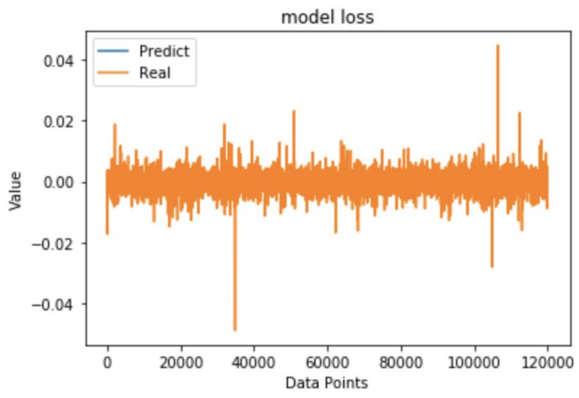
```
array([[0.00019559],
       [0.00019703],
       [0.00019701],
       ...,
       [0.00019777],
       [0.00019841],
       [0.00019862]], dtype=float32)
```

Figure 7: Model loss plot for LSTM

We thought this model can be further improved and therefore went for LSTM with relative returns.

LSTM on relative returns

To further improve the model, we used LSTM model on relative returns. After fitting the model, we plotted and calculated the loss as can be seen in the figure below. The MSE was 9.41 and the MAE was 5.77.



```
array([[-0.00018277],
       [-0.0001808 ],
       [-0.00018328],
       ...,
       [-0.00018785],
       [-0.00018224],
       [-0.00018004]], dtype=float32)
```

Figure 8: Model loss plot for LSTM on relative returns

c) Experiment results

For evaluating our models, we calculated 'mean squared error' and 'mean absolute error' as can be shown below in the figure.

Evaluation metrics:

Model	No. of Epochs	Loss (MSE)%	Error (MAE)%
Sequential	10	9.54	5.92
LSTM	10	9.47	5.86
LSTM on relative returns	10	9.41	5.77

Table 1: Evaluation metrics for part 1 models

We can say that LSTM with relative returns gives us the best prediction accuracy of 94.23%.

4.2. Part 2: Using Real time data set provided by Alpha Vantage

For Part 2, we are using the real time data provided by Alpha Vantage. We are using the free API key to fetch the data.

4.2.1. Methods/Implementation

a) Exploratory Data Analysis

We use Pandas library to fetch the data. First thing we do is to read the global equity data which gives us open, high, low, close and volume and plot the intraday time series.

```
                     1. open  2. high   3. low  4. close  5. volume
date
2019-12-13 16:00:00  74.3100  74.3200  74.2900   74.2900    54277.0
2019-12-13 15:59:00  74.2900  74.3300  74.2900   74.3200    47568.0
2019-12-13 15:58:00  74.2300  74.2900  74.2300   74.2900    24800.0
2019-12-13 15:57:00  74.2250  74.2400  74.2250   74.2400     7914.0
2019-12-13 15:56:00  74.2200  74.2300  74.2000   74.2250    35263.0
...                      ...      ...      ...       ...        ...
2019-12-09 09:35:00  73.2700  73.2700  73.2500   73.2600     4900.0
2019-12-09 09:34:00  73.2700  73.3000  73.2700   73.2700    11460.0
2019-12-09 09:33:00  73.2300  73.2900  73.2300   73.2600    12960.0
2019-12-09 09:32:00  73.3000  73.3200  73.2300   73.2500     9993.0
2019-12-09 09:31:00  73.4031  73.4706  73.3105   73.3131   101592.0

[1922 rows x 5 columns]
```

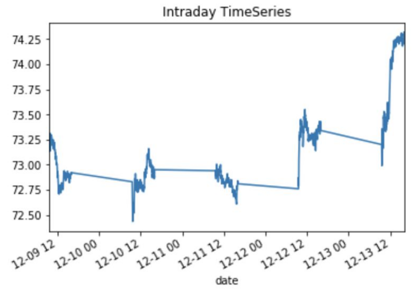


Figure 9: Intraday TimeSeries plot

We then fetch the Bollinger bands which are a technical analysis tool. These bands consists of: a simple moving average which is the middle band, an upper and lower band. The upper and lower bands are basically two standard deviations away from the middle band. The closer the stock prices are to the upper band, the more overbought the market is and the closer to the lower band, the more oversold it is. Alpha Vantage has a total of 52 technical indicators.

