Predicting stock price changes by using NLP and Machine Learning on Google News Data

Michael Jasper

December 17, 2015

Abstract

Textual data is gathered from Google News using a web scraper (querying the top 10 holdings of the S&P 500) as well as historical stock data for these companies. A feature set is extracted from this corpus and a neural network is trained to create a model between the language features and daily stock price changes. Using this model, we can predict with moderate accuracy the stock price changes based on current news data.

1 Introduction

There are many[2] learning methods for predicting changes in stock market prices, including analysis of quantifiable data[5] and classification and sentiment analysis of news articles[10], press releases, and even Twitter posts[3]. The goal of this project is to perform natural language processing on news article headlines and summaries gathered from Google News by extracting features vectors from that corpus and historical outcome data based on the change in individual stock prices, and to train a neural network to predict future changes based on that data.

The desired outcome is a model which will be able to take current Google News headlines and summaries and predict a future change in price of the stock. The scope of the training data is limited to the Top 10 holdings of the S&P 500 from January-November 2015. The technologies used in this project are Node.js, NodeNatural (an open-source library for natural language processing)[4], Python, and PyBrain[9] (an open-source package for Machine Learning).

2 Problem Description

The question that this project seeks to answer is: Can a model be created which will predict future stock price change direction, using English language news and corresponding historical stock data? In addition to answering this question, a secondary challenge was creating or acquiring a suitable data set.

3 Approach & Methods

3.1 Data set creation

Upon a thorough search, no suitable data set was found which would meet the conditions for this model. The requirements for the data set are: it contained current textual news data from a variety of news sources about stock holdings of the S&P 500, along with stock price information for the corresponding dates. Although only ancillary to the research question, a somewhat novel approach was required to build the required data set. This method is described here.

The Google News service was deemed an adequate provider of news data from a variety of sources, as well as enabling querying news data by data and search term. Although it is not stated in the terms of use, Google discourages scraping the news data by "cutting off" repeated programmatic requests.

By using the following method, news data was obtained to build a data set:

- Randomly selecting a User Agent String (UAS) from a set of the 10 most popular UAS's and sending that value as a request header.
- Making requests serially, with a wait-time of 2500 milliseconds, plus or minus a random value between 0 and 1000 milliseconds, between requests.
- Alternating IP addresses every 100 requests.

Using these three methods, a data set of news data from January-November 2015 was created.

Historical stock data was obtained using the Yahoo! Finance[1] historical data web service, and associated with its related textual news data.

Listing 1: Data Point Example

```
{
   "bodies" : [ "No companies have done this better over the past decade than
       JPMorgan Chase and Wells Fargo ..." ],
   "date" : "7/30/2015",
   "headlines" : [ "Buy Bank Stocks Like a Boss", "Wells Fargo to Withdraw from
       Mortgage Marketing..." ],
   "symbol" : "WFC"
 }
```

3.2 About the data set

- Corpus consists of 2352 entries (news data per stock per day with price change).
- Corpus contains 740651 words.

3.3 Processing of Textual Data

Using the Bag-of-Words technique[7], the text data was transformed into a vectorized representation, suitable for ingestion into a learning algorithm.

First, a feature set was created using the following procedure:

1. A set of distinct words is extracted from the corpus.

2. This set is normalized for spelling and tense using the Stemming feature of NaturalNode[4] (a Node.JS natural language processing library).

3. Stop words (common words such as “and” or “the”), as well as stock specific words (such as “iphone” or “drilling”) are removed from the set.

4. A dictionary is created of word and word-counts.

5. The median 80% of words are selected as features for the Bag-of-Words representation.

Individual data points (news articles) are then compared against this feature set to create a vectorized representation of the data point.

Listing 2: Vectorized Data Example

```
{
    "input":[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,
        0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
        0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
        0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0,
        0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0,
        1, 0, 1, 1, 0, 1, 1, 1],
   "output":{"direction":1}
}
```

3.4 Neural network model creation

The data set of vectorized news data and stock price changes is split into two groups: training and testing data. Two methods were were used to divide the data set: The first method was to iterate through the data set, and randomly place a certain percentage of the data rows in a training array or testing array. The second method explored was to select a date that represented a mid-point of the data, and place data points prior to that date in the training set, and data points after that date in the testing set.

3.5 Neural network characteristics

The network is a standard Trained Network from the PyBrain library, using the library’s Back Propagation Trainer. Over the course of developing and training the network, each of the following configurable characteristic was adjusted to find the most optimum combination:

- Number of Hidden Layers (formula below)
- Learning Rate = 0.001
- Momentum = 0.99
- Maximum Epochs = 20

The number of Hidden Layers that was found to produce the best outcome is represented by a simple formula related to the number of input features (1st layer neurons) to the network:

Figure 1: Formula for finding the optimum number of hidden layers

$$Hidden\ Layers = \lfloor \frac{Features}{3} \rfloor$$

3.6 Measuring model accuracy

After the network is trained, each test data point is run through the model, and the model's predicted outcome is measured against the actual outcome of that days stock price. The accuracy of the model as a whole is calculated simply by measuring the total number of predictions, divided by the number of correct predictions (actual outcome is saved as meta-data with the training and test data):

Figure 2: Formula for determining model accuracy

$$Model\ Accuracy = \frac{Correct\ Predictions}{Total\ Predictions}$$

4 Results

4.1 Method 1

Method 1 of grouping the data uses a mid-point value to separate data into test and training groups. This method shows a positive value in the predictive power of the model generated by the network, with the most predictive model generated being 76.7% accurate. This is less accurate than the the models generated using Method 2 for grouping test and training data – but perhaps a more accurate simulation of real world use.

