Computer Engineering Department
CMPE-255 | Data Mining | Professor David C. Anastasiu

Final Project Report

STOCK PRICE PREDICTION USING HISTORICAL DATA AND TWITTER SENTIMENTS

Team 9
Manika Kapoor (011539203)
Masi Nazarian (011822044)
Sowmya Gowrishankar (005699733)

Fall 2017

Table of Contents

Chapter 1. Introduction

Stock market plays a pivotal role in financial aspect of a nation's growth, and forecasting its behaviour is among the most attractive and yet challenging data mining problems. This a is a highly volatile and complex problem in nature that merely considering market price fluctuations as the basis for prediction seems like an oversimplification. The individual company stock prices as well as market indices are both affected by significant political issues, news and media, analyst calls, company's future plans of expansion and growth, legal fights between companies, new laws and regulations, acquisitions, advent of new technologies, and many more. Moreover, according to the efficient market hypothesis [8], stock prices should follow a random walk pattern and thus should not be predictable with more than about 50 percent accuracy. [4] In this study, we apply several algorithms to forecast stock prices, and compare the results using the Mean-Squared Error (MSE) as the performance metric. The individual company price data were combined with several derived features such as 10-day volatility and 50-day moving average as well as the S&P 500 market index data for the purpose of training some of the models. Also, the prediction results were compared with sentiments analysis of tweets using Twitter's search API, and a meaningful correlation was observed on the predictions and Twitter sentiments. This suggests a possible improvement on forecast performance if enough historical data from Twitter is available and can be combined as an extra feature in the model.

Chapter 2. System Design & Implementation details

Algorithms, Technologies, and Tools

Five algorithms were implemented and compared to forecast future stock prices of time-series data fetched from Yahoo Finance API. As libraries specifically designed for time-series data exploration and forecasting tools, ARIMA and FB Prophet were used. An Artificial Neural Network (ANN) model was also implemented that yield the highest performance among all models. Support Vector Regressor model was also implemented which showed best results after ANN and ARIMA. Linear regression model was also developed that showed comparatively good performance on stock price time-series data. Below is a brief description of each algorithm used, along with their possible cons and pros.

1. Artificial Neural Networks

Artificial Neural Networks (ANNs) or connectionist systems are computing systems inspired by the biological neural networks that constitute animal brains. [9] They can be viewed as weighted directed graphs in which artificial neurons are nodes and directed edges with weights are connections between neuron outputs and neuron inputs. [10]

2. ARIMA (AutoRegressive Integrated Moving Average)

ARIMA is a model designed for time series data either to perform data exploration or forecasting the future points in the series. It is especially useful for non-stationary data, where an initial differencing step i.e. the "integrated" part of the model, can be applied one or more times to eliminate the non-stationarity. [3] The AR part of ARIMA indicates that the evolving variable of interest is regressed on its own lagged (i.e., prior) values. The MA part indicates that the regression error is actually a linear combination of error terms whose values occurred contemporaneously and at various times in the past. The I (for "integrated") indicates that the data values have been replaced with the difference between their values and the previous values (and this differencing process may have been performed more than once). [3]

3. Facebook Prophet

Prophet an open-source package by Facebook that implements forecasting for time-series methodology. It is based on an additive model where non-linear trends are fit with yearly and weekly seasonality, plus holidays. It works best with daily periodicity data with at least one year of historical data. Prophet is robust to missing data, shifts in the trend, and large outliers. [2] It also supports modeling logistic growth, where each data point is measured against a maximum possible capacity. [1]

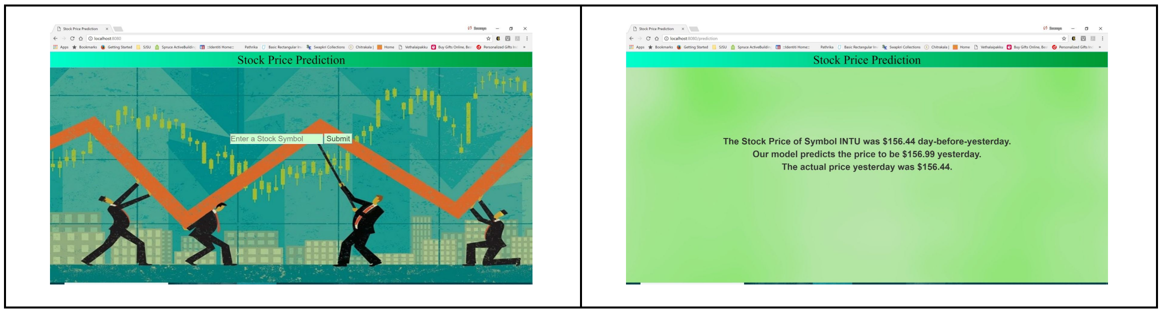
4. Support Vector Regressor

Support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. The support vector regression is a type of SVM that uses the space between data points as a margin of error and predicts the most likely next point in a dataset. [7]

