Dao Vang
SpringBoard – Capstone Project 1: Thesis
Mentor: Srdjan Santic
3/22/2020

Stock Predictions Using Data from News Sources, Financial Resources, and Historical Stock Prices

Abstract

The purpose of this capstone project was to predict the price movement of individual stocks using analysis on current and past trends (uptrend, downtrend, or no trend), published articles from various news outlets, earnings reports, and recognition of a common trading pattern. The problem in predicting the movement of the stock market seemed to mostly concern the multiple different variables that affect the stock price. Some important variables, such as political events, are difficult to transform from qualitative information to quantitative-numerical values that are justifiably scaled to the importance of the variable concerning the individual stock. Other variables require technical knowledge or investment experience to understand how these variables could affect stocks. By using random forest, AdaBoost, and gradient boosting as the machine learning models, the results of this capstone project showed R^2 values mostly ranging from 0.90 to 0.99 for stocks that did not return errors when performing 5-fold cross-validation. However, it was concluded that the R2 values do not have any useful meaning to provide to an investor. The results showed potential in predicting the direction of stock prices by using a procedure that combined the statistics of two trained machine learning models.

Introduction

Trading in the stock market as an individual trader can be compared to gambling at a casino—where luck and skill dictate if the trader will be losing or gaining on their investment. However, in contrast to gambling, stock traders are allowed a vast array of computational tools to assist in their investment decisions. These tools simplify statistical calculations on present and historical stock prices and transform those calculations into visuals that can be quickly interpreted when investors need to make fast decisions. However, most of those tools do not perform calculations on data that does not come numerical formats—that is, data which is in the form of words are not analyzed by these programs. When overviewing the data that these tools use, the calculations performed are generally only done on the stock price changes throughout a certain period. Although these calculations can be useful, they do not provide decisive information for everyday stock trading. Unless the trader is with an investment firm/company will plenty of monetary assets to utilize before making investment decisions, traders continue to stay more in the realm of gambling with increased risks in each trade they make.

In the trading world, there are three main types of traders. Institutional traders are traders that are employed by companies with the financial capabilities to make large and long-term investments on behalf of the company or its clients. Some of these companies are venture capitalists, private investment firms for individuals wishing to gain a higher return on their savings, and companies that manage retirement savings. The next type of trader is the day trader. These traders will typically make investments and remove/exit their investments in the same or next day, looking for quick gains on the fast movement of the price during the trading hours. Day traders typically trade in only a few companies at a time each day but do not attempt to understand the company's financials as they will most likely invest different stocks the next day. The last trader type is the swing trader, which is a hybrid between a day trader and an institutional trader. Swing traders will invest in stocks that they expect will continue moving in the same trend over a certain period. After estimating that the trend will end or finding a

better stock to invest in, swing traders will exit their investment. However, unlike institutional traders, swing traders are typically limited in the number of funds they can use to invest and cannot handle the risks of holding their investments the same length as institutional traders do.

With the recent developments in machine learning, it should be possible to create algorithms that are more informative on helping traders make investment decisions. The difficulties in creating a system that can perform the decision-making task become apparent when attempting to quantify variables that are originally in the form of words. The first difficulty before even attempting to interpret data was identifying the variables that could potentially be important in determining the price direction of the stock. The next step was to decide how and in which form will the variables fit into the modeling. Then, if required, the step of transforming the data into a form that the model will understand comes up next.

Although many variables were not covered in this project, the project attempted to use data from news articles, trading patterns, and earnings reports with historical stock prices to assist in predictions made by the modeling algorithms—each of these four variables tried to cover certain areas that could affect stock price movements.

News articles inform the public about the actions of a company, the actions taken against the company, or public opinions on the company. In all cases, any news about a company can be likely to affect its business and therefore, the direction its stock prices. Examples of how news can affect stock prices can be seen in everyday news. If a company reports that it will be filing for bankruptcy, it is expected for the stock price to fall to the cents per share. If a pharmaceutical company passes a clinical trial on a developing drug or product, the stock price can dramatically jump up by tens of dollars from its current price. A current and large-scale effect is the coronavirus, which has drastically decreased the price for almost all stocks.

