UDACITY MACHINE LEARNING ENGINEERING CAPSTONE PROJECT

QUALIFYING GOOGLE TREND's SEARCH TERMS' VOLUMES (STV) AS A SIGNIFICANT CONTRIBUTOR TO INCORPORATE IN FINANCIAL TRADING STATEGIES

REVISITING PREIS' GOOGLE TRENDS TRADING STRATEGY
WITH SUPERVISED AND RNN LEARNINGS BASED INSIGHTS

1. Definition

Project Overview

There are two schools of thought on whether collective decision-making precedes or is already incorporated into financial market pricing. One school of thought is well represented by an Economic Nobel Prize winning Eugene Fama. Mr. Fama is known as "The Father of Finance". Fama' University of Chicago PHD thesis concluded that short-term price movements are unpredictable and approximate a random walk. The futility of trying to gain advantages over Fama's efficient-market hypothesis was later widely reinforced by Princeton University economist Burton Malkiel's in his book "A Random Walk Down Wall Street" who posits: " A blindfolded monkey throwing darts at a newspaper's financial pages could select a portfolio that would do just as well as one carefully selected by experts". Per Fama, "the past history of a series of cannot be used to predict the future in any meaningful way and that the future path of a security's price is no more predictable than the path of a series of random numbers.

The other school of thought is well represented by Fama's nemesis Robert J. Shiller who famously won and shared winning his own Nobel Prize the same year with Fama. Shiller wrote a book entitled "Animal Spirits" about the impact of human psychology on the market and believes Fama incorrectly minimizes the role of investor psychology and emotion in financial markets. Also supporting the idea that financial markets are not information efficient is Herbert Simon, an Economic Nobel Prize and computer science Turing Prize winner. Simon posited that an actor begins their decision-making processes by attempting to gather information. Both economists inspire a good question: If you can measure the information gathering process before making a decision, can you use it successfully to predict short-term financial market movements?

In 2013, Tobais Preis (Warwich Business School), Helen Susannah Moat and Eugene Stanley (Boston University Physic Department) submitted a paper to www.nature.com entitled "Quantifying Trading Behavior in Financial Markets Using Google Trends" (https://www.nature.com/articles/srep01684). This paper posits that by using Google Trends within an investigated time period their "Google Trends Theory" or search-term volumes (STV) theory not only reflected the current state of the market but could have anticipated future trends. They summarize by saying, in retrospect (from 2004 to 2011), their STV could have been used in the construction of a profitable trading strategy.

This Capstone Project analyzes Preis' STV strategy, reviews its methodology, implement a supervised learning approach with Dow Jones Industrial Average (DJIA) 2004 to 2011 training data, update this research with DJIA 2012 to 2017 test data, and suggest how to further improve research efforts with RNN learning.

This Capstone's mission is to learn if the key financial terms (selected in Quantifying Trading Behavior in Financial Markets Using Google Trends) along with Google term search volumes are statistically proven predictors to financial market price movements.
This Capstone's null hypothesis is the market-efficiency hypothesis (Fama) is correct, information is already factored into financial market pricing, and accordingly key financial terms volumes are not statistically for financially valid predictors for future financial index price movements.

To test this hypothesis, Preis' top twenty financial terms will be tested on 2012 to 2017 financial indexes and its financial return results will be compared to random investing and long term buy and hold returns. To continue testing this null hypothesis, this capstone project compares the power of STV to the effectiveness of known financial trading leading, lagging, and breakout strategies.

In summary, the central question addressed is are the financial terms listed in the python dictionary below (combined with their Google search volumes) effective predictors for the closing prices for each of the financial indexes listed in the python dictionary below:

terms = ['debt', 'stocks', 'restaurant', 'economics', 'portfolio', 'dow jones', 'credit', 'revenue', 'inflation', 'housing', 'investment', 'return' , 'markets', 'unemployment', 'growth', 'hedge', 'money', 'derivatives', 'leverage']

indexes = ['DJIA_Close', 'NASDAQ_Close', 'SP500_Close', 'Russell_2000_Close']

Problem Statement

Simon's observation that "an actor begins their decision-making processes by attempting to gather information before making it" makes sense. People do engage information before acting upon it. *The problem is the difficulty of figuring out if you can capitalize on the chasm between investors engaging information and then their acting upon it in financial markets.*

Before getting into how this Google Trends Strategy (GTS) or Search-Term Volume (STV) strategy works, it is important to review the important questions being raised that underscore the importance of this capstone inquiry:

- Can you predict short-term stock movements (up/down) based upon search terms combined with its search-term volumes?
- Are notable drops in the market traceably preceded by investor concern and are notable increases in the market traceable preceded by investor confidence?
- Do the majority of trades implemented by machine learning algorithms today completely reinforce the efficient market hypothesis or is there a significant negation to be found in a time delay impact between those researching information and then acting on it?
- If you can predict short-term stock movements using sentiment search key words volumes:
    - Is the DJIA the best index for profit?
    - What is the relationship between search term volume and trading volume?
    - What are the best key terms to use for profit per index?
    - How do you determine key word efficiency?
    - Is there an optimal combination of key word, search volumes, and technical indicators?
    - What is the most profitable GTS/STV methodology for getting into and out of financial trades?
- Does RNN closing price analysis confirm or deny that search-term volumes can be profitable predictive indictors?

This capstone inquire answers if using a STV strategy a day or two before a week of financial trading and getting out of either the buy or short position later is more profitable and statistically sound in comparison to using a random or buy and hold investment strategy. This answer is provided by first replicating Preis' GST methodology and 2004 to 2011 STV results and then applying this methodology to the test data (2012 to 2017). Profitability and keyword statistically significant and standard deviations over random investments results between strategies are compared.

After reviewing these test results, this report then use supervised learning to compare how these key financial terms predict closing stock prices in comparison to other leading, lagging and breakout predictive stock strategies.

All results are analyzed for a final judgment on the null hypothesis that believes the market is efficient and all information is immediately incorporated into financial index market pricing. This research concludes with thoughts on how to use RNN analysis to move this research forward.

Metrics

Here are the metrics use to compare these STV, random and buy and hold strategies:

- Test data return ratio – If you invested $10,000 January 1st 2012 and the Test data return ration is 1.867126, then on November 17th 2017 the investment is worth $18,671.13

- Buy and Hold return ratio - If you invested $10,000 January 1st 2012 and the Buy and Hold ratio is 1.960, then on November 17th 2017 your investment is worth 19,600.

- Random return ratio – If you invested $10,000 January 1st 2012 and the Random return ratio is 1.093746, then on November 17th 2017 your investment is work $10,937.46.

