Predicting Apple's Stock Price with Machine Learning and Google Searches

Submitted by Jarred Glaser

Department of Economics

Econon 411 - Forecasting Project Paper

University of Wisconsin - Milwaukee

Spring 2018

1. Introduction

One of the most difficult entities to predict in our world today is the stock market. For many years researches and investors have attempted to more efficiently forecast stock market price changes. However, due to its stochastic nature, consistently predicting stock prices can be very challenging. Regardless, stock price prediction continues to be one of the most popular forecasting tasks today, and the reward for those who can find a way to "beat" the stock market is very large.

Traditionally, stock market prediction has followed the random walk theory and the efficient market hypothesis. The random walk theory was popularized by Princeton economist Burton Malkiel (1973). He stated that the stock market follows a random walk, and therefore cannot be accurately predicted. Under these guidelines Malkiel believed that one was better off simply guessing whether stock prices would go up or down than spending large sums of money paying financial experts to make predictions (Malkiel 1973).

The efficient market hypothesis was developed by Eugene Fama (1965). His hypothesis stated that it was virtually impossible to accurately predict the stock market. This is because, in the stock market, there are many competing actors, with equal access to information, all attempting to maximize profits. Under these circumstances, the stock market acts as an efficient market, where, at any given point in time, current prices always reflect the actual, or intrinsic, value of a given stock. This means that it is impossible for any, one actor to "beat" the market, because every other actor has equal access to the same information.

Although stock market prediction is still largely governed by these theories, many recent papers have begun to challenge these concepts. This recent literature uses advanced statistical techniques that borrow from other fields such as machine learning and computer science to create more accurate stock market predictions (Weng, 2017; Choudhry and Garg 2008; Preis, Moat, & Stanley 2013). Advances in computational efficiency and artificial intelligence over the past decade, make these methods extremely effective. Using these methods combined with newly available datasets gathered form online interactions, some research has had success in creating more accurate stock price predictions than previously thought possible (Weng, 2017).

This paper attempts to build on recent literature in this field by using machine learning and disparate data sources collected from Google Trends to predict stock market changes. I predict stock closing price directional changes for Apple's stock (AAPL). This transforms the problem from regression to classification (the stock price went up: 1 or the stock price did not go up: 0). For forecasting, I use a popular machine learning model called random forests. To improve the predictability of the model I will use a combination of popular financial technical indicators as well as Google search volume data collected form Google trends as features in the model. After computing the models, I compare the effectiveness of a model that uses just technical indicators and one that uses both technical indicators and Google search data to see which performs better.

2. Literature Review

Traditionally, stock market prediction has relied on the idea that stock markets follow a random walk model, and therefore cannot be accurately predicted. However, new research shows that taking advantage of new machine learning methods and data collection techniques can lead to an improvement in stock market forecasting. This research has been made possible by recent advancements in computing power in the past decade.

2.1 Prediction with Technical Indicators

Much of the recent research has focused on using machine learning models with technical indicators as inputs to predict stock market changes. Technical indicators are simply different equations that use historical prices as inputs to try and predict future trends and price movements in stocks (Weng, 2017). Using these indicators to predict stock prices is nothing new (Vaiz and Ramaswami, 2016). However using them as inputs for machine learning models has shown to be worthwhile in some recent literature.

Madge (2015) finds prediction accuracy improvements over longer forecast horizons with a support vector machine model using technical indicators as inputs. However, these improvements diminish as the forecast horizon shortens. Choudhry and Garg (2008) find stock price forecast prediction accuracy of up to 61% using a hybrid Genetic Algorithm – Support Vector Machine model with technical indicators as inputs. They find that this hybrid model shows significantly better results than just a standalone SVM model. Di (2014) uses a radial basis function SVM algorithm to predict stock price trends for Apple, Amazon, and Microsoft, achieving prediction accuracy up to 77% for 3-10 day forecast horizons. While SVM's have been proven to be very effective in predicting stock price changes, Khaidem, Saha, and Dey (2016) show that using a tree based, random forest model for predicting 30, 60, and 90-day stock price directional changes can lead to prediction accuracy of as much as 96%.