Day and swing traders often rely on identifying trading patterns, which most patterns seemingly are traded without any logical explanation. Setting institutional trading aside, which do not rely on trading patterns, daily traders look for patterns that are often displayed by the statistical tools. The trading pattern that was focused on this project is the interception between the 30-day moving average and the 60-day moving average. Moving average, concerning stocks, is the average of a price that is calculated over a time range, which moves as new days are added. In actual pattern recognition by traders, 30 and 60 days no longer seem to be the typical standard for this pattern. However, since there was no logical reason that was found to trade under this pattern, 30 and 60 days were used in this project. Recognition of trading pattern is an example of attempting to understand and include the trading behaviors of individual and small investors. Understanding how large/institutional and small investors think in various situations plays a huge part in how stock prices move as large investors can manipulate the stock market, which forces the small investors to follow.

The foundation of all stock price is the earnings reports, which informs the public on the performance of the company over the time between earnings report. Opinionated, earnings reports are the bases of stock prices because the price of the stock will correct itself back to the earnings or the projected earnings of the next earnings reports. Earnings reports inform investors on what that company's stock price should be and how the company might expect to perform in the next earnings report, which is important to institutional investors as they typically make long-term investments. In contrast, many day and swing traders typically do not trade in companies that are soon to report

earnings as these traders do not typically analyze the financials of companies to be able to better determine what the company might report in their earnings reports. Also, as the price of the stock corrects itself to the price in the earnings report, trading patterns and trading behaviors used by day and swing traders are minimized, reducing the signals used by these traders to invest and increasing the risk of the gamble.

Lastly, to include the main data that is used by all traders, the historical price of stocks creates a basic understanding of the trading behaviors that have been and could be used in a particular stock, the reputation of a company, and the opinion of institutional traders. When examining the historical price movement of a stock, it is possible to identify where events might have occurred, such as trading patterns, news events, and earnings effects. It is also possible to see how reliable a stock/company was by cross-checking large drops/gaps in price with news, such as company executives being prosecuted by government investigators. Since many of the stocks of a company are being held long-term by institutional traders, when institutional traders invest or exit a stock, the price of that stock will increase or decrease and other traders will follow the behavior of the institutional trader, especially since the institutional spent a large amount of money and time before making those investment decisions.

In this project, these four types of variables were used to create machine learning models for each stock. The methods used in this project were data extraction through APIs and web parsing, data manipulation, and data analysis using simple statistics. The machine learning models created were using random forest, AdaBoost, and gradient boosting. The R^2 values in most models were above 0.90, including the averaged cross-validation scores (if there were no errors in the execution of the code in a for loop). However, it was found that the R^2 values had no meaningful indications on the predictive performance of the model and that the frequency that the models predicted the price change in the correct direction was to be the main indicator of the predicted power of each model. In conclusion, this project could provide potentially a well-performing procedure for obtaining a predictive value close to the true price value.

Methods

To begin the project, a list of stocks was needed to be obtained, which would become the scope/stock coverage of the project. This was obtained by finding resources that provided the lists in .csv files. However, since the website used specifically in this project was lost during a computer reformat, the website was not be provided.

After obtaining a list of the stocks, the historical price of stocks was required. Using the API provided by Alpha Vantage in conjunction with the list of stocks, it was possible to obtain almost all the stocks on the list. For reasons unclear, this API does not perform well to request with special characters. As a result, some of the stocks were skipped on purpose. Another unclear oddity of the API is that rerunning the script will not always pull the same number of stocks in each iteration. This API does have daily restrictions for free users, which were followed in this project.

Once the historical price was obtained, the next variable to gather was news articles. There were two sources for news articles used in this project, which were the New York Times API and web parsing Google news searches. For free users of the New York Times API, the daily restrictions were followed in this project. Since the Google news searches were done using web parsing, it was theoretically possible to perform the searches without restrictions. However, Google has a detector that

prevents multiple pull requests within a certain period. By trial-and-error, about 500 requests can be pulled every 4-5 hours. A method to work around the detection by Google is to connect to different VPNs in the US after every 500 requests. (The VPNs must be in the US if using the scripts in this project as the language of the requests was not specified.) In the end, although the New York Times API was used and considered in the machine learning modeling stage, the articles pulled from the API were not significant nor required (explained in the conclusion). In Google searches, it should be noted that only search results were parsed, not the actual website of the search results. This resulted in potentially missing the relevant information in the article that was not displayed in the headlines of the search results.