5. Linear Regressor

Linear regressor tries to model the data using a linear function, and predict the future by interpolating the results.

GUI Screenshots



The Stock Price of Symbol INTU was $156.44 day-before-yesterday.
Our model predicts the price to be $156.99 yesterday.
The actual price yesterday was $156.44.

Figure 1. Screenshots from project user interface

Chapter 3. Experiments and Proof of Concept Evaluation

Model Flow Diagram

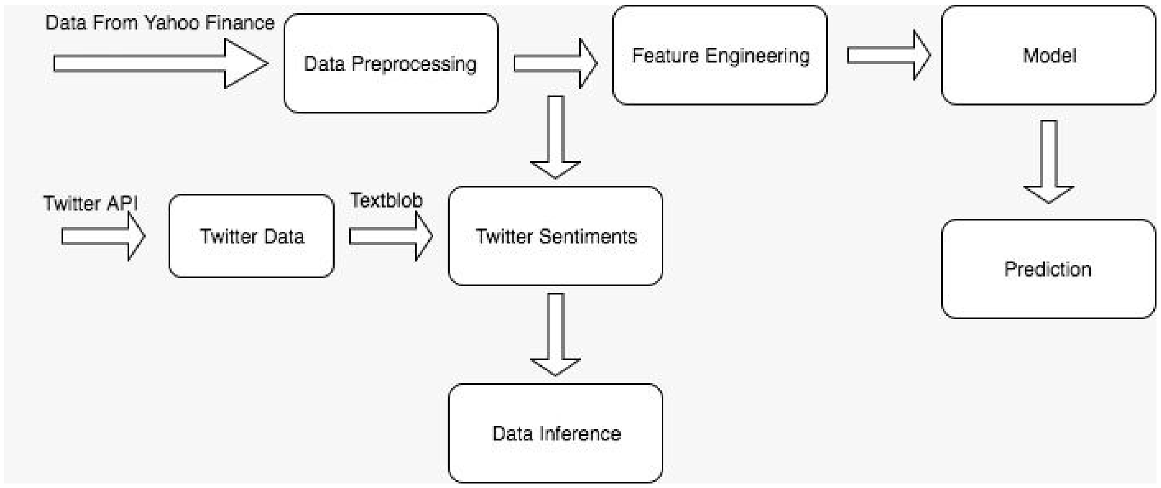


Figure 2. Project flow diagram

Datasets

Almost 7 years of daily price history was fetched from the Yahoo Finance API for different stocks. The evaluation of our model was done using 7 stock symbols, including tech companies, i.e. AAPL, GOOGL, INTU, and IBM; and Energy related stocks, i.e. DWDP, and TOT. The stock market data was also fetched and then the two datasets were combined for prediction. The features in both included open, high, low, and close prices, adjusted close price and volume.

Also, Twitter API was used to get the tweets for the particular company for a particular day so that the twitter sentiments can be used to compare with the stock price fluctuations. **Table 1** shows the number of instances along with start and end date that data is available for each stock.

Source	Dataset	Start Date	End Date
Yahoo! Finance	Stock Price Data	Apr 01, 2010	Present
Yahoo! Finance	S&P Index	Apr 01, 2010	Present
Twitter API	Tweets	Oct 29, 2017	Dec 01, 2017

Table 1. The details of datasets used

Data Exploration

Checking The Time Series Data For Non-Stationarity

Stationary in random time series data means that the statistical values of data do not change over time. This implies that the fluctuations of data around its mean in consistent, and follows no trend. Hence the correlation of time series data with its own prior deviations from the mean do not change over time. Such random variable can be treated as combination of signal and noise. The signal component can be interpreted as a sinusoidal oscillation, pattern of fast or slow mean reversion, or fast sign alternation, plus a possible seasonal component. So an ARIMA model tries to separate signal from noise and then extrapolate the results to future data in order to make predictions. [5]