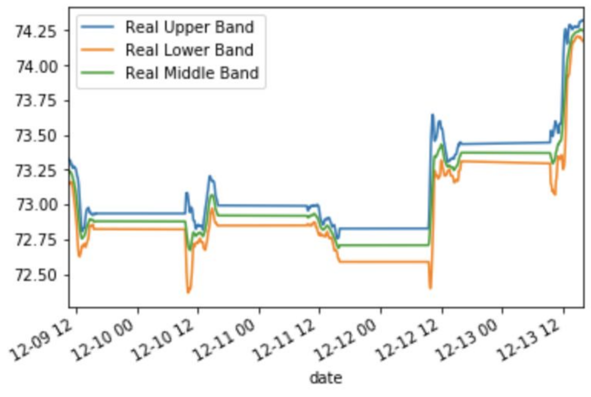


Figure 10: Bollinger bands as a tool for technical analysis

The real analysis begins now! We analyze our stocks using two key measurements: Rolling Mean and Return Rate.

Rolling Mean:

Rolling mean or the moving average is a technical analysis tool that smooths out the price data by creating a constantly updated average price. The moving average helps minimize the amount of noise on a price chart. It also acts as "resistance" i.e. it will follow trends and it is less likely to deviate outside its resistance point. For our project, we take the moving average of the last 100 windows of stocks closing price and take the average of each of the window's moving average. From the figure below, you can see that the rolling mean/moving average doesn't follow the jagged line of the stocks price chart. It sort of makes the line smooth and shows the increase(upturn) or decrease(downturn) trend of stock price.

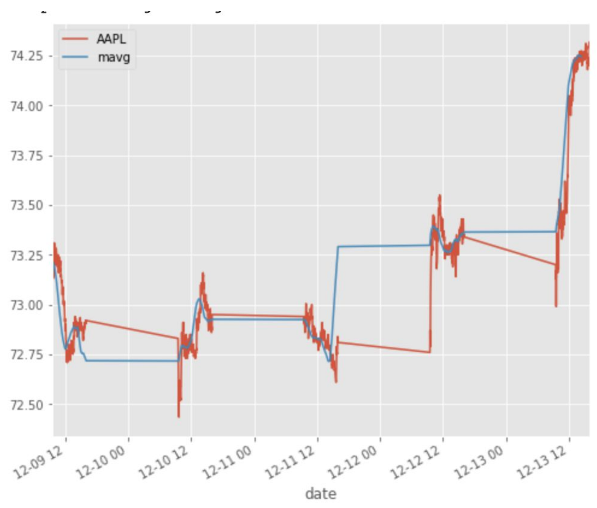


Figure 11: Rolling Mean plot

Return Rate:

Return rates helps determine the risk and returns. The expected returns measures the expected value or the mean of the probability distribution(PDF) of investment returns. It is calculated based on the formula below:

$$r_t = \frac{p_t - p_{t-1}}{p_{t-1}} = \frac{p_t}{p_{t-1}} - 1$$

Figure 12: Return rate formula

From the formula we can see that the expected return of a portfolio is calculated by multiplying the weight of each asset with its expected return and adding each value for each investment. Once we calculate the return rates we plot the same as shown below.

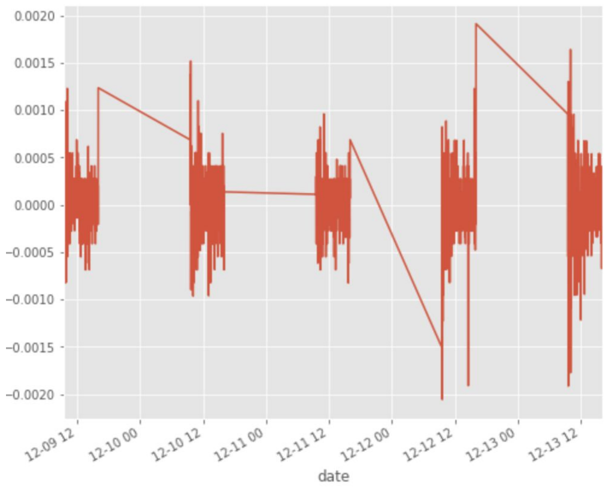
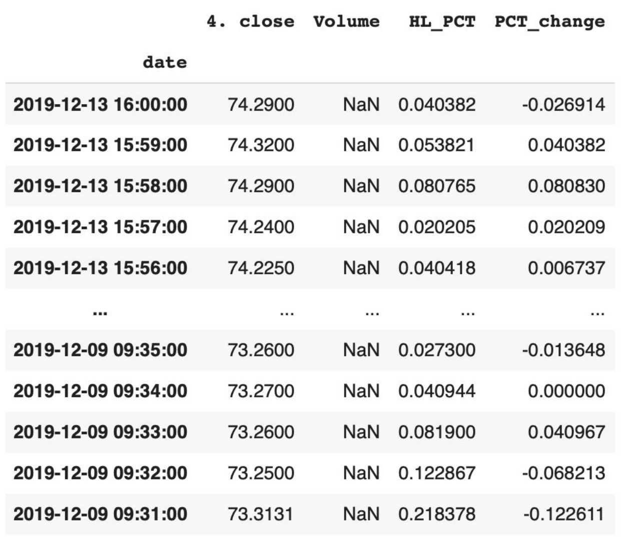