The next variable obtained was the trading patterns. The dates of all occurrence of where the 30-day and the 60-day moving averages intercepted were identified and recorded. This used the historical price as the base data.

The last variable to obtain was the earnings reports. This was obtained by web parsing Yahoo Finance to obtain the earnings summaries and statistics for most stocks. Some stocks were either not covered by Yahoo Finance or had different ways to report their earnings reports.

After the data for all the variables were obtained, each variable (except for the historical stock price) needed to be given a value that is relevant to the importance of that variable to the price movement.

For scoring and evaluating the words in the articles and search results, all the words from all the results containing the company's name were parsed and separated into a .csv file. The .csv file, at the time, contained about 1300 words with the number of articles the word appeared in. Then the file was manually examined, and values were given for each word. Each word was given two values: a sentence value and an importance value. The sentence value meant that the word had a positive or negative meaning in the sentence (similar to a verb). Examples of these words with sentence values are sell, buy, drop, and increase. Importance value meant the word could have a large influence on the stock in general (like earnings and bankruptcy). All the sentence values are added together and separately, all the importance values are added together. Then the total sentence values are multiplied by the total importance values to give the text value. Examples: (A conversion of "0" in the examples means that the sentence value and importance values—depicted by the *—are both zeroes.)

Example 1:	Example 2:
Sentence = The stock will increase in price due to earnings. Conversion = 0 0 0 (+5, *0) 0 (+ 0, *2) 0 0 (+0, *10) Text Value = 5 * (10 + 2) = 60 Here, the word "increase" has a sentence value of +5, while "price" and "earnings" have 0 sentence values but have an importance value.	Sentence = The dog ran up the hill. Conversion = 0 0 0 (+5, *0) 0 0 Text Value = 5 * (0) = 0 Here, the word "up" has a sentence value of +5. However, since there is no word with an importance value, the word "up" results in not being informative and ends up with a text value of zero.

In both examples, and for many of the manually valued words, valued words typically only have a value for the sentence value or the importance value, but not both. There were a few words that were assigned values for in both sentence and importance values, such as the word "bankruptcy", which has negative values for both sentences and importance values. After all the words have been evaluated, a

new .csv file was created to contain only the words with non-zero values in either sentence or importance values—this was done to attempt to speed up the searching process. The text of the articles was then valued based on the values of each word to provide the text value for the article. Only New York Times articles with the company's name were evaluated for a text value. All Google search results were considered relevant to the company being searched and no screening was performed before scoring the results.

The purpose for the two values for evaluated word was to eliminate the need to bring in a library/method to perform the word evaluations, allows the word evaluations to be tailored for identifying texts that are relevant to trading, and to remove the need to identify words as adjectives, verbs, and nouns. In consideration of time, to incorporate a new library or method was not ideal as manual corrections had to be made in the end to tailor the word evaluations for trading. Popular methods of evaluating words were not necessary as the words that were being examined were simply to assist in determining if the stock price was going to go up or down, while many of these methods attempt to derive more complex and multi-categorical meaning. As such, the popular method of vectorization of words and hashing, suggest by the SpringBoard curriculum, was also not used. In the placement of these methods, the manual assignment of two values to each relevant word seemed to perform as intended with good enough results.

The next score to valuate was the earnings report. Although the earnings report is a numerical variable, the scoring was done to make the earnings reports more informative to the machine learning algorithm on the estimate earnings report done by Yahoo. Simply, if the estimated earnings of the company were positive, then the earnings score was a large positive value (relative to other scorings). And if the estimated earnings were expected to be negative, then the scoring was negative.

The next variable to score was for the trading patterns of the interception between the 30-day and 60-day moving averages. This trading pattern has two general expected price movements. If the 30-day moving average is intercepting from above the 60-day moving average, then the price is expected to downtrend. If the 30-day moving average is intercepting from below the 60-day moving average, then the price is expected to trend up. The next days in the range of 2 days to 30 days after the interception was examined to find how long the trends might last. The length of the trend was determined by fitting lines of best fit over the date ranges. The scoring was done by giving a higher score depending on how close the two moving averages were together and then added a positive (if the 30-day moving average was below the 60-day moving average) or negative sign (if the 30-day moving average was above the 60-day moving average) to that score. After the interception has occurred, a depreciating value was giving to days where the trend was followed until the best number of days when the trend was expected to stop was reached. The scoring criteria were set to be different depending on the average price of the stock as a to attempt to minimize over predicting for stock with lower values as a prediction of a $1 increase in the price of a $15 is a large difference than the same price change in a stock valued at $100 per share.