The closing price was used to check the data for non-stationarity for AAPL stock. For this purpose, the 50-days rolling mean and 10-day rolling standard deviations were plotted. The result shows non-stationarity in the data as shown in **Figure 3 (left)**. This characteristic is also apparent by splitting the data into equal partitions and drawing the histogram of mean and standard deviation for each partition. As shown in **Figure 4 (left)**, the non-stationary nature of the data can be seen in the histogram for Bitcoin prices. As seen in **Figures 3 and 4 (right)** the distribution of data after transformation shows less variation in rolling mean and rolling standard deviation.

Machine learning algorithms typically work better with (or in some cases require) a stationary data in order to generate a reliable model. For that reason, the data was transformed using a log function, to increase stationarity. **Figure 5** shows the correlation between closing prices of the six cryptocurrencies using Pearson similarity measure in form of a heat-map, as well as a correlation matrix.

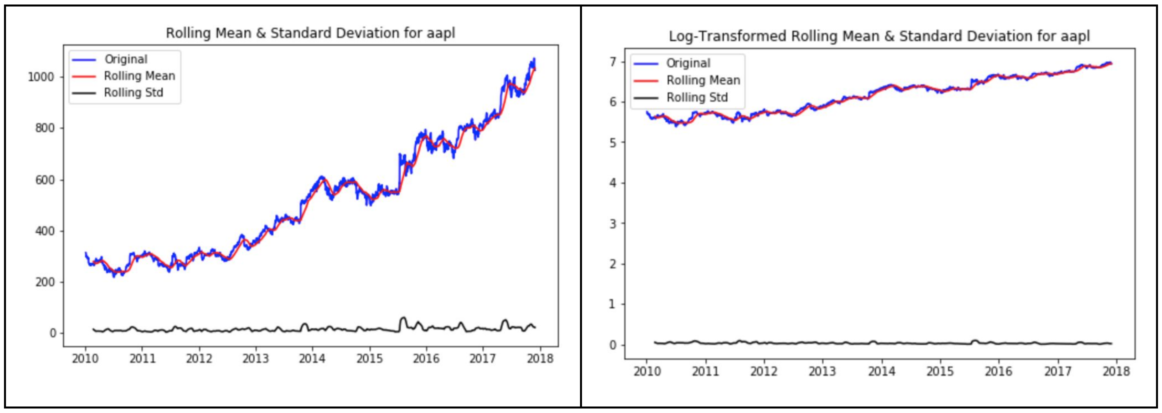


Figure 3. Rolling Mean and Rolling Standard Deviation before (left) & after (right) removing non-stationarity for AAPL

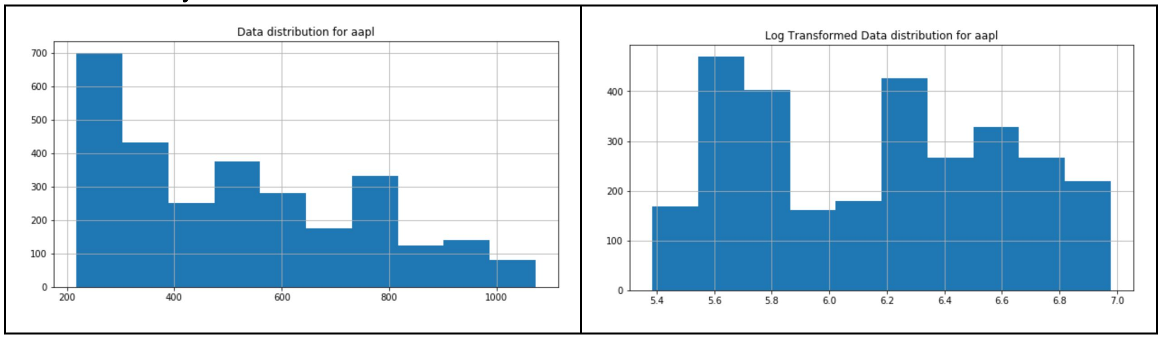


Figure 4. Data distribution before (left) & after (right) removing non-stationarity for GOOGL