2.2 Prediction with other Variables

While technical indicators can be very useful tools in predicting stock prices, recent research reveals that other variables can be used alongside technical indicators as inputs for machine learning models. These variables often come from a wide range of newly available information from online resources. This internet-based information can be a useful predictor for stock price changes as more of society transfers its daily activity to online interactions.

Preis, Moat, and Stanley (2013) show that by using Google search query volumes from Google Trends, trading behavior can be quantified to reveal early warning signs in stock market movements. Weng (2016) finds that Google news results and Wikipedia page hits for Apple's stock can be used as inputs alongside technical indicators to achieve stock price prediction accuracy up to 85%. Varian (2014) discusses several machine learning methods and how they can be used with large amounts of data, including data gathered from online interactions, to create powerful forecasting models.

3. Data Collection

In this paper, I use both technical indicators and search volume data collected from Google Trends as features in the random forest model.

3.1 Technical Indicators

Financial technical indicators are a popular way to predict stock price trends and price changes. There are hundreds of technical indicators, but for the purposes of this paper, I will use a select few. The technical indicators that this paper uses are: the Relative Strength Index (RSI), the Stochastic Oscillator (%K), the Stochastic Momentum Index (SMI), Williams %R, On Balance Volume (OBV), Moving Average Convergence Divergence (MACD), and the MACD signal. For the more on these measures, the reader is encouraged to reference Vaiz and Ramaswami (2016) and Mitchell (2014). All of these indicators were calculated in R using the library TTR (Ulrich, 2018).

3.2 Google Search Data

Google search volume data is collected from a free service provided by Google called Google Trends. Google trends allows a user to retrieve a time series of search volume data on any term. Google Trends only returns daily data if the time selection is less than 90 days. To get around this I wrote a script in the programming language Python that connects to a psuedo API called *pytrends*. The program collects data on any set of search terms in one month increments and combines the data together to get longer daily time series data.

Google Trends also returns data in a relative format. So rather than raw search numbers for a given term, Google Trends returns a set of "scores" between 0 and 100 that represent a ratio of that day's search count to the most searched day in that time period. This score adjusts for each time period selected, meaning that daily data cannot simply be combined together. To get around this, the script normalizes the combined data by using weekly data of each search term to rescale as a relative score over the new longer period (Johansson, 2014). Once the data is combined and normalized, it can be viewed in a time series graph as seen in figure 1.