The last score to assign was for the historical price of the stock, which was used to determine what trend was the stock currently experiencing. The reason for this was that since many of the modeling algorithms shuffle the rows before training the model, there needed to be a way to maintain historical data/direction of the stock in a value that will shuffle along with that row in training. Therefore, the 1-day moving average was determined and the averaged slope between the 1-day moving averages of the previous five, seven, and ten days was calculated. The calculated slope was a

crude estimate that contained information from previous days to maintain the price direction of the previous days and provide a rough estimate of the price of the next day (before the model predictions). However, depending on the machine learning model used, this method required the removal of the first five days of the historical price data as there would be no values for those averages. Since most of the historical price data had many rows/observations, the removal was not expected to be too influential.

A final accumulation of all the scores and values calculated was done and all empty values (NumPy.NaN) were assigned a zero value. The data was labeled using the value of the closing price of the next day (except for the last observation/newest date since there was no next day information). The unlabeled newest date was removed and stored for prediction and manually checking the closing price for the next day.

The data was split in two ways: shuffled and ordered. Since the effects of the coronavirus are new and still occurring and there is no historical data with this much impact in a short time, it was thought it should be better to force the inclusion of the most recent data in the training data (ordered) to increase the change that shuffling during random forest will include some of the recent dates. The dataset training-testing split was done using default settings (75% training data and 25% testing data).

For all machine learning models used (random forest, AdaBoost, and gradient boosting), the script created attempts to find the number of trees that gave the best 5-fold cross-validation averaged R^2 value. For results that returned a negative value for the cross-validation averaged R^2 value, the number of trees was set to 128, which was the highest recommended number of trees after researching for the best default setting. All models, with the best trees or 128 trees as its parameters, were used to predict on the saved last unlabeled observation. Then an average of the predicted values between all model variants was obtained. The model variants trained in this project were:

- Random forest with shuffled dataset
- Random forest with ordered dataset
- Random forest with ordered dataset and no bootstrapping
- AdaBoost with shuffled dataset
- AdaBoost with ordered dataset
- Gradient Boosting with shuffled dataset
- Gradient Boosting with ordered dataset

The Python libraries used in this project were Pandas, NumPy, JSON, requests, RE, BeautifulSoup, SciPy, Sci-Kit Learn/sklearn, and Matplotlib.

Results

The statistics for the success rate of the interception trading pattern was examined and resulted in an overall averaged frequency of 55.8%, the percentage of times the pattern was followed when the 30-day moving average was intercepting from above the 60-day moving average (the price was expected to downtrend), and 68.6% when the 30-day moving average was intercepting from below the 60-day moving average (the price was expected to uptrend). If the trend moved in parallel with the trading pattern, the trend was expected to last 9.29 days when the 30-day moving average was above and 10.4 days when it is below the 60-day moving average. The summary of the statistical results:

	R^2, Below	R^2, Above	Price Difference, Below	Price Difference, Above	Frequency & Trend Length, Below	Frequency & Trend Length, Above
Days	4.132394366	4.346028646	18.64785276	15.60199387	10.40792836	9.295041525
Stats	0.653859374	0.64612794	9.378212147	-4.959520491	0.686173335	0.558072357

The values provided in the table for "Price Difference" was the subtraction of the closing price of the day that was being examined and the closing price before the interception. Because the statistics for this specific trading pattern was like the performance of a coin toss, the information provided did not seem to be of high importance when determining the direction of the stock price. However, if the trading pattern is followed, a swing trader might be given an average period that the new/current trend could last and provide an idea of when to exit the trade.

Although many high accuracy scores (R^2) were obtained for models set to the best number of trees or defaulted to 128 trees, by visually inspecting and comparing the high R^2 to other statistical values obtained, it was clear that the R^2 does not provide any insight on the performance of all the models for all stocks.