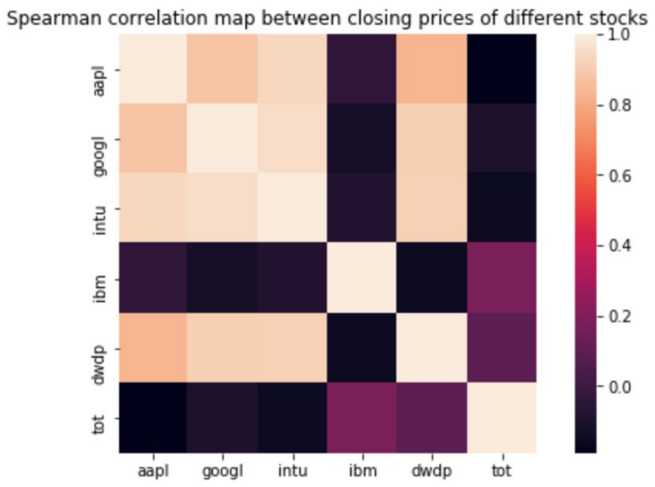


Figure 5. Correlation heat map between closing prices of different stocks

Data Preprocessing

The first step taken in data preprocessing was adding rows with all NaN values for any missing dates, i.e. mostly weekends. After this step all data values were converted to numerical format, and then the attribute values were interpolated for the weekends. Also after combining the two datasets, all the rows with missing values were removed from the dataset.

Feature Selection

The stock price data initially had 6 features, including open, high, low, close, and adjusted close prices, along with volume. Since individual company stocks are likely to be affected by the whole stock market, we also fetched the data for S&P 500 index and included its features in the dataset. Also some common derived features in stock market analysis were calculated and included in the model. Those features include prev_diff that is the difference in closing price of any given day, and the day before it, 50d moving average that is the 50 day window moving average of closing price, and ten_day_volatility that is the rolling standard deviation for 10 day window of the closing price.

The **S&P 500** market index data was then combined with current data, and lastly any dates with missing values were dropped from the dataset.

The final dataset features include open, high, low, and close prices, adjusted close price, volume, **prev_diff**, **50d**, **10d_vol**, **sm_open**, **sm_high**, **sm_low**, **sm_close**, **sm_adj_close**, **sm_volume**, **sm_prev_diff**.

In order to select an optimum feature set, the correlation matrix between initial features of the dataset were calculated. Since open, high, and low prices showed high correlation with the closing price and with each other, only the close column was kept, and the other three columns were deleted from the data. The Pearson correlation metric was used to calculate the correlation between attributes. The “Kandall” and “Spearman” correlation metrics could be alternatively calculated for this step. **Figure 6** shows the results in from of heat map.

Considering the sequential nature of time series data, the train/test split was done such that the sequence is maintained. Three different split ratios were examined, i.e. 80/20, 90/10, and 95/5, and 90/10 split yielded the best performance measured by Mean Squared Error.

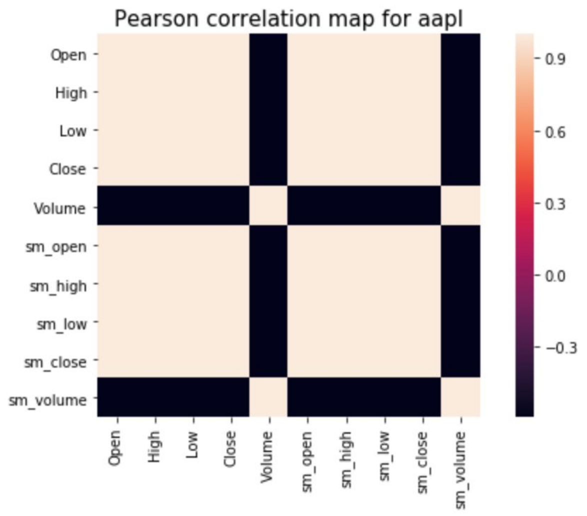


Figure 6. The correlation matrix between features of the AAPL dataset after normalization

Results

Performance

Overall, ANN was the winner algorithm in terms of MSE performance, followed by ARIMA and SVR. Logistic Regression also performed relatively well, followed by FB Prophet in the list.

Table 2 shows the Mean-Squared Error values, comparing the performance of different algorithms on price forecasting of different cryptocurrencies.