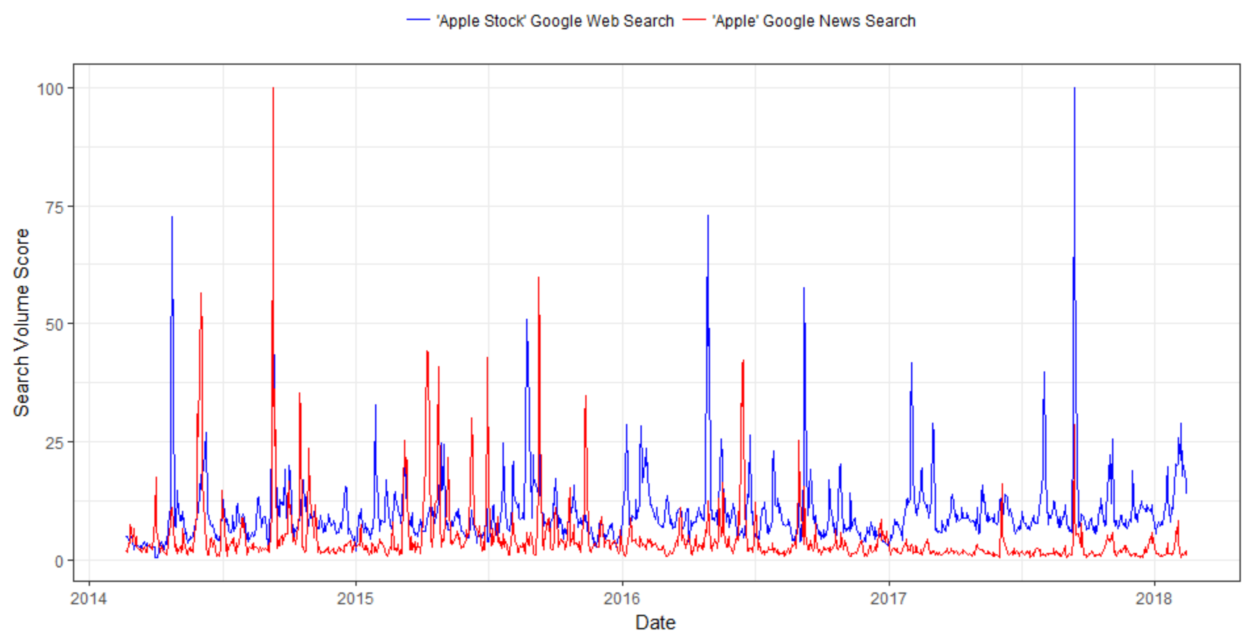

Figure 1: Apple Google Search Volume Data

The above figure shows the daily Google search volume data for the selected terms from 2014 to 2018. The blue line shows the Google web search volume score for the term "Apple Stock". The red line shows the Google news search volume score for the term “Apple”. It should be noted that the above plot shows the all of the data for the selected time period, but when the data is used in the model, all non-trading days are removed.

3.3 Target

The target I am predicting is the directional change of Apple’s closing stock price for a 3, 5, 15, and 30 day forecast horizon. The equation for the target is,

$$\mathrm{Target_d} = \mathrm{Sign}(\mathrm{close_{t+d}} - close_t)$$

The target for each forecast horizon *f*, will take a value of 1 if the closing price in period t+d is greater than the closing price in period *t*, and -1 if the closing price in period *t*+*d* is less than the closing price in period *t*. I then convert the numbers so that the target takes the value 1 if the price goes up and 0 if the price does not go up.

4. Random Forest Model

The random forest model is a very popular and powerful machine learning model that uses a tree based supervised learning method to make predictions. Random forests can be used to predict both regression and classification problems. For this paper, the forecast will be classification.

Random forests are built on a simple machine learning method called decision trees. Decision trees are a rather simple, but very powerful, machine learning algorithm. A decision tree works by splitting the predictor space into various points, thereby creating separate groups to use for prediction. A simple graphical depiction of a decision tree can be seen in figure 2 where a decision tree model is used to predict whether a passenger of the titanic would have lived or died.

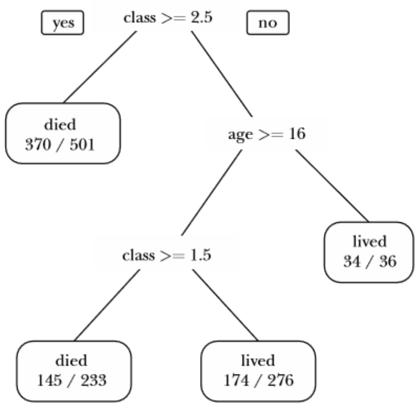

Figure 2: Decision Tree for Titanic Survival (Varian, 2014)

In the above example, the decision tree uses different predictors to split the data into several, non-overlapping regions (James et al., 2013). Those splits are then used to classify out of sample data and make predictions.

To create the splits, the tree uses a top-down, greedy process called recursive binary splitting (James et al., 2013). The initial set of observations is first split into two regions (m_1 and m_2). In the above example this would be the class being greater than or equal to 2.5 or less than 2.5. This split is made by deciding on a predictor variable X_j and a split *s* that minimizes some objective function. For regression the RSS is usually minimized. For classification problems several objective functions can be minimized, most commonly the *Gini Index* (equation 4.1) or *Entropy* (equation 4.2).

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{4.1}$$

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \tag{4.2}$$

where $\hat{p}_{mk}$ is the proportion of observations that are in the m^{th} region in the k^{th} class. After the 1st split is made, another split is created on one of the two newly created regions, minimizing the same objective function. This creates a 3rd region. This process continues until there are enough regions that hold some small number of observations (James et al., 2013).