A better value to examine and judge the performance of the model variants was the frequency that the predicted price movements went in the same direction as the actual value. This value informs whether the model was good at predicting in the correct direction—how good the model was at potentially predicting the correct price. Many of the models, primarily AdaBoost and gradient boosting, did poorly and resulted in a ~50-65% frequency of predicting in the correct direction. In contrast, all the random forest variants performed better, providing frequencies in the 75-85% ranges. A problem with using this value to judge the models was that for smaller priced stocks and stocks that do not typically have much activity in daily price movements, it was easy for the models to predict in the incorrect direction as the predicted price was in cents/close to zero price change. However, as this project was mostly geared towards swing and day traders (and not institutional traders), these types of traders will only trade in stocks with high everyday price movements (high volatility).

To better evaluate the frequency of a model's correct prediction of price movement, the examination of the difference and absolute difference between the actual and predicted values were obtained. The average difference between these two values showed if the model tended to over-predict or under-predict. The average of the absolute difference showed how close the predictions were to the actual values. The best consistent and overall performer for all the models was the random forest trained with the shuffled data. This model tended to have the average difference that was closest to zero. However, it did not always have the best average of the absolute difference but did perform better than many of the other model variants. The standard deviations of the difference were obtained to provide a better insight into the predicted price range (+/- from the predicted price). For many of these statistical values, having a value closest to zero was ideal.

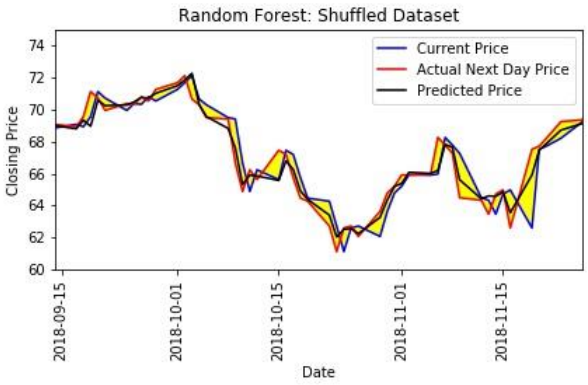
Accumulating all the statistical data, a potential procedure to determine which models to use in trading was to take parts of the statistics of the models:

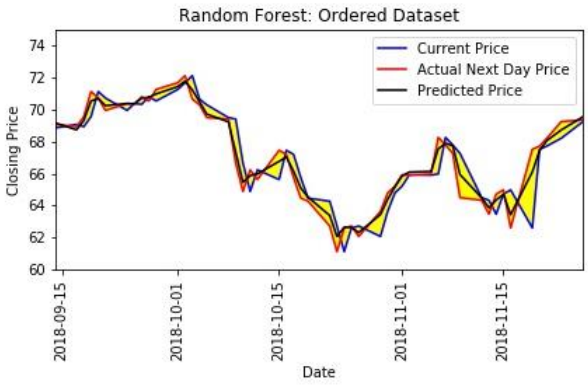
1. Choose the model with the best frequency of predicting the correct price movement. The model variant was typically the random forest with ordered dataset and no bootstrapping.
2. Next, choose the model with the lowest average differences and absolute differences. The model variant was typically the random forest with shuffled dataset.

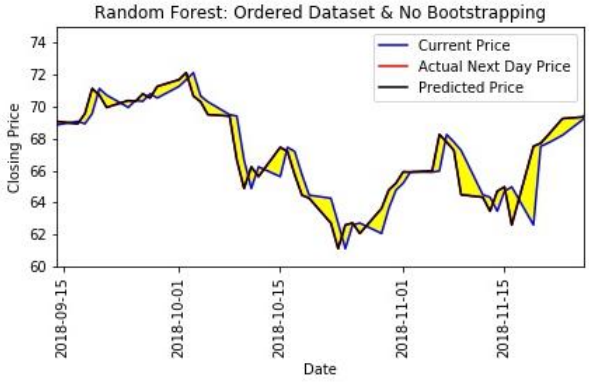
3. If the predicted prices for the models in the first and second step move in the same direction, meaning that the predicted prices are expected to both increase/decrease from the current day, choose the predicted price from the model in the second step.
4. If the predicted price from the model in the second step does not agree with the model in the first step, then use the predicted price from the model in the first step.