	ARIMA	FB Prophet	ANN	SVR	LR
AAPL	1.493	1639.284	0.638	2.065	1.430
GOOGL	48.482	11647.023	43.5441	83.290	44.347
INTU	1.241	79.222	0.357	1.804	1.177
IBM	1.544	1143.301	3.77034	1.779	1.667
DWDP	0.265	19.465	0.160506	0.334	0.249
TOT	0.139	4.837	0.00597506	0.132	0.208

Table 2. The MSE of different algorithms on different stock data

Artificial Neural Network Graphs

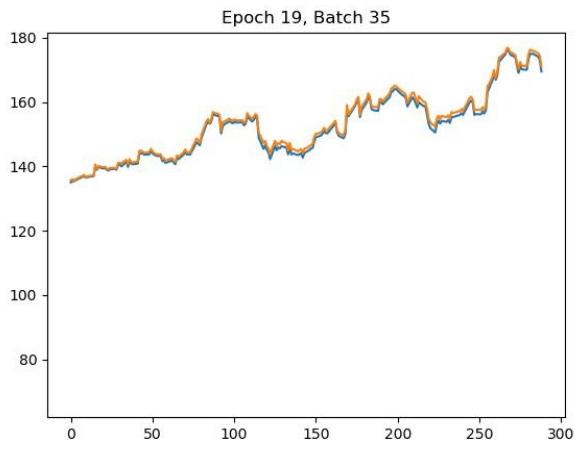


Figure 7. ANN prediction results for AAPL

ARIMA Graphs

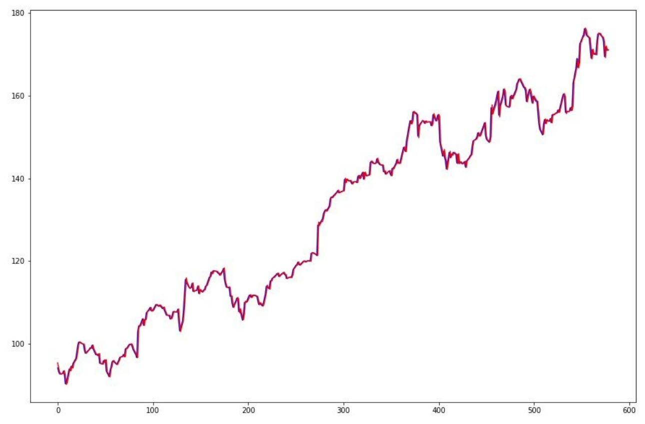


Figure 8. ARIMA prediction results for AAPL

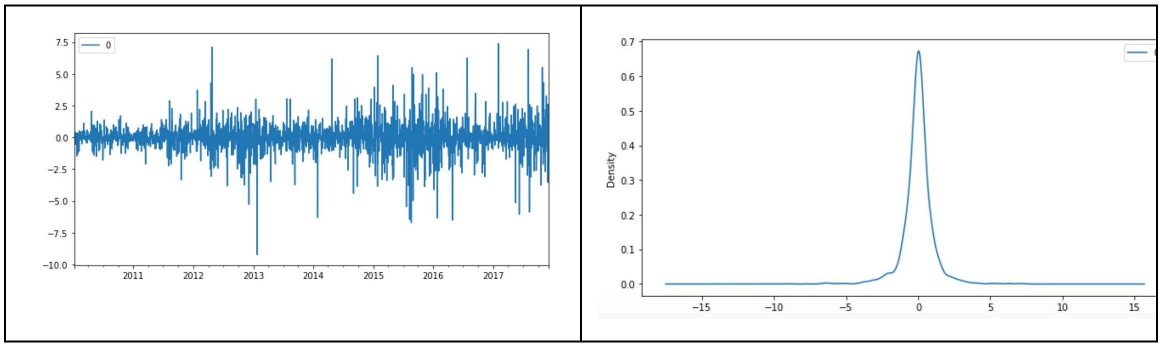


Figure 9. The residual graphs produced by ARIMA

Facebook Prophet Graphs

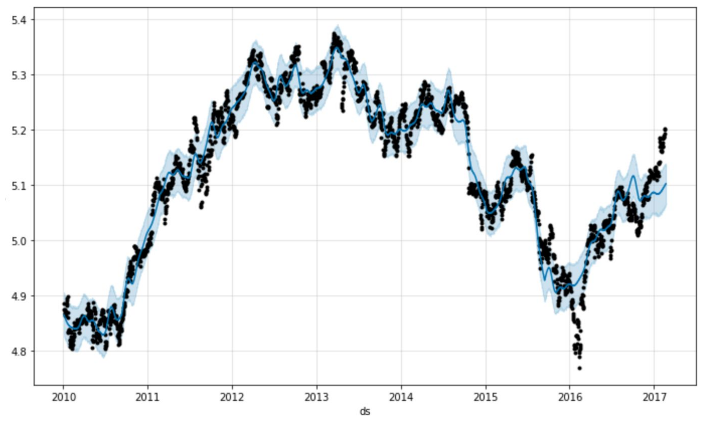


Figure 10. FB Prophet forecast for AAPL

SVR Graphs

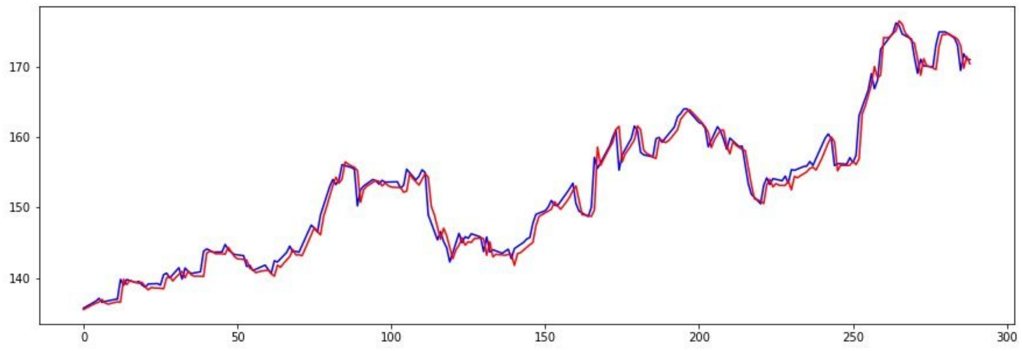