Figure 13: Return rate plot

The ideal stocks are the ones for which the returns are high and stable. If there is a major dip, one should avoid that particular stock. For example from the figure above you can say that stock at 12-13 12 is particularly unstable as there is a dip.

b) Feature Engineering

To engineer features for Part 2 of our project, we calculated “High Low Percentage” and “Percentage Change”. High-low percentage change is a breadth indicator that measures the percentage of Net New Highs. It is a lagging indicator that helps define the medium to long term trend. Percentage change is representative of degree of change over time and positive values indicate percentage increase whereas negative indicates percentage decrease. Below is a table tabulating values for both high-low percentage and percentage change.



	4. close	Volume	HL_PCT	PCT_change
date				
2019-12-13 16:00:00	74.2900	NaN	0.040382	-0.026914
2019-12-13 15:59:00	74.3200	NaN	0.053821	0.040382
2019-12-13 15:58:00	74.2900	NaN	0.080765	0.080830
2019-12-13 15:57:00	74.2400	NaN	0.020205	0.020209
2019-12-13 15:56:00	74.2250	NaN	0.040418	0.006737
...	...	...	...	...
2019-12-09 09:35:00	73.2600	NaN	0.027300	-0.013648
2019-12-09 09:34:00	73.2700	NaN	0.040944	0.000000
2019-12-09 09:33:00	73.2600	NaN	0.081900	0.040967
2019-12-09 09:32:00	73.2500	NaN	0.122867	-0.068213
2019-12-09 09:31:00	73.3131	NaN	0.218378	-0.122611

Figure 14: High low percentage and percentage change

c) Data Pre-processing and Cross-validation

Before fitting the data to prediction models, it is very important to clean and process the data. For data preprocessing we will first replace the missing values and then scale X to maintain a uniform distribution for linear regression. Additionally, we also separate labels and separate the training and testing of models by cross validation train test split.

d) Models

For models, we are using Scikit-Learn package.

Simple Linear Analysis

For our problem definition, we started off with the most basic machine learning algorithm: linear regression since it determines the relationship between the independent variable(s) and the dependent variable. It is also a useful measure for technical and quantitative analysis. In our case, we are trying to define a relationship between price and time and a stock’s price and time period determine the system parameters for linear regression, making the method universally applicable. Below are the results after fitting the model and predicting on test set. We have done a bar plot and a line plot between the actual and predicted values here.

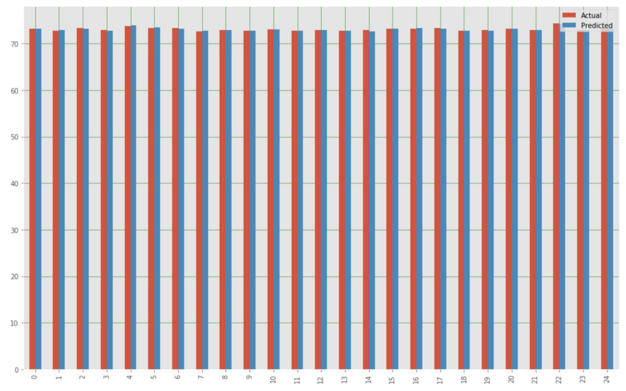


Figure 15: Bar plot for linear analysis

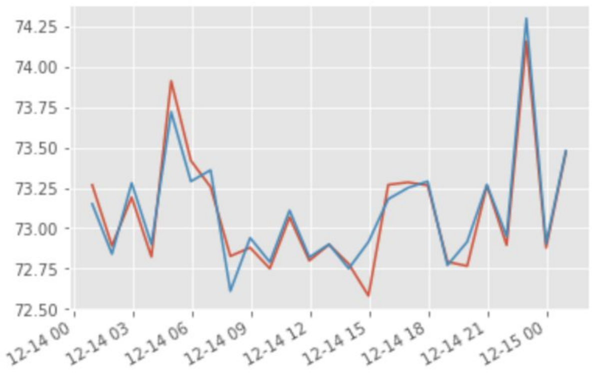


Figure 16: Line plot for linear analysis

From the figure above you can say that the results are quite accurate(94.27%) as the predicted(blue) value is close to the actual value(red).