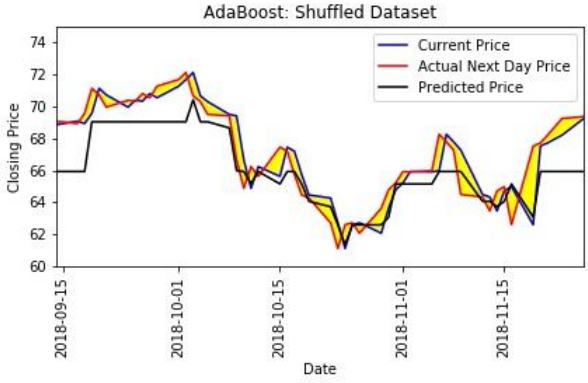
By following this method, this attempts to decrease the risk that the model with the best-predicted price value does not predict in the incorrect direction while increasing the chance that the predicted price will be close to the actual value to maximize the trade. But since there are many stock choices to choose from, it could be better to choose the stocks with high frequency and prediction performance to reduce risk.

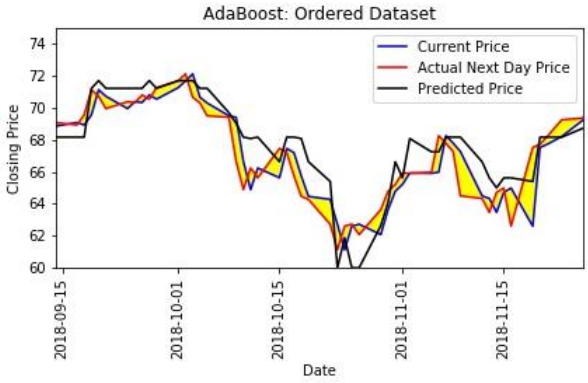
In the figure below, all the model variants are graphed against the current day closing price (blue) and the next day closing price, which was the label of the dataset (red). The prediction values (black) were expected to move within or near the yellow shaded regions, in which the region depicts the true price change. Of all the model variants and only taking into consideration of the graphs, the best model was random forest trained with the ordered dataset and no bootstrapping parameter, predicting the next day closing price almost perfectly. The models that did the worst were both the AdaBoost variants with many of the predictions being largely over or under the next day closing price.

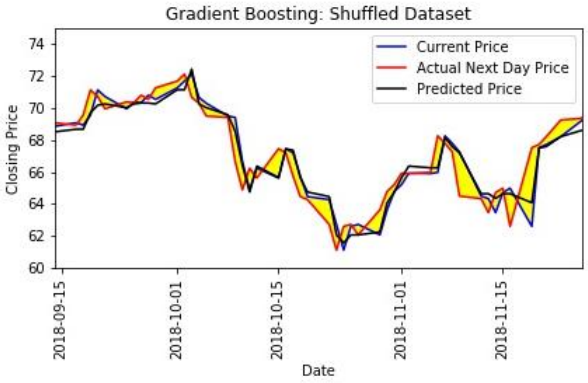

Random Forest: Shuffled Dataset
Current Price
Actual Next Day Price
Predicted Price
Closing Price
Date

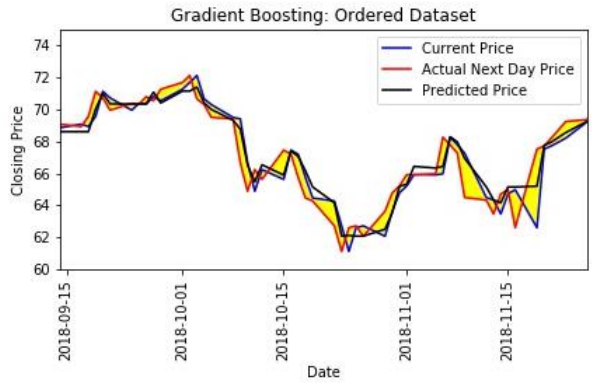

Random Forest: Ordered Dataset
Current Price
Actual Next Day Price
Predicted Price
Closing Price
Date


Random Forest: Ordered Dataset & No Bootstrapping
Current Price
Actual Next Day Price
Predicted Price
Closing Price
Date


AdaBoost: Shuffled Dataset
Current Price
Actual Next Day Price
Predicted Price
Closing Price
Date


AdaBoost: Ordered Dataset
Current Price
Actual Next Day Price
Predicted Price
Closing Price
Date