One major problem with decision trees is that they tend to have very high variance. High variance means that a tree fitted to one set of observations may look very different from a tree fitted to a similar set of slightly different observations. Consequently, decision trees often overfit the data, that is, they tend to work well on in sample predictions, but work very poorly out of sample (James et al., 2013). To fix this, random forests can be used.

Random forests use a method called bagging, where many tree models are estimated on different subsets of the data and the average is taken (or a simple majority vote if the problem is classification) to make predictions. This method greatly diminishes the variance problem. Random forests also only give each tree model a random subset of predictors to use. This process decorrelates the trees and further reduces variance (James et al., 2013).

5. Recursive Feature Elimination

To further avoid overfitting the data, I use a method called recursive feature elimination (RFE). Recursive feature elimination uses a backwards step approach to select a subset of important variables. I use the *caret* library in R to run RFE on the data (Kuhn, 2018). Once all of the features are calculated there are 58 features total in the dataset. I allow the algorithm to select an optimal subset of variables between 1 and 20. The RFE algorithm will first estimate a random forest model using all of the variables, then remove the least important variable and continue backwards, cross validating the model accuracy at each step. Finally, the RFE algorithm selects the optimal variables and number of variables between 1 and 20 to use (Kuhn, 2018). The results of running RFE on the data for each forecast horizon can be seen in figure 3 and 4.

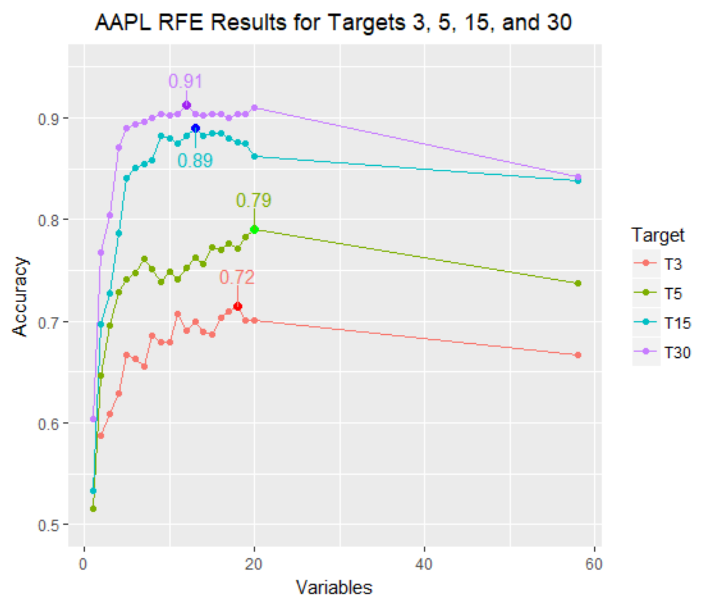

Figure 3: AAPL RFE Path for the different forecast horizons

T3	T5	T15	T30
OBV	MACD_Sig	MACD_Sig	OBV
RSI	OBV	SMI	Apple_Stock_SMA20
Apple_Stock_SMA8_Change	Apple_SMA10	OBV	Apple_Stock_EMA20
Apple_Stock_EMA8_Change	Stoch	Apple_Stock_SMA20	SMI
Apple_EMA10	WPR	MACD	Apple_SMA20
Apple_SMA10	Apple_EMA10	Apple_Stock_EMA20	Apple_Stock_SMA10
MACD_Sig	Apple_SMA8	Apple_SMA10	Apple_EMA20
Apple_SMA8	Apple_Stock_SMA20	Apple_EMA10	Apple_Stock_EMA10
Apple_Stock_Disparity8	Apple_Stock_EMA20	RSI	MACD
Apple_Stock_EMA20	Apple_EMA8	Apple_EMA8	MACD_Sig
Apple_Stock_EMA10	SMI	Apple_EMA20	Apple_SMA10
Apple_Stock_Disparity10	MACD	Apple_SMA20	Apple_EMA10
SMI	Apple_EMA6	Apple_Stock_EMA10	
Stoch	Apple_Stock_SMA8		
Apple_Stock_SMA20	Apple_Stock_SMA10		
Apple_EMA8	Apple_SMA6		
Apple_EMA6	Apple_Stock_EMA10		
Apple_Stock_EMA8	Apple_SMA20		
	RSI		
	Apple_Stock_Disparity8		