Here are the metrics used to compare the relative effectiveness of the twenty financial terms:

- Standard Deviations from Normal Deviation – the standard deviation of the return

- Statistical Significance – the return minus the mean divided by the standard deviation

Here are the metrics used to measure the effectiveness of the models:

- Classifier Accuracy

- Final Model Accuracy

- Random Model Accuracy

II. Analysis

Data Exploration

The dataset used for this research inquiry has been assembled from three main data sources: Yahoo Finance, the search terms reported in Preis' report, and Google Trends. Here is an explanation on the data collected from each source:

The first data source is Yahoo finance (https://finance.yahoo.com). From Yahoo Finance, financial index closing prices and volumes have been collected starting from 01/01/04 until 11/17/2017.
These indexes include the Dow Jones Industrial Average (DJIA), NASDAQ, Standard and Poor's 500, and the Russell 2000. The reason for including this index information is to explore the predictive relationship Google Search Term volumes have with each index. The dictionaries below list the closing price and volume dataset's column names:

- index_closing = ['DJIA_Close', 'NASDAQ_Close', 'SP_Close', 'R2000_Close']

- index_volumes = ['DJIA_Volume', 'NASDAQ_Volume', 'SP_Volume', 'R2000_Volume']

Also collected from Yahoo Finance, is the stock closing price and volume within the DJIA that occurred within the same date timeframes (01/01/04 until 11/17/2017) listed above. DJIA individual stocks and volumes are included to explore term and volume's predictive measures.
The stock symbol name and stock_volume dictionaries below list the stocks closing price and volumes dataset's column's DOW stock symbols:

- stocks_closing = ['MMM_Close', 'AXP_Close', 'APPL_Close', 'BA_Close', 'CAT_Close', 'CVX_Close', 'CSCO_Close','KO_Close', 'DIS_Close', 'XOM_Close', 'GE_Close', 'GS_Close', 'HD_Close, 'IBM_Close',INTC_Close', 'JNJ_Close', 'JPM_Close', 'MCD_Close', 'MRK_Close', 'MSFT_Close', 'NKE_Close','PFE_Close', 'PG_Close', 'TRV_Close', 'UNH_Close', 'UTX_Close', 'VZ_Close', 'WMT_Close']

- stock_volumes = ['MMM_Volume', 'AXP_Volume', 'APPL_Volume', 'BA_Volume', 'CAT_Volume', 'CVX_Volume', 'CSCO_Volume','KO_Volume', 'DIS_Volume', 'XOM_Volume', 'GE_Volume', 'GS_Volume', 'HD_Volume', 'IBM_Volume', 'INTC_Volume', 'JNJ_Volume', 'JPM_Volume', 'MCD_Volume', 'MRK_Volume', 'MSFT_Volume', 'NKE_Volume', 'PFE_Volume','PG_Volume', 'TRV_Volume', 'UNH_Volume', 'UTX_Volume', 'VZ_Volume', 'WMT_Volume']

The second data source is Preis' research (https://www.nature.com/articles/srep01684).
Twenty of the top performing buy and short terms listed in Preis' Figures 3 and 4 are included in this research projects dataset along with the Google Trends Volume (GTV) normalized search volume values given for each term from 01/01/2004 to 11/17/2017. Note that these normalized values are provided on a weekly (not daily) basis. The terms dictionary below lists each top 20 performance term listed in the dataset's column names:

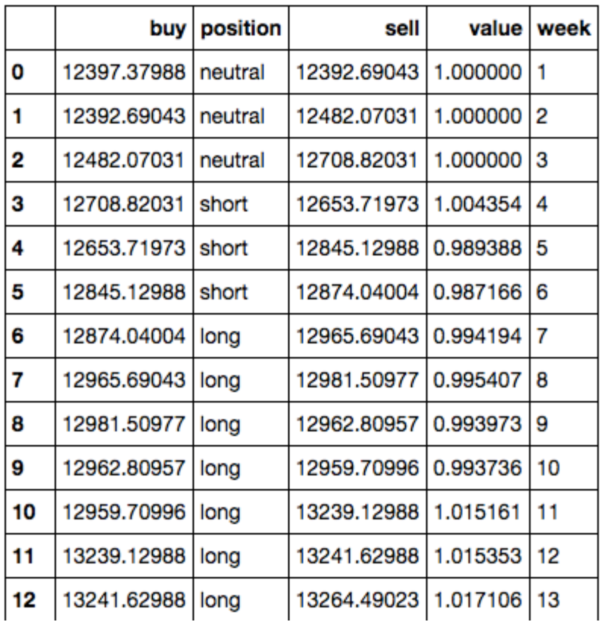
- terms = ['debt', 'stocks', 'restaurant', 'economics', 'portfolio', 'dow_jones', 'credit', 'revenue', 'inflation', 'housing', 'investment', 'return' , 'markets', 'unemployment', 'growth', 'hedge', 'money', 'derivatives', 'leverage']

The third data source is Google Trends. The volume of searches done between January 2004 and November 17th 2017 on each term above was recorded and added to the main dataset.

The dataset's training and testing data is divided by date. The training set contains all data between January 1st 2004 and December 31st, 2011. The test set contains all data between January 1st 2012 and November 31st 2017.

The dataset has 3,496 rows and 85 columns. To account for the missing daily search term volume data, the normalized Google search term volume value that is provide a day or two before a week of trading is placed on the same row as the first day of trading. To manage a missing data abnormality, sometimes the dataset is processed to use only the rows where the term volumes and close price are combined. In these cases, the dataset is reduced to 724 rows with 307 rows in the test data and 417 rows in the training data. Predictive algorithms that review trading volumes and closing prices use the full 3,496 rows for training and testing datasets. Predictive algorithms that review Google Search Term volume use reduced 724 rows for training and testing datasets.

To begin exploring the buy and sell signal behavior of each term and to implement the Google Trend Strategy, we use the relative_change_volume function to determine the three week volume average based buy and sell signals. This function is also used by the term_signal code to visually see the intensity of each signal as shown in the next section. After implementing the DoStrategy function (more on this later) we also see how relative_change_volume is used to trade. For example, below we see the first two trades using the term debt on the test data. First we see three weeks go by to get a three week average so there is no trading and the position is neutral. On the fourth week, we see the debt search term volume is higher than the past three weeks so we **short** by buying at 12708.82 and selling when the short signal ends three weeks later at 12874.04 that results in a loss of $165.22. The next trade is a buy at 12,874.04 and is sold six weeks later at 13,264.49 for a gain of $390.45. The Google Trend Strategy is now working as shown below:

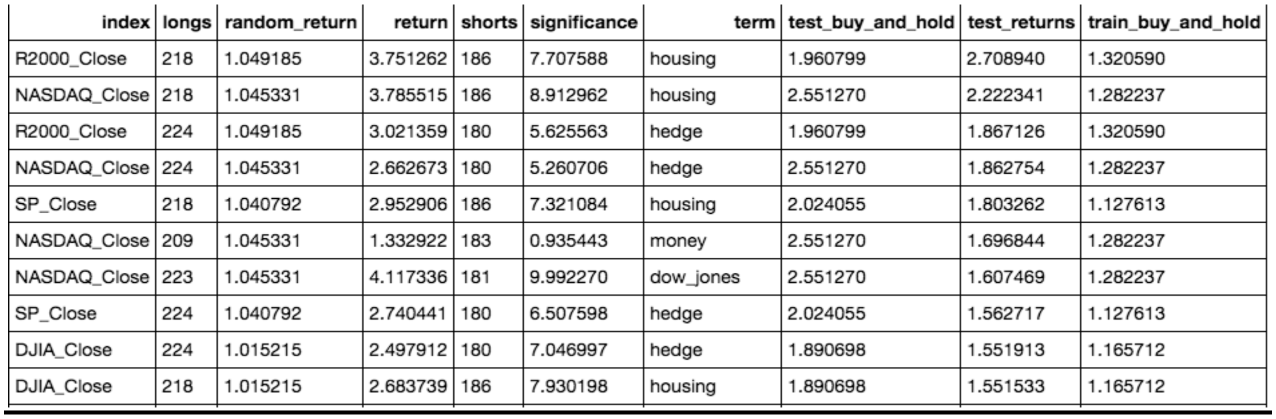


	buy	position	sell	value	week
0	12397.37988	neutral	12392.69043	1.000000	1
1	12392.69043	neutral	12482.07031	1.000000	2
2	12482.07031	neutral	12708.82031	1.000000	3
3	12708.82031	short	12653.71973	1.004354	4
4	12653.71973	short	12845.12988	0.989388	5
5	12845.12988	short	12874.04004	0.987166	6
6	12874.04004	long	12965.69043	0.994194	7
7	12965.69043	long	12981.50977	0.995407	8
8	12981.50977	long	12962.80957	0.993973	9
9	12962.80957	long	12959.70996	0.993736	10
10	12959.70996	long	13239.12988	1.015161	11
11	13239.12988	long	13241.62988	1.015353	12
12	13241.62988	long	13264.49023	1.017106	13

By implementing VST using Preis' highest performing term ("debt") we have confirmed his training data results by capturing the same 105 trades and the same highest return value of 3.274.

However, when we implement the VST strategy on the test data, we get the much lower return value of .802 that can indicate evidence of over fitting the training data. In fact, when we test the training data's top twenty performing terms on the test data, we get mixed results that raise question on the effectiveness of this VST strategy overall.

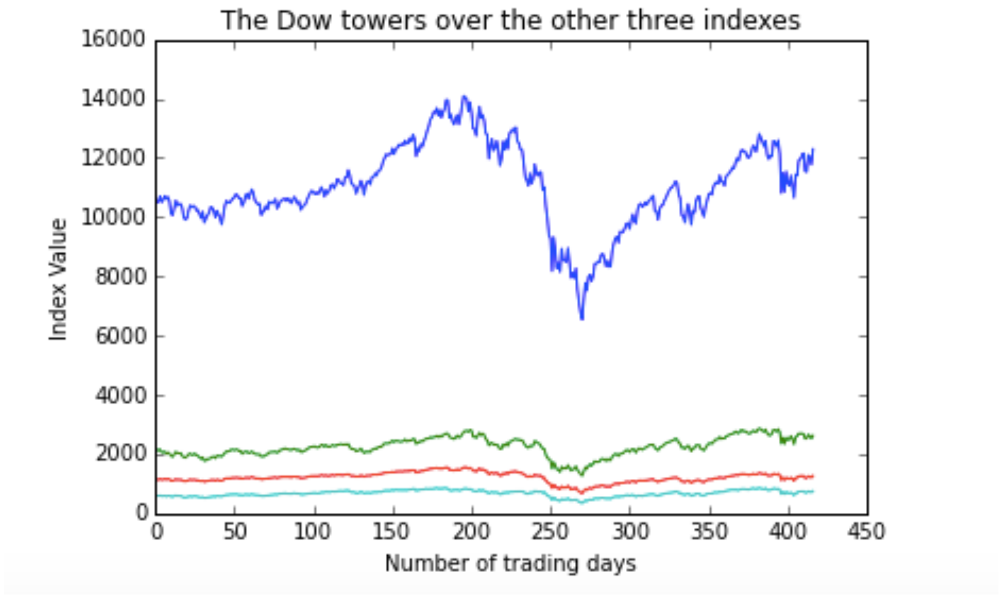
Below we see the top ten out of eighty returns from all twenty term on each of the four market indexes. Note the top test (**test_returns**) return value of 2.708 is the only one to beat a **test_buy_and_hold** strategy value of 1.96. The random strategy was always the lowest performer. The other seventy-nine STV results returned less than their corresponding test buy and hold results despite not including all STV transaction costs as well. This is strong support for Buy and Hold long term being the best strategy.



index	longs	random_return	return	shorts	significance	term	test_buy_and_hold	test_returns	train_buy_and_hold
R2000_Close	218	1.049185	3.751262	186	7.707588	housing	1.960799	2.708940	1.320590
NASDAQ_Close	218	1.045331	3.785515	186	8.912962	housing	2.551270	2.222341	1.282237
R2000_Close	224	1.049185	3.021359	180	5.625563	hedge	1.960799	1.867126	1.320590
NASDAQ_Close	224	1.045331	2.662673	180	5.260706	hedge	2.551270	1.862754	1.282237
SP_Close	218	1.040792	2.952906	186	7.321084	housing	2.024055	1.803262	1.127613
NASDAQ_Close	209	1.045331	1.332922	183	0.935443	money	2.551270	1.696844	1.282237
NASDAQ_Close	223	1.045331	4.117336	181	9.992270	dow_jones	2.551270	1.607469	1.282237
SP_Close	224	1.040792	2.740441	180	6.507598	hedge	2.024055	1.562717	1.127613
DJIA_Close	224	1.015215	2.497912	180	7.046997	hedge	1.890698	1.551913	1.165712
DJIA_Close	218	1.015215	2.683739	186	7.930198	housing	1.890698	1.551533	1.165712

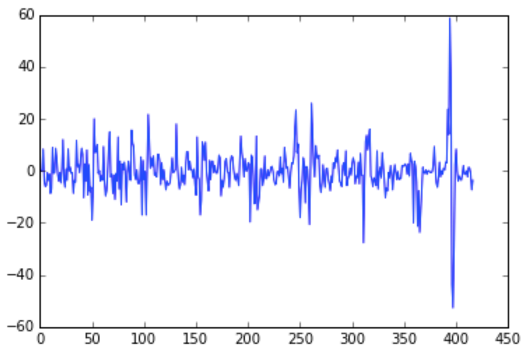
Exploratory Visualization

The business profit potential of learning how to predict DJIA price movements particularly shorting the indexes on a 30% drop becomes clearer when visualize how much more the DJIA has room to fall:

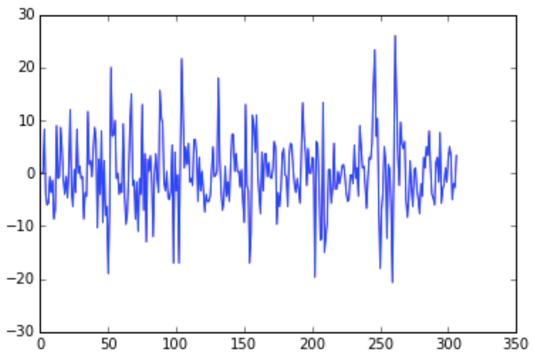


Below we compare signal strengths between the training and test data. We see the buy and short signals are actually stronger on the test data then the training data despite the testing data's poorer returns performance. Therefore, strong signals to buy and short do no equate to more profits:

Training data signals on "debt" term



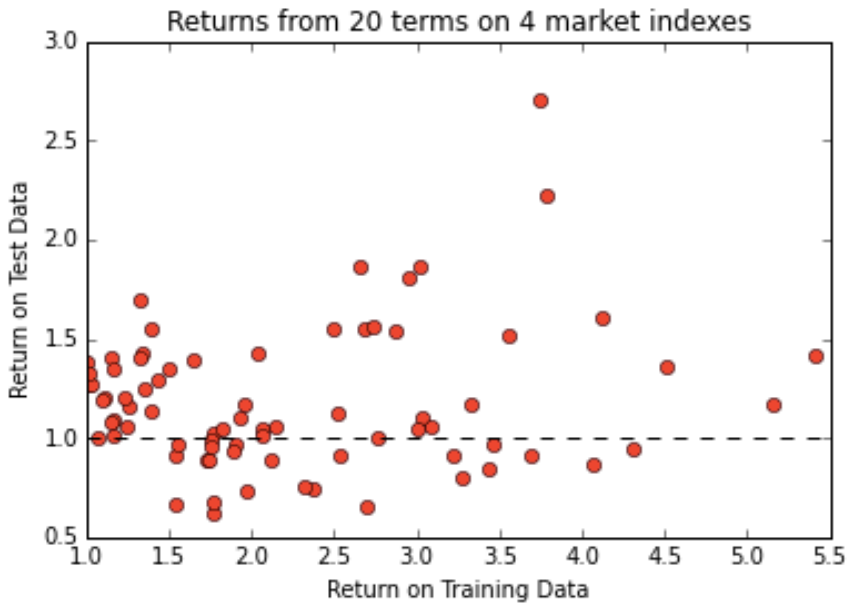
Test data signals on "debt" term



Below when we review the results of each of the twenty terms on each index (DOW, NASDAQ, SP500, and Russell 5000) we see that the returns on the training data (x-axis) are higher than the returns on the test data (y-axis).

We also see the risk of not using the right term with the results below that are below the 1.0 test data line.

The visualization below clearly counters the belief that you can use these twenty terms to profitably predict future market trends without much risk.



Replicating Preis' research methodology and applying it towards new test data gives preliminary evidence that there is merit to the market efficiency null hypothesis.

Yet, this data exploration has not answered some previously raised important questions.

- Why are some terms are effective during the training period but not during the test period?
- Can the STV strategy be a successful predictor when applied with other trading technical indicators?
- What can we learn about using the STV strategy from supervised and RNN learning?

This capstone inquiry continues to investigate the role that STV strategies can potentially play predicting financial index price movements because these questions remain unanswered and because it is likely that a weakness in Preis' study is that the context is limited by focusing only on the relationship between STV and index price movements. Perhaps a broader context will clarify the strong contributing role that STVs play in market prediction? It makes sense that to figure out if you can capitalize on a chasm between investors engaging information and then acting on it you need a broader context.

Another tact on testing our null hypothesis is to use supervised learning algorithms and techniques to investigate the relationships between financial index price movements, search terms, search volumes, and other technical indicators used to predict index price movements. This capstone introduces three categories of technical indicators: a leading indicator (relative strength) that considers momentum, a breakout indictor that considers capturing breakout price movements, and a lagging indictor (moving averages) that accounts for current trends.

Another advantage with this tact is we can use a larger dataset that includes all trading days between 2004 and 2011 by reloading the original dataset without removing the search term volume missing data rows previously eliminated and add the weekly search term volume normative value for each day since this is the value used.

By measuring model efficiencies and feature importance on each key term in predictive contribution comparison to each of these leading, breakout and lagging indicators we have more information to make judgment on the merit of our null hypothesis.

Algorithms and Techniques

This section introduces the algorithms and techniques used within the methodology to provide insightful accuracy ratings on our model.

It is important to note that each trading strategy (moving average, relative strength, and breakout) each has a 15 trading day cycle to make the comparison with the STV strategy comparable. STV uses the prior three weeks to decide to buy, short or do nothing. Moving average is based on a 15-day average. The range breakout strategy is based on a 15-day range. Relative strength is calculated using the price movements that have gone up and down over 15 days.

Search-Term Volume (STV) ~ def relative_change_volume

Google's Search Term values are normalized from 1 to 100. Preis' STV strategy is to compare this week's term volume to the three weeks before. If the volume is higher or lower than the three weeks before it is a signal to taking an action in the market. The action is either buy, buy a short, sell or end the short.

Moving Average ~ class MovingAvg:

Moving Average is commonly used trending tool for financial traders. A moving average is calculated by adding up the prices for the most recent (n) days and then dividing by the sum total by n. The number of days (n) determines how sensitive it is.

When prices stay within a range, moving averages oscillate sideways while in a trend trading market-moving averages tend to move in a definite upward or downward direction. When the market price is lower than the moving average the market is bearish and the trading action is to sell. When the market price is higher than the moving average the market is bullish and the trading action is to buy.

This capstone implements a 15 moving average to gauge its sensitivity and effectiveness.

Relative Strength Index ~ Compute new features

Relative Strength is the average number of days that the market closes up divided by the average number of days (15 in this capstone) that the market closes down. It is simply an index of RS and shows overbought or oversold market conditions. If the RSI is below 30, the market is considered oversold and this is considered a signal to buy. If the RSI is above 70, the market is considered overbought and near a possible top. The trading technique is to sell.

This RS algorithm counts or equals (=) the sum of days up over the sum of days down over the total number of days.

The RS index shows either overbought or oversold using this question: RSI = 100- [100/(1 + RS)].

Range Breakout ~ Compute new features

A range breakout strategy believes that when a breakout out of range occurs it pays to capitalize on the direction of the breakout. So, if the price is the highest than it has been in the last n (fifteen) days, then buy and conversely if the price is the lowest it has been in fifteen days than sell. This strategy is the opposite of the moving average strategy which is why it is included in this study.

Term-Signal ~ def Get_Signal

Term-Signal reads the STV normalized value signal for every week it is available and uses this signal all week until a new STV normalized signal is available.