Figure 11. SVR prediction results for AAPL

Linear Regression Graphs

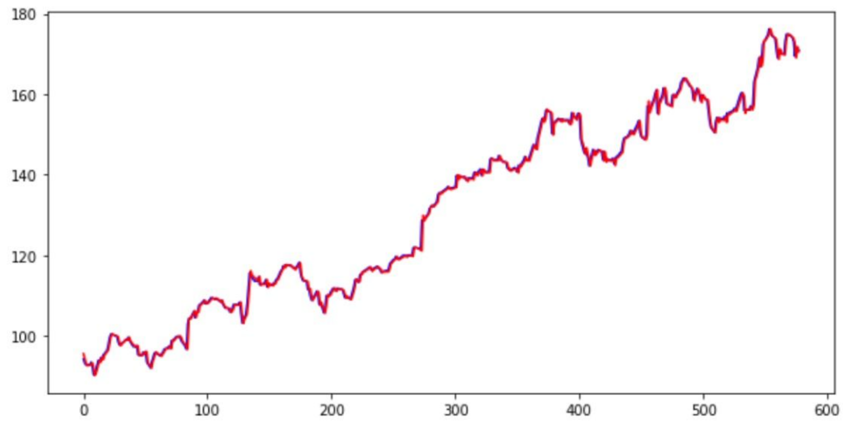


Figure 12. Linear Regression prediction results for AAPL

Stock Price Comparison with Twitter Sentiments

The tweets specific to 15 companies were collected for a month and then the python text analysis package, TextBlob was used to get the average polarity of the tweets. This average polarity is compared against the change in the stock price to see whether the twitter sentiments can be used as a feature to predict stock price.

It was observed that generally when the twitter sentiment was positive, there was a growth in stock price and vice-versa. The data cannot be used as a feature for stock price prediction as the twitter dataset was very small (only one month). So, we used it for inference.