Quadratic Discriminant Analysis

After linear analysis, we thought we should go for something that gives us more than a linear boundary decision i.e. a non-linear decision boundary. Thus, we chose to pick QDA models for line of fit 2 and another line of fit 3. After fitting the model and predicting on test set, we plotted a bar chart and a line chart between the actual and predicted values as shown in the figure below.

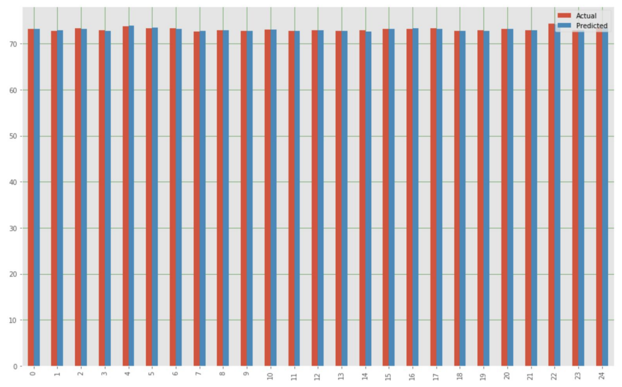


Figure 17: Bar plot for QDA with line of fit two

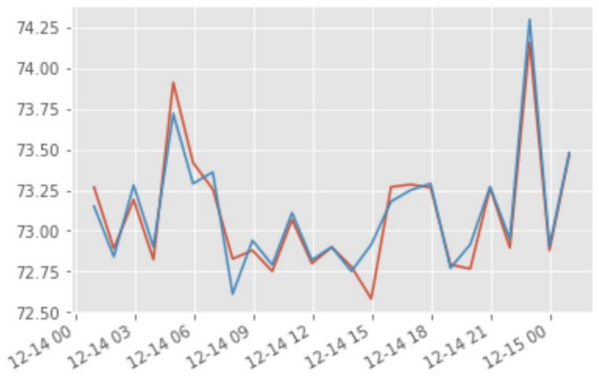


Figure 18: Line plot for QDA with line of fit two

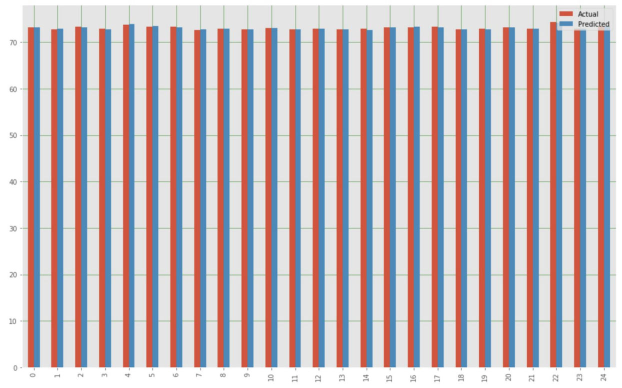


Figure 19: Bar plot for QDA with line of fit three

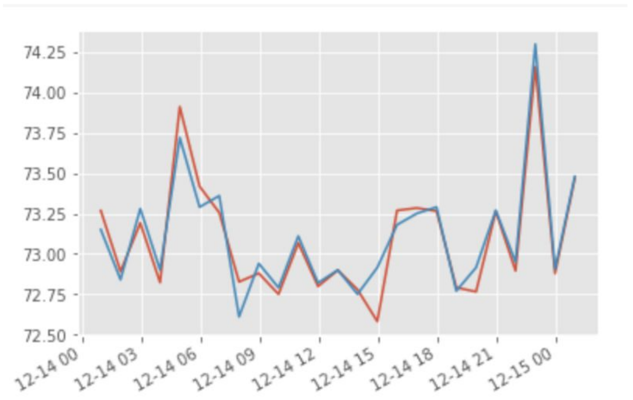


Figure 20: Line plot for QDA with line of fit three

From the figures above we can conclude that QDA with line of best fit of two gives better prediction accuracy(94.48%) than QDA with line of best fit of three which has accuracy(93.03%).