Action Label ~ def Get Label

The Action Label is a very important to the success of capstone. A rate of daily price changed is introduced that looks at the daily change of the market index price and determines to buy if the price change goes up enough, short if the price change goes down enough and to do nothing if the price is within a daily price change range. When the rate of change was 1% there were very few trades and .05% had too much activity so the rate of chance selected was right in the middle at .0075%

Prediction DataFrame per date:

A prediction dataframe was created to store the signal value for each term and its corresponding label. So for each date we have either a do nothing label with a 0 signal, a buy label with a 1 signal or a short label with a -1 signal.

This prediction table also has: the 15 day moving average value for the DJIA (JDIA_Close_15), daily change in the DJIA (DJIA_Close_day_change), daily close breakout signal (DJIA_Close_breakout), and daily Relative Strength value (DJIA_Close_RSI).

All of these values are used to make stock index price movement indexes.

Predictive Algorithms Pipeline

Since training (2,500) and testing (994) data sets are relatively small, supervised learning algorithms that excel with small datasets were selected. This is an important reason these datasets were processed by a Gradient Boosting, Random Forest, and SGD Classifiers.

The Gradient Boosting Classifier is selected because of its robustness (infamous Netflix competition winner that combined over 500 models) and very effective predictive algorithms with data samples less than 100K and when text has been converted to numeric data.

The Random Forest Classifier is selected because they are fast, flexible, and perform well even in the presence of a large number of features and small number of observations. Like GBCs, Random Forest Classifiers excel with data samples less than 100K and when text has been converted to numeric data.

Stochastic Gradient Descent is selected because of its efficiency given its ease to implement particularly to address natural language processing and text classification problems.

Benchmark

The DJIA Gradient Boosting Classifier accuracy rating of .757 sets the benchmark.

The DJIA Gradient Boosting Classifier accuracy result was in the middle between an extremely over fit Random Forest Classifier (training set @ .98) and an ineffective SGD Classifier with a training accuracy of .55 and a lower testing accuracy of .53. In contrast, there is a slight improvement between training and testing with Gradient Boosting.

III. Methodology

Data Preprocessing

One of the biggest challenges with exploring the Google Trends data from 2004 to 2017 is that the normalized weekly STV data consists of a very small dataset. The initial data explorations previously described how this research replicated Preis' 2004 to 2011 results and applied his methods to 2012 to 2017 test data using his top performing twenty terms on four market indexes instead of one to discover only one instance out of eighty in which a STV strategy beat a buy and hold strategy. While this is pretty convincing evidence that the STV strategy isn't a predictive solution by itself and supports the null hypothesis, perhaps working with a larger dataset and comparing STV features and performance to established trading techniques will provide more insights on how to capitalize using a STV strategy and counter the null hypothesis.

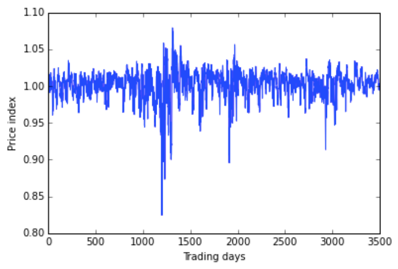
With this in mind, this research created a new predication data frame (prediction_df) with data for each day instead of each week to enlarge the dataset.

Then a new features function created filled in the daily (from 2004 to November 17th 2017) three new features moving average, breakout strategy, and relative strength. The change in the daily DJIA close price was record for each day. The daily DJIA 15 day moving average value was recorded. The daily recording of when a DJIA moving price range breakout occurs and whether it is a buy or short signal was captured. The DJIA relative strength value was capture for each day.

A get signal function captures the weekly STV signal and then applied it to all the days of the weeks. Each term on each day now has either a do nothing, buy or short signal for each term.

Thus far the twenty terms data is based upon normalized Google Trends Search terms volume values. DJIA relative strength (RS) values are normalized. The data preprocessing work did not normalize the breakout or moving average data because it is important to capture the outliner values and their signals.

Below we see the moving average signals over all daily DJIA price movements noting again that we want to include these averaged outlining price movements:



Implementation

Here is the implementation methodology used within this capstone project:

1) Load the dataset with terms, indexes, and index volumes.

First step is to load the dataset that has the top twenty normalized search term volumes, the closing price history for the four indexes and corresponding four indexes trading volumes.

2) Implement the 3 week (15 days) Google Trends and Search Term Volume Strategy.

   The next step is to active the fifteen-day search term volume logic function that will be called in a future step.

3) Define the Moving Average Logic

   The next step is to active the technical trend indicator or moving average function to be called in future steps.

4) Compute the new technical indicators features (moving average, breakout, and relative strength) and create predictions.df

   The next step is to create the predictions data frame and fill it with computed values for three technical indicators (moving averages, breakouts, and relative strength).

5) Create weekly term signals with Get_Signal function

   The next step is to activate the Get_Signal function that captures the Google Trends normalized weekly search term volume value and to fill in this value for each day within the trading week.

6) Fill in daily normalized weekly search term volume value by calling Get_Signal function

   The next step is to call the Get_Signal and fill in the missing daily search term volume data.

7) Create a Buy, Short, or do nothing label values based on Closing Price value change

   The next step is to define a daily change in closing price between today's closing price and the closing price the day before and use it to determine one of three classification actions (Buy, Short, or do nothing) to take.

8) Review features and label before running the model

   The next step is to review all the features and the label that will be used in the upcoming supervised learning pipeline.

9) Create a training and predicting pipeline

   The next step is to create the training and predicting pipeline that sets the accuracy foundation for three selected supervised learning algorithms.

10) Create training and testing datasets

    The next step is to create the training and testing datasets. In this scenario, 2500 data are put in a training data set and 994 data are put in a testing data set.

11) Implement three supervised learning algorithms

    The next step is to fill the pipeline with supervised learning algorithms and run them to see which algorithm helps deliver the best model. In this case, the supervised learning algorithms imported are LinearSVC, SVC, Logistic Regression, SGD, Random Forest, AdaBoost, Gradient Boosting and Decision Tree Classifiers. The three algorithms selected mostly for there skills with small datasets are Gradient

Boosting, Decision Tree and SGD Classifiers. Gradient Boosting was always the best performing supervised learning algorithm on every index.

12) Rank feature importance

    The next step is to rank features by their importance per each index.

13) Refine model with top four features

    The next step is to reduce the model to include only the most important features. On every index, terms were always a distance fourth compared to the breakout, relative strength and moving average indicators values. To evaluate the impact of the most important term, a minimum of four features are included in the refined model. Improvements in the model are noted in comparison with the original Gradient Boosting Classifier's testing data accuracy score.

14) Compare top four features and supervised learning results with doing nothing

    The second to last step is to compare the effectiveness of the refined model (four essential features) do the accuracy score of doing nothing (random strategy).

15) Account for accuracy by comparing the actions taken with the correct actions

    The final step is to compare and count each action take with each action that should have been taken.