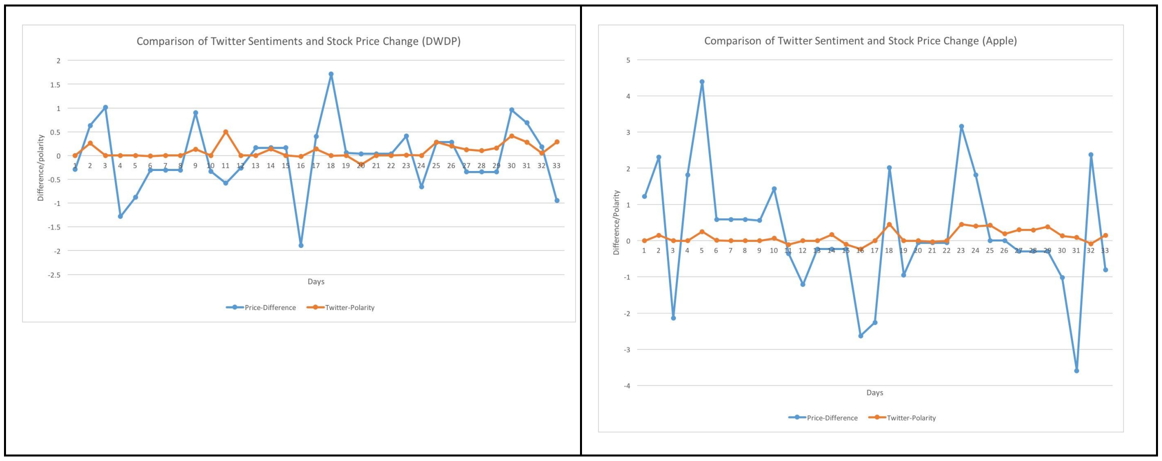


Figure 13. Correlation between Twitter sentiments and stock price fluctuations

Chapter 4. Discussion and Conclusions

Decisions, Difficulties, and Discussion

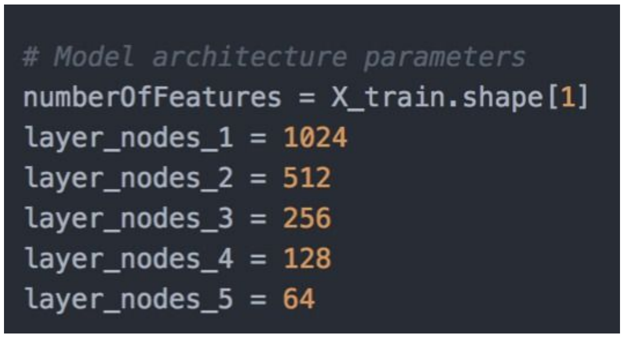
Time-series data analysis, especially stock market, is a highly sought after and hot area. Considering the inherent complexity of such markets, and large number of factors that can affect its behaviour, it is not an obvious or a routine data mining task to come up with a model that performs consistently well.

For the above mentioned reason, our analysis may have given more reliable results if the current datasets were augmented with some side information, such as news headlines relevant to each stock symbol, or relevant tweet analysis. However, the team decided to focus on exploring the algorithms well-suited for time-series analysis, and dive deeper on understanding how those algorithms/libraries work. The result was very satisfactory for us, since we ended up learning about the details of some algorithms and libraries specifically designed for time series data. We also learned about common time-series data exploration practices.

Among all algorithms used, ANN and ARIMA gave the best results in terms of the performance and reliability of their forecasts. ARIMA library had handful of very useful built-in methods to explore and preprocess time-series data.

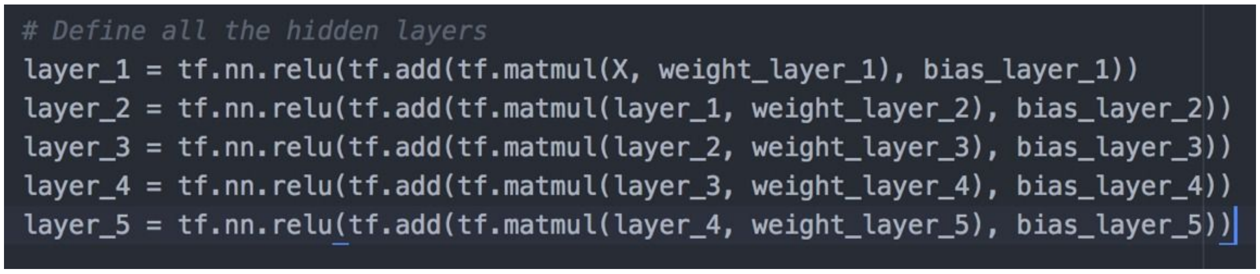
Conclusion and Future Work

For ANN, the best results were achieved using 5 hidden layers with relu activation function. **Figure 14** shows number of nodes in each layer, and **Figure 15** shows the layer architecture.



```
# Model architecture parameters
numberOfFeatures = X_train.shape[1]
layer_nodes_1 = 1024
layer_nodes_2 = 512
layer_nodes_3 = 256
layer_nodes_4 = 128
layer_nodes_5 = 64
```