KNN regression

After linear and quadratic analysis, we went for K Nearest Neighbor. The KNN uses feature similarity to predict values of data points. This makes sure that the new point that is assigned is similar to the points in the data set and a good pick of 'k' ensures there is no overfitting. It will basically extract the points to release the minimum Euclidean distance for similarity. So, we fit the data to model, predict and plot line chart between the actual and predicted values as shown in the figure below.

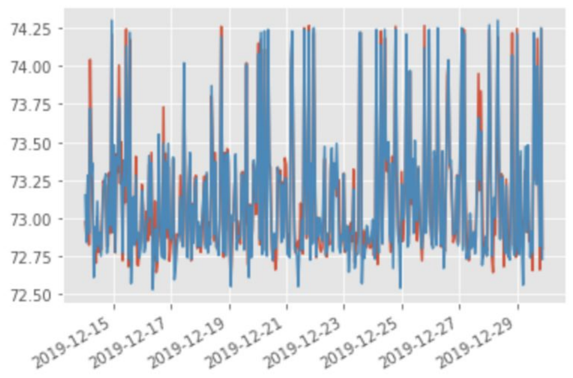


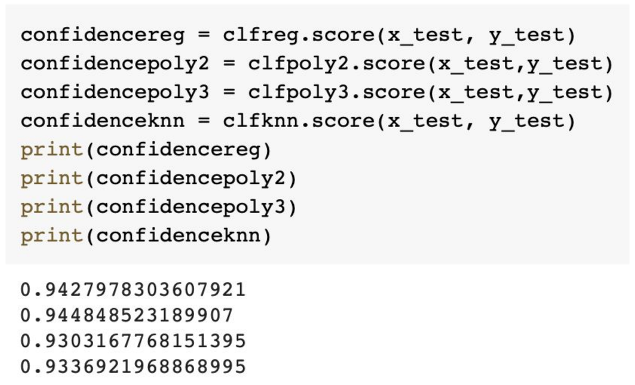
Figure 21: Line plot for KNN regression

From the figure above we can say that predicted values are very close to actual values. Therefore, it is a good fit and the prediction accuracy being 93.36%.

e) Experiment results

Evaluation metrics

For evaluating our models we are using the score method in each trained model. The score method calculates the mean accuracy of self.predict(X) with 'y' of the test data set. The figure below show gives the confidence values for the same.



```
confidencereg = clfreg.score(x_test, y_test)
confidencepoly2 = clfpoly2.score(x_test,y_test)
confidencepoly3 = clfpoly3.score(x_test,y_test)
confidenceknn = clfknn.score(x_test, y_test)
print(confidencereg)
print(confidencepoly2)
print(confidencepoly3)
print(confidenceknn)
```

```
0.9427978303607921
0.944848523189907
0.9303167768151395
0.9336921968868995
```

Figure 22: Scoring method as evaluation metrics

Comparing the results i.e. the prediction accuracy/scores from the figure above we can say that QDA with line of fit two gives us the most accurate results. Therefore, it is the final model of choice.

5. Conclusion

We can conclude that predicting stocks using historical and real time data gives meaningful insights to an investor. It helps make decisions related to trading stocks. Additionally, it's an indicator of the economic situation of a company. Thus, we can say that we successfully deployed machine learning models that gave a great prediction accuracy of around 95% without overfitting.

However, to better analyze the stocks one can experiment with various other kinds of sources like news, tweets etc.

6. References

[1]https://www.kaggle.com/c/the-winton-stock-market-challenge/data
[2]https://www.alphavantage.co/#page-top
[3]https://towardsdatascience.com/the-basics-knn-for-classification-and-regression-c1e8a6c955
[4]https://www.investopedia.com
[5]https://medium.com/@samanamp/fetching-live-stock-market-data-with-python-and-alphavantage-7d0ff8a8d2e4
[6]http://cs229.stanford.edu/proj2017/final-reports/5212256.pdf

7. Appendix

7.1 Individual student contributions

Work	Prutha Parmar prp2126	Prajwal Prakash pp2719	Heetika Gada hg2532
Proposal ppt	33.3%	33.3%	33.3%
Code for Part 1:	20%	40%	40%
Midpoint ppt	20%	20%	60%
Code for Part 2:	60%	20%	20%
Final ppt	33.3%	33.3%	33.3%
Report	60%	20%	20%
Video	33.3%	33.3%	33.3%