Figure 4: RFE Results

T3, T5, T15, and T30 represent the targets for forecast horizons 3, 5, 15, and 30 respectively. Figure 3 shows the RFE selection path for each subset of variables, and the accuracy score assigned to the subset of variables that maximized the accuracy. Figure 4 shows a list of all of the variables selected by RFE for each target. A full list of the different variables and their meanings can be seen in the appendix. One interesting observation about the results, is that for every target, there are Google search related features selected, which means that there is important information for predicting AAPL's stock price changes in the Google search volume data.

6. Results

The random forest model is run four times total: one time for each target. Usually, in machine learning, it is common practice to use some method of cross validation to avoid over-fitting. A popular method of cross validation is k-fold cross validation. However, this method requires randomizing the data. Because we are dealing with time series, this is not an option. Therefore, I use a different cross validation method adjusted for time series called recursive forecasting. Recursive forecasting trains the model on some initial subset of data and then uses the next observation in the series as test data to make a prediction. The results from that prediction are saved and then that observation is added to the training set. This process continues until all observations in the data set have been used. An illustration of this process from a blog post by Rob Hyndman (2016) can be seen in figure 5.

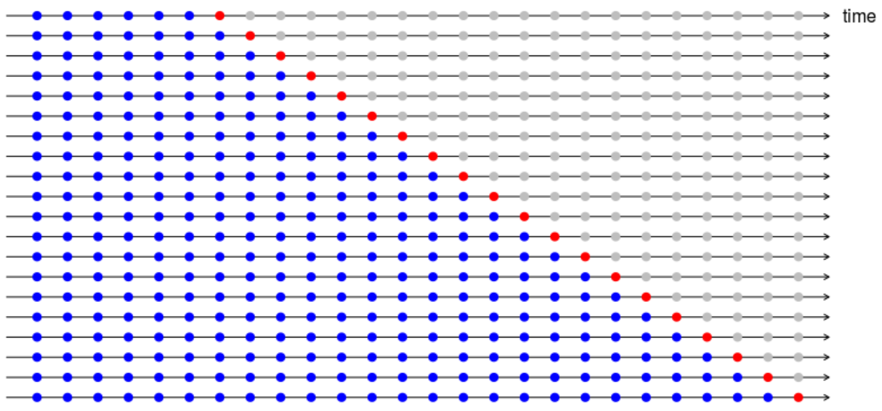

Figure 5: Recursive forecasting (Hyndman, 2016)

In the above illustration the blue dots resemble the training set and the red dots resemble the test set in each iteration.

The final results of the recursive forecast using random forests with the selected features from RFE can be seen in figure 6 and 7.

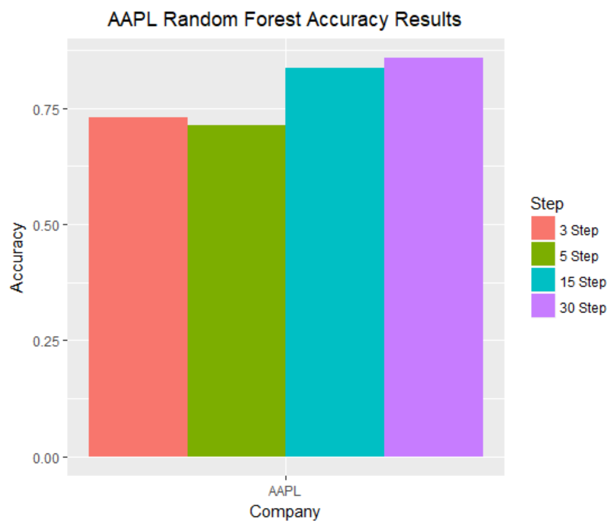

Figure 6: AAPL Random Forest Accuracy Results

	Accuracy	Precision	Recall	Specificity
3 Step	0.731225	0.732591	0.867987	0.527094
5 Step	0.713439	0.719444	0.854785	0.502463
15 Step	0.835968	0.857527	0.91404	0.66242
30 Step	0.857708	0.861333	0.941691	0.680982