Gradient Boosting: Shuffled Dataset
Current Price
Actual Next Day Price
Predicted Price
Closing Price
Date


Gradient Boosting: Ordered Dataset
Current Price
Actual Next Day Price
Predicted Price
Closing Price
Date

Conclusion, Issues, and Improvements

To conclude, this project was not able to create a machine learning model that fully understood the stock market effects of the media and earnings reports partially due to API and web parsing issues. However, this project was able to create a potentially viable model to assist traders in trading, which seemed to be stable in performance even in the stock market movements caused by the coronavirus. The best performing model seemed to be the combination of two random forest variants: random forest trained with a shuffled dataset and random forest trained with a dataset that maintains its ordered after splitting (to force train the model with the newest dates) and no bootstrapping option as the random forest parameters.

Some issues that appeared in this project, which were impassable with the resources chosen, created difficulties in achieving the goal of identifying relevant data on the news of companies. Because the New York Times API's search query did not return relevant articles, almost all the articles extracted ended up not being evaluated due to the article not containing the name of the company. Oddly, many of the articles were about irrelevant topics, such as articles about the best trending music when the search query was for a biotechnology company. In contrast, web parsing on Google's search website was largely more effective but was limited due to Google detecting systems that pull too many URL requests. To better assist the models in identifying if the text scoring was important, the scripts were meant to obtain article results for the specific dates around price high peaks and low dips. The New York Times API was able to successfully search between specific dates, but the business coverage for companies not based in New York was not diverse nor immense. Although Google web searches provide tools to perform searches between specific dates, manipulating the URL to perform the search on specific dates oddly returned search results with the latest news relevant to the company. After examining the HTML file, the manipulated URL request was maintained until just before the results were given, in which the URL request changed to a default search URL. Since web design was not part of the SpringBoard curriculum and the time it could have taken to solve these issues was unclear, the method of article searches using specific dates was abandoned.

Another issue in this project was the validity of the historical price information. When cross-checking the price history of stock from Alpha Vantage against the historical price histories from Yahoo Finance and Google, it was clear that some of the values were not the same. This misalignment in data creates potentially invalid models as the dataset that has been used to train the models could have been incorrect.

The last major issue of this project was the performance and time of the scripts. From the start of the project, timing became an issue because of the free usage of the APIs was limited and the detection by Google when web parsing. To extract all the stocks and articles from the Alpha Vantage and New York Times APIs, it would take at least 9-12 days before data for all the stocks could be obtained. For the Google searches, performing pulls for 200 stocks per day resulted in 22 days before data for all the stocks could be obtained. To perform all the calculations on all the stocks could take a few hours to a few days, with the script used to determine the best number of trees for the models taking the longest (maxed at evaluating 250-300 stocks per day on an updated computer). All the scripts in this project typically create multiple files containing repetitive data. However, since storage has been getting more inexpensive and advanced, this 10+ GB project was not an issue. Although the performance of the scripts can be improved dramatically, the time it would take to implement the improvements would be out of the scope of this project.

There are many improvements, in terms of variables and resources, that can be implemented that could better improve the models' performance. One improvement that is similar to earnings that could be added is dividends. Dividend announcements change the stock price with precise increases, which could assist the model in understanding why the price is moving in that direction and amount. Removing the New York Times API and changing it to Yahoo Finance would be an improvement since Yahoo does not seem to detect systems from making large and multiple URL requests (as seen with the script used to pull the information on the earnings reports). The news from Yahoo Finance about companies is also more precise and relevant to investors, which news from Google could be about anything containing the name of the company. Yahoo Finance also allows direct downloads of the historical data that provides the same information that the Alpha Vantage API does. The historical data from Yahoo Finance also gives information on historical dividends, which saves time in attempts to extract that information. For web parsing, using Scrapy or more advanced libraries would allow the web parsing to pull more information from one input URL request. With Scrapy, it would be possible to obtain the actual news article from the original webpage by accessing it from the Google search. This would allow for better evaluations of the articles and give a more diverse score range for all articles stored. The addition of other trading patterns could also improve the models' performance.

Although there was still a lot of work left to do before the project is fully completed, due to time becoming a pressing issue, the project was concluded at this stage and resulted in a stable and well-performing prediction model.