The entire methodology and algorithms used are documented for all four indexes.

Refinement

After assessing the Gradient Boosting Classifier model accuracy for each of the four indexes, a featuring ranking was done to access the influence of each feature on the model and the power of the top twenty terms as an important component within a financial trading strategy. To create a more effective and refined model, the top four features were selected to create a new model and then record its accuracy.

For the DJIA, the top-five feature importance ranking is:

1. 15 Day Moving Average is .63%
2. Relative Strength Index is .093%
3. DJIA Breakout is .04%
4. Investment Term is .026%
5. Revenue Term is .018%

The top four terms were used in a new model and its accuracy improved very slightly over the Gradient Boosting Classifier from .757 to .761.

For the NASDAQ, the top-five feature importance ranking is:

1. 15 Day Moving Average is .60%
2. Relative Strength Index is .077%
3. Breakout is .041%
4. Investment Term is .032%
5. Revenue Term is .02%

The top four terms were used in a new model and its accuracy improved very slightly over the Gradient Boosting Classifier from .669 to .678 .

For the S&P500, the top-five feature importance ranking is:

1. 15 Day Moving Average is .59%
2. Relative Strength Index is .096%
3. Breakout is .037%
4. Markets Term is .023%
5. Stocks Term is .023%

The top four terms were used in a new model and its accuracy did not improve over the Gradient Boosting Classifier so the accuracy slightly decreased from .754 to .752.

For the R2000, the top-five feature importance ranking is:

1. 15 Day Moving Average is .60%
2. Relative Strength Index is .081%
3. Inflation Term is .026%
4. Breakout is .025%
5. Debt Term is .023%

The top four terms were used in a new model and its accuracy improved very slightly over the Gradient Boosting Classifier from 0.564 to .582.

As we will see in the next section, these very modest improvements are quickly put in their place by random chance. It is also important to note that adding terms as features didn't change the results above.

IV. Results

Model Evaluation and Validation

The tell tale sign on all of these models is how little they vary from the results realized if a trader didn't buy or short anything. In other words, doing nothing or randomly following the market as if it were completely efficient yields slightly better results.

When we compare the model's accuracy results from each indexes first gradient boosting classifier accuracy rates, and from the four-featured refined models to not trading at all, we see no reason to believe their was a predictive information gain made before random market efficiencies. Here are the accuracy rates:

	DJIA	NASDAQ	S&P500	R2000
Gradient Boosting Classifier	0.757	0.669	0.754	0.564
Refined Model with 4 Top Features	0.761	0.678	0.751	0.582
Do nothing ~ Random Strategy	0.76	0.68	0.75	0.585

These final models are reasonably aligned with our null hypothesis and market efficiency expectations. With each index, STV term feature importance is significantly lower than the other three breakout, relative strength, and moving average strategies. The highest percentage realized by any term is .032% with the search term "investment" on the NASDAQ market index.

It is important to note that each index is a diverse composite made up of many stocks. The DOW is an evolving composite encompassing 28 to 30 of the largest US companies. The NASDAQ is the largest composite index representing over 3000 companies of differing sizes. The S&P 500 represents 500 of the largest corporations. The Russell 2000 represents 2000 mid to small capitalization stocks. Yet despite this diversity, this model reaches almost identical results on each index.

With that said there are good reasons to trust this model. The Gradient Boosting Classifier training and testing accuracy results show a healthy transition accuracy rate transition from training to testing. Gradient Boosting accuracy has consistently lead on each index. Feature importance rankings have the same ranking on each index. The results above support early data explorations observations.

Yet, despite the models ability to generalize well on four different market indexes, this is not a robust model. This model's development has always been limited by the dependency on weekly Google Trend term volume data that started to be available January 1st 2004. While transforming the data from weekly to daily is a step in the right direction, the restricted dataset of less than four thousand trading days has been an obstacle.

It is easier to see the fragility within this model by studying the actions that were taken compared to the actions that should have been taken. For example, the DJIA benchmarked model did the following:

Correct Action	Action taken	Comments
4 Buys	4 Buys	4 Correct buys is a small number
123 Buys	123 ~ no action	Here are 123 missed opportunities!
3 Shorts	3 Buys	Here are 3 trades in the wrong direction!
1 Short	1 Short	Only 1 short was correct
107 Shorts	107 ~ no action	Here are 107 shorts missed
3 Do nothing	3 Buys	Here are 3 minor expenses
2 Do nothing	2 Shorts	Here are 2 minor expenses
751 Do nothing	751 Do nothing	Practicing a random strategy

This model missed 230 buy or short opportunities, got 5 buy or shorts right, got five buy or shorts wrong and was correct 751 times by doing nothing. Another way of looking at this is:

The DJIA Wrong/Right ration is 235/756. This DJIA model result was incorrect 235/991 or 23.7% of the time. This is deceiving because the main reason the model was correct was by doing nothing. Its five correct actions from other than doing nothing where nulled out by its five incorrect actions.

Here are similar results from the NASDAQ index:

Correct Action	Action taken	Comments
4 Buys	4 Buys	4 Correct buys is a small number
1 Buy	1 Short	Here is a trade in the wrong direction.
167 Buys	167 ~ no action	Here are 167 missed opportunities!
4 Shorts	4 Buys	Here are 4 trades in the wrong direction!
142 Shorts	142 ~ no action	Here are 142 shorts missed
6 Do nothing	6 Buys	Here are 6 minor expenses
670 Do nothing	670 Do nothing	Practicing a random strategy

The NASDAQ Wrong/Right ratio is 324/ 674. The NASDAQ model result was incorrect .32% of the time. Again, this is deceiving when you consider that 670 out of 674 correct actions were correct because no action was taken.

Another realization on the inefficiency of the STV strategy is when you compare term feature importance to the range breakout strategy. The range breakout strategy did nothing. It never bought or shorted. Yet, from a feature importance perspective it ranked higher than the highest feature importance ranked term on three out of four indexes and index exception (Russell 2000) was a tie.