Figure 14. ANN number of nodes in each layer



```
# Define all the hidden layers
layer_1 = tf.nn.relu(tf.add(tf.matmul(X, weight_layer_1), bias_layer_1))
layer_2 = tf.nn.relu(tf.add(tf.matmul(layer_1, weight_layer_2), bias_layer_2))
layer_3 = tf.nn.relu(tf.add(tf.matmul(layer_2, weight_layer_3), bias_layer_3))
layer_4 = tf.nn.relu(tf.add(tf.matmul(layer_3, weight_layer_4), bias_layer_4))
layer_5 = tf.nn.relu(tf.add(tf.matmul(layer_4, weight_layer_5), bias_layer_5))
```

Figure 15. ANN layer architecture

Figure 16 shows features extracted by FB Prophet, indicating trend and seasonal components of data. As a future work, we plan to combine side information with FB Prophet to heighten the reliability of predictions.

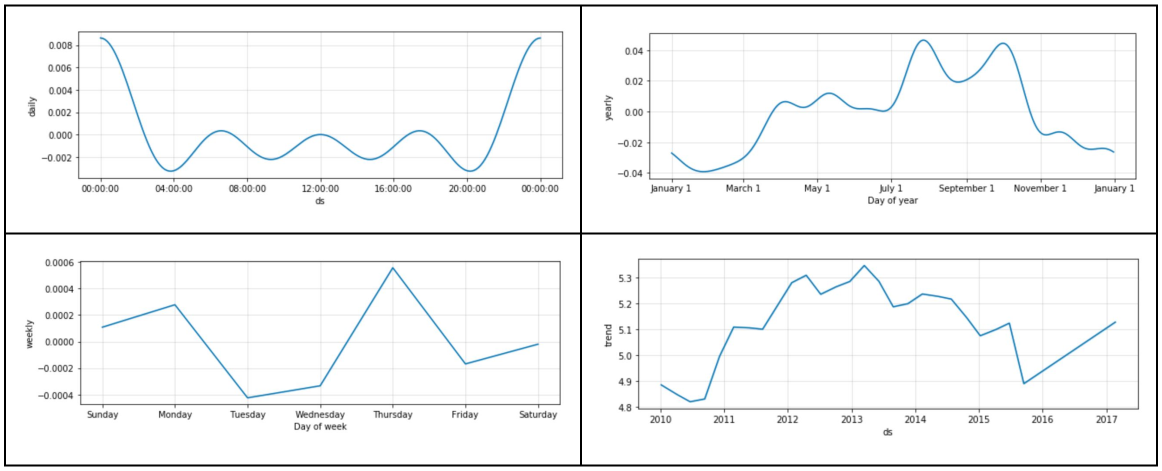


Figure 16. The overall daily (TopLeft), yearly (TopRight), weekly (BottomLeft), and trend (BottomRight) seasonal components of AAPL prices generated by FB Prophet.

Chapter 5. Project Plan / Task Distribution

Initially, the team decided to assign each team member an algorithm to apply on the available data. However, since time-series analysis was new to all of us, we ended up exploring and discussing all algorithms together. Roughly speaking, Sowmya spent the most time on ANN model and GUI, Manika on the ARIMA model, Masi on FB Prophet, and all of us equally learned and contributed on building the SVR and LR model. The data preprocessing and exploration was also done together, since it was the very initial step and we had to resolve several errors together.

Project GitHub Repository: https://github.com/Masea/stock-market-analysis
The Datasets: https://finance.yahoo.com/ | Twitter data: Python's tweepy function

References

[1] http://blog.revolutionanalytics.com/2017/02/facebook-prophet.html
[2] https://facebook.github.io/prophet/
[3] https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average
[4] Stock market prediction with multiple classifiers Bo Qian · Khaled Rasheed
[5] https://people.duke.edu/~rnau/411arim.htm
[6] http://colah.github.io/posts/2015-08-Understanding-LSTMs/
[7] https://en.wikipedia.org/wiki/Support_vector_machine
[8] https://en.wikipedia.org/wiki/Efficient-market_hypothesis
[9] https://en.wikipedia.org/wiki/Artificial_neural_network
[10]https://hackernoon.com/overview-of-artificial-neural-networks-and-its-applications-2525c1addff7