Figure 7: AAPL Random Forest Results

Figure 7 shows the corresponding results for the random forest model in predicting stock price directional changes for AAPL at each forecast horizon (3 step, 5 step, 15 step, and 30 step). Accuracy measures the overall proportion of correct predictions made by the model. Precision measures the ratio of true positives to the total true and false positive predictions. Recall measures the ability for the model to predict positive labels (an increase in price), and specificity measures the ability for the model to predict negative labels (a decrease in price). A list of the formulas for these calculations can be found in the appendix.

Overall the model predictions did well. At its best, the model was able to achieve 85% accuracy in predicting the 30 day-ahead target. One interesting note about the results is that specificity is very low compared to the other measurements. This shows that the model does not do as well in predicting when the price will decrease. I also run the model again, but this time only using technical indicators as features. I then compare these results to the results above to see if the features calculated on Google search volume do in fact increase the prediction accuracy for a given target. The results of this comparison are shown in figure 8.

	RF with Just Technical Indicators				RF with Google Features Included			
	Accuracy	Precision	Recall	Specificity	Accuracy	Precision	Recall	Specificity
3 Step	0.54	0.59	0.67	0.35	0.73	0.73	0.87	0.53
5 Step	0.64	0.68	0.78	0.45	0.71	0.72	0.85	0.50
15 Step	0.71	0.77	0.82	0.46	0.84	0.86	0.91	0.66
30 Step	0.80	0.85	0.85	0.69	0.86	0.86	0.94	0.68

Figure 8: Technical Indicators vs. All Selected Indicators

The above figure shows that there is significant prediction accuracy improvement when including Google search volume data features in the random forest model. Therefore, Google search volume data does provide better predictive power over more traditional forecasting models that just use technical indicators as features.

7. Conclusion

Predicting stock market changes is one of the most difficult tasks in the field of forecasting. The stock market is governed by the random walk theory and efficient market hypothesis, leaving many researches to question whether it is possible to make accurate forecasts of stock prices. However, recent advancements in statistical learning methods, computational efficiency, and data collection methods, have made way for the possibility of much stronger forecasts of stock market prices.

We find that random forests prove to be an effective way to predict stock price directional changes. Using popular financial indicators as features in these types of models works well, but adding data collected from internet-based, disparate data sources can have an even greater impact on prediction accuracy. Building upon this research, further improvements in the model could be achieved by adding other online data sources, such as the number of news stories published, or the number tweets made about a given company. As more advancements are made in machine learning, one can expect further improvements to forecasts on stock market prices, leading to great payoffs for researchers and investors.

References

Choudhry, R., & Garg, K. (2008). A hybrid machine learning system for stock market forecasting. *World Academy of Science, Engineering and Technology*, *39*(3).

Di, X. (2014). Stock Trend Prediction with Technical Indicators using SVM.

Fama, E. F. (1995). Random walks in stock market prices. *Financial analysts journal*, *51*(1).

Hyndmann, R. (2016, December 05). Cross-validation for time series. Retrieved May 10, 2018, from https://robjhyndman.com/hyndsight/tscv/

James, G., Tibshirani, R., Witten, D., & Hastie, T. (2013). An introduction to statistical learning-with applications in R. New York: Springer series in statistics.

Johansson, E. (2014, December 07). Global Payment Trends. Retrieved May 10, 2018, from http://erikjohansson.blogspot.com/2014/12/creating-daily-search-volume-data-from.html

Khaidem, L., Saha, S., & Dey, S. R. (2016). Predicting the direction of stock market prices using random forest. *arXiv preprint arXiv:1605.00003*.