DJIA	NASDAQ	S&P500	Russell 2000
Breakout ~.042	Breakout ~ .041	Breakout ~.037	Inflation term ~ .025
Investment term ~ .026	Stocks term ~ .032	Markets term ~ .023	Breakout ~ .025

Justification

By comparing the Gradient Boosting Classifier accuracy values to the revised model accuracy values and then the no action random strategy accuracy values, we see the true ineffectiveness of using the STV strategy to gain a competitive trading advantage. There is literally no difference between the revised term model and a random strategy:

	DJIA	NASDAQ	S&P500	R2000
Refined Model with 4 Top Features	0.761	0.678	0.751	0.582
Do nothing ~ Random Strategy	0.76	0.68	0.75	0.585

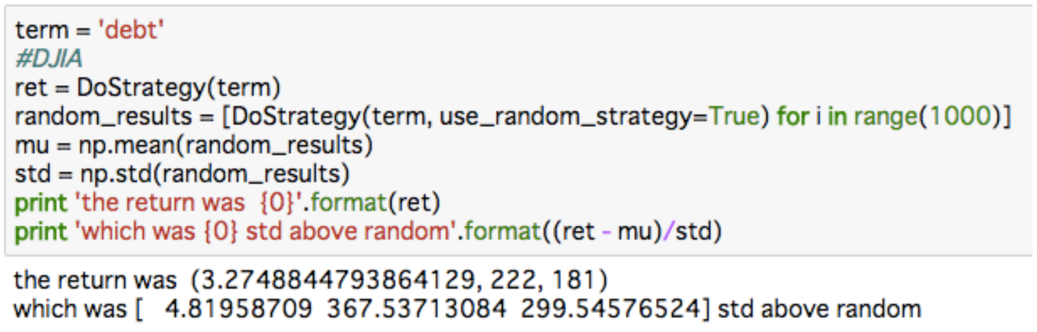
In the preliminary data explorations we examined the financial returns from a buy and hold strategy, a random investment strategy, and a STV strategy.. The important point is that only one STV term did better (didn't include transaction costs) than a buy and hold strategy out of eighty.

The STV strategy doesn't provide any edge to financial technical indicators when trading and is not profitable compared to a buy and hold strategy 99% (1/80) of the time.

V. Conclusion

This capstone project started by drawing the contrast between Nobel Prize winners who believe the market is efficient and those who believe there is a predictive opportunity gap between getting information before making a market trading decision. One of the most convincing arguments made supporting this opportunity gap was there is an opportunity to use Google Trends and use the meaning of the term with its volume changes to make buy and hold beating profits over the long term. The logic of high volume searches on the term "debt" means concern thus is a short the market opportunity and low volume searches with "debt" means low concern thus a good time to buy long. This logic makes sense and for that reason we investigate.

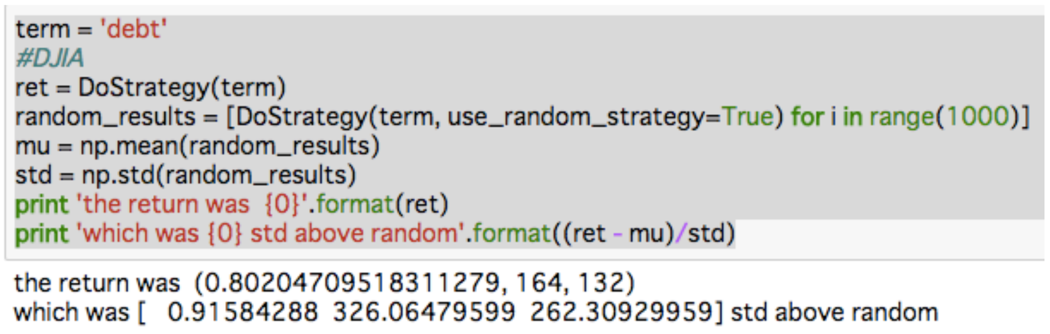
Debt training-data results: Return ratio = 3.27 | Standard deviation above random investments = 4.82



```
term = 'debt'
#DJIA
ret = DoStrategy(term)
random_results = [DoStrategy(term, use_random_strategy=True) for i in range(1000)]
mu = np.mean(random_results)
std = np.std(random_results)
print 'the return was  {0}'.format(ret)
print 'which was {0} std above random'.format((ret - mu)/std)
```

```
the return was  (3.2748844793864129, 222, 181)
which was [   4.81958709  367.53713084  299.54576524] std above random
```

Debt test-data results: Return ratio = .80 | Standard deviation above random investments = .91

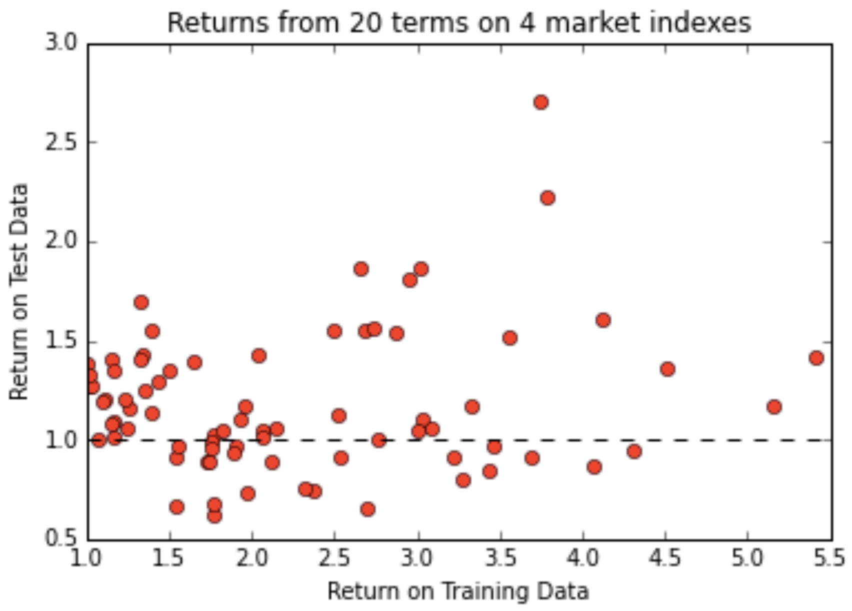


```
term = 'debt'
#DJIA
ret = DoStrategy(term)
random_results = [DoStrategy(term, use_random_strategy=True) for i in range(1000)]
mu = np.mean(random_results)
std = np.std(random_results)
print 'the return was  {0}'.format(ret)
print 'which was {0} std above random'.format((ret - mu)/std)
```

```
the return was  (0.80204709518311279, 164, 132)
which was [   0.91584288  326.06479599  262.30929959] std above random
```

Every search term return, standard deviation, trading signal, and returns reported by Preis' research was replicated.   However we quickly learned that using the term "debt" from 2012 to 2017 performed poorly and that the correlative relationship between using a VST strategy to outperform buy and hold financial gains doesn't exist.

In fact when we compared VST strategy training and test results we see financial returns hover around 1.5 and those returns over 1.5 are quickly more than countered by the multiple returns under the line or under 1.5 as seen below:



Notice how much larger the training axis is than the testing axis. The testing data has to account for failures.

The failure of implementing any VST strategy against buy and hold strategies was itemized in detail when ranking the financial results of twenty terms on four market indexes and only one out of eighty terms (investments) beat an index (Russell 2000) buy and hold strategy from 2012 to November 17th 2017.

However, the failure of using a VST strategy compared to buy and holds becomes painfully clear when you visually scan the return ratios of the entire eighty columns and notice how difficult it is to find a return below 1.8. Compare that to how few of the test data red circles above are higher than 1.8.

Expectations have been reset and the market efficiency null hypothesis that we seek to test appears soundly true. However, we want to understand the role a STV strategy has in the financial traders toolbox along with other technical indicators so we compare STV to a standard leading (relative strength), lagging (moving averages) and momentum (range breakout) indicators looking for a contribution that will help beat out random market efficiencies.

We didn't find any evidence of STVs improving market opportunities over a random market. Buy and hold strategies far outperform STV strategies and random investment strategies. All twenty terms ranked as less important a feature to a breakout strategy that didn't implement one trade. After calculating a Gradient Boosting Classifier accuracy rating for the model per each market index, refining the model using the most important term features didn't improve performance at all. Finally the four market indexes Gradient Boosting Classifier accuracy values and their following revised model with important financial term features both did poorer than simply doing nothing.

All of this supports the null hypothesis, that the market is efficient and that STV strategies do not take advantage of a gap between looking for information to make financial trades and making them. One is much better off by buying and holding index funds.

Reflection

One of the more interesting questions raised doing this capstone is: how does one know a term is relevant and when it stops becoming relevant? Whether the return relationship with debt in Preis' research was spurious or not is a topic for discussion, there is no doubt that the term "debt" has some traction with financial returns during 2004 to 2011. However, it appears once the housing and financial crisis was over and we entered into the 2012 to 2017 bull market, the term "debt clearly doesn't have the same financial relevance.

This question gets more interesting when we notice that different terms rank higher in different indexes? Why? What is it about their meaning that makes them the highest-ranking feature compared to others? Why does investments mean more in the Russell 2000 then stocks in the NASDAQ or markets in the S&P 500? If we use "investments" in our STV strategy, how do we know when this is relevant and when does its relevancy dissolve. Perhaps have hundreds of words being continually tested and what their feature importance movements is the way to go.

Another area of interest is gaining a better understanding of the relationship between momentum, leading and lagging technical trading tools and market returns. For example after learning that the breakout of range algorithms didn't work, it would have been good to have a plethora of trading strategies to replace it. Perhaps a volatility breakout strategy would have been effective. Perhaps other momentum strategies like on balance volume (OBV), directional movement (DM) or parabolic stop and reverse (SAR) would have been much better. Perhaps using a lagging indicator like a fifteen-day moving average is too correlated with the market indexes and have played a role in model inefficiencies rather than being the most important feature.

One cannot help but wonder what types of budgets and models the largest investment firms are using and what it would take to get a competitive advantage over them and if this is possible at all?

Since there doesn't appear to be possible to use STVs for short-term gain, perhaps next research project hypothesis asks: How to use STVs and technical indicators to support long-term buy and sell decisions?

Personally, this research capstone has been satisfying because it puts Preis' Google Trends Strategy in context and made me value this process.

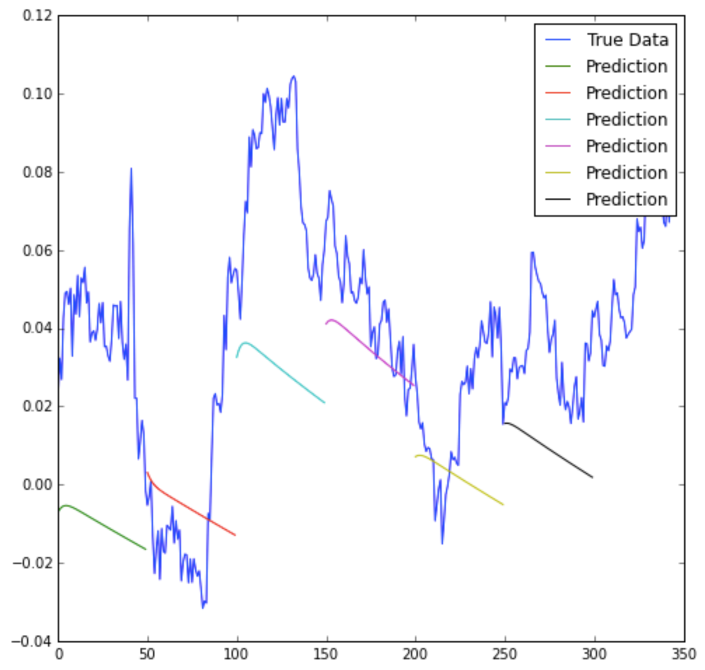
Improvement

One idea (algorithm and technique) I tried to incorporate into this study but struggled with was using Siral Raval's RNN LSTM code to predict a fifteen-day trends and using this as another technical indicator. Siral Raval is a brilliant and engaging Udacity instructor who shows how he uses keras with a tensorflow backend to program a fifty-day S&P 500 predictor. Here is the link where he explains his approach and provides his code that I have used: https://www.youtube.com/watch?v=ftMq5ps503w

His program predicts price movements by using a RNN to remember and incorporate market indexes' closing price in time-series patterns. Below you see that I've used his code and applied it to my DJIA data:

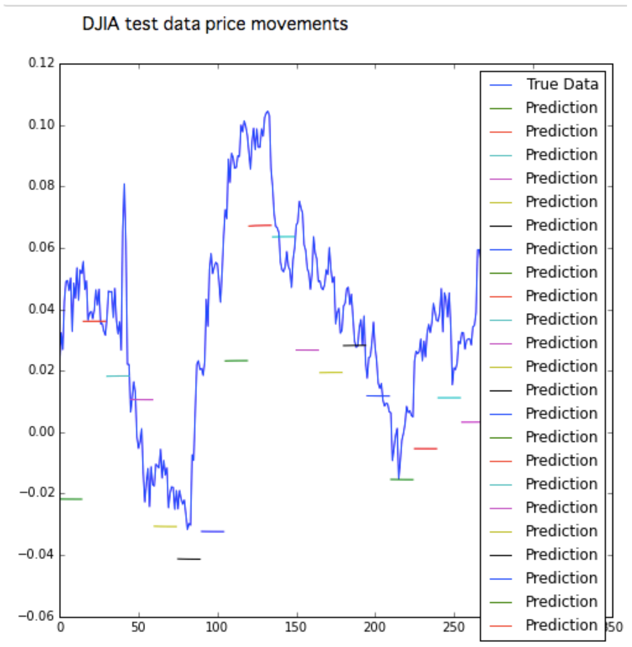
While the predict lines don't line up well in this visual, I was encouraged because I see value in focusing on the first fifteen days and comparing RNN forecasts to the other tools discussed. There appears to be an interesting early trend identifications being made.

Fifty-day RNN forecasts



When I tried to apply this to daily or fifteen-day forecasts, I struggled with Raval's code to get any shape and precision in the prediction lines especially for other than 50 days.

Fifteen-day RNN forecasts



FINAL REMARKS

As previously stated, the next revision of this research project would:

- Create many more leading, lagging and momentum indicators to train and test for results.
- Tailor Siral Raval's RNN code to produce fifteen, five and one day market index price predictions and compare with meaningful trading indicators.

This capstone is an explicit confirmation supporting its market efficiency hypothesis.