Kuhn, M. (2017, September 04). The caret Package. Retrieved May 10, 2018, from https://topepo.github.io/caret/index.html#

Kuhn, M. (2018). caret: Classification and Regression Training. R package version 6.0 79. https://CRAN.R-project.org/package=caret

Madge, S., & Bhatt, S. (2015). Predicting Stock Price Direction using Support Vector Machines. *Independent Work Report Spring*.

Malkiel, B. G., & McCue, K. (1985). *A random walk down Wall Street* (Vol. 8). New York: Norton.

Mitchel, C. (2014, June 25). Ultimate Guide to the Stochastic Oscillator. Retrieved May 10, 2018, from https://traderhq.com/stochastic-oscillator-ultimate-guide/

Preis, T., Moat, H. S., & Stanley, H. E. (2013). Quantifying trading behavior in financial markets using Google Trends. *Scientific reports*, *3*, srep01684.

Ulrich, J. (2018). TTR: Technical Trading Rules. R package version 0.23-3. https://CRAN.R-project.org/package=TTR

Vaiz, J. S., & Ramaswami, M. Forecasting Stock Trend Using Technical Indicators with R.

Varian, H. R. (2014). Big data: New tricks for econometrics. *Journal of Economic Perspectives*, *28*(2), 3-28.

Weng, B. (2017). Application of machine learning techniques for stock market predictions. *Doctor of Philosophy Dissertation Auburn University*.

Appendix I -List of All Features

Close_Diff_SES	The 1st difference of the exponentially smoothed closing price.
RSI	Relative Strength Index
SMI	Stochastic Momentum Index
Stoch	Stochastic Oscillator
WPR	William %R
MACD	Moving Average Convergence Divergence
MACD_Sig	Moving Average Convergence Divergence Signal
OBV	On Balance Volume
Apple_Stock_RSI_Change	1st difference of RSI of search term "Apple Stock"
Apple_RSI_Change	1st difference of RSI of search term "Apple"
Apple_Stock_SMAY	Y(6,8,10,or 20) day simple moving average of search term "Apple Stock"
Apple_Stock_SMAY_Change	1st difference of Y(6,8,10,or 20) day simple moving average of search term "Apple Stock"
Apple_SMAY	Y(6,8,10, or 20) day simple moving average of search term "Apple"
Apple_SMAY_Change	1st difference of Y(6,8,10, or 20) day simple moving average of search term "Apple"
Apple_Stock_EMAY	Y(6,8,10,or 20) day exponential moving average of search term "Apple Stock"
Apple_Stock_EMAY_Change	1st difference Y(6,8,10,or 20) day exponential moving average of search term "Apple Stock"
Apple_EMAY	Y(6,8,10, or 20) day exponential moving average of search term "Apple"
Apple_EMAY_Change	1st difference Y(6,8,10, or 20) day exponential moving average of search term "Apple"
Apple_Stock_DisparityY	Ratio of "Apple Stock" search term volume to its Y day moving average
Apple_Stock_DisparityY_Change	1st difference of Ratio of "Apple Stock" search term volume to its Y day moving average
Apple_DisparityY	Ratio of "Apple" search term volume to its Y day moving average
Apple_DisparityY_Change	1st difference Ratio of "Apple" search term volume to its Y day moving average

Appendix II – Calculation of Accuracy, Precision, Recall, and Specificity

$$Accuracy = \frac{\mathrm{tp} + \mathrm{tn}}{tp + tn + fp + fn}$$

$$Precision = \frac{\mathrm{tp}}{tp + fp}$$

$$Recall = \frac{\mathrm{tp}}{tn + fp}$$

$$Specificity = \frac{\mathrm{tn}}{tn + fp}$$

Where,

tp = Number of true positives (prediction is 1 and actual value is 1)

tn = Number of true negatives (prediction is 0 and actual value is 0)

fp = Number of False positives (prediction is 1 and actual value is 0)

fn = Number of False negatives (prediction is 0 and actual value is 1)

Appendix III – Link to Code

The code for the Google Webtrends data collection Python script can be found here: https://github.com/jdglaser/mathesis/blob/master/GoogleSearchVolume.py

The R code for the paper can be found here:

https://github.com/jdglaser/Forecasting/tree/master/ForecastingProject

.