Figure 3 shows the distributions of predictions for up-change and down-change predictions. The left distribution (blue) shows predictions for data who's actual outcome was a downward change. The right distribution (red) shows predictions for data who's actual outcome was an upward change. While there is an overlap between distributions, a distinct distribution is formed for each category of prediction.

Figure 4 Shows the prediction error and success rates for both downward and upward change predictions, if a threshold is set at 0.5 to differentiate between negative and positive change predictions.

4.2 Method 2

Method 2 of selecting training and test data involved iterating through the data set, and randomly assigning data points to either training to testing groups.

Figure 3: Model accuracy distribution

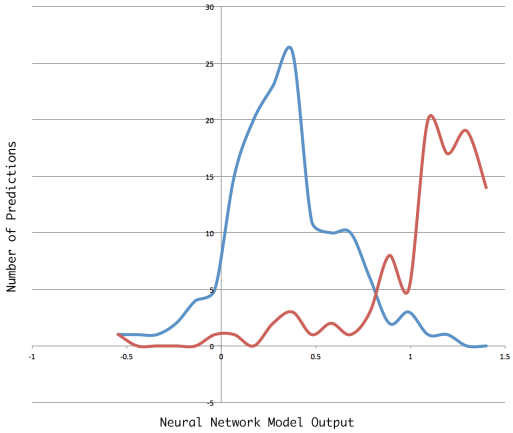
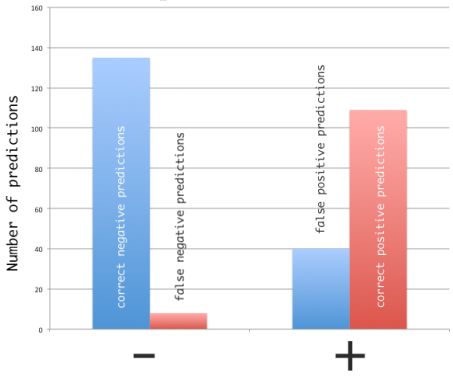


Figure 4: False and positive error and success rates



This model achieved a maximum accuracy of 82.9%.

4.3 Model Accuracy

Over the course of developing, testing, and tuning different neural network configurations, data was recorded about the accuracy of each model generated by the network. Figure 5 shows the accuracy distribution of 200 models (100 created using each method of assigning groups).

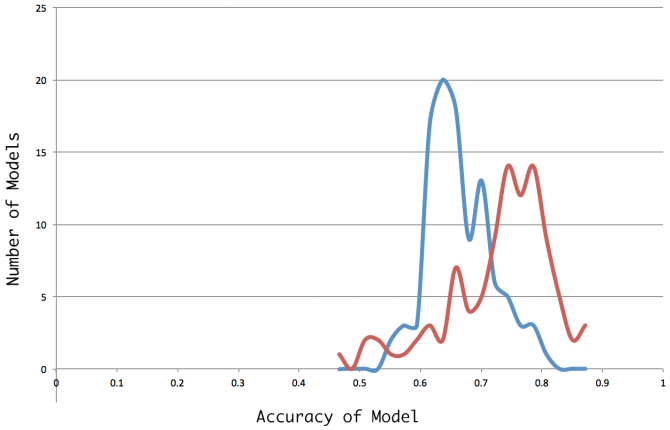
Method 1 (blue series on the left) has an average accuracy of 65%, with a standard deviation of 5%.

Method 2 (red series on the right) has an average accuracy of 72%, with a standard deviation of 8%.

5 Discussion

Returning to the initial research question – Can a model be created which will predict future stock price change direction, using English language news and

Figure 5: Accuracy of models



corresponding historical stock data? – It is fair to answer: yes. With mean accuracies in the 60-70% range, it is conclusive that there is predictive power in Natural Language Processing of news data and Machine Learning techniques on this vectorized data.

5.1 Practical Limitations

This research project has shown that it is possible to predict future changes in stock price, based on a machine learning model of textual news data of certain stocks. There are however, practical limitations to this approach:

- Time to acquire news data: The model and technique could be employed by traders with fast access to news data (for example, a subscription to the AP News Wire). A service such as this could provide news articles or press-releases within milliseconds of publication. Trades by this group of investors (High Speed Investing) would affect the price of the stock long before the 5-10 minutes required for that same news data to appear on the Google News Service[6].
- Location of trades: Trades located physically closer to a stock exchange (with the same prediction information) will be processed first. The price would adjust accordingly, before a further away trade was executed[8].

6 Future Work

Questions that may be answered through further research are:

- Would increasing the scope of the data to more than the top 10 holdings of the S&P 500 provide more accurate predictions?
- Could another Natural Language Processing technique other than the Bag-of-Words method provide better vectorized data?
- Could the accuracy of Method 1 for data group division be improved?

References

[1] Yahoo! finance, Dec. 2015.

[2] A. C. Andersen and S. Mikelsen. A novel algorithmic trading framework applying evolution and machine learning for portfolio optimization. Master's thesis, Department of Industrial Economics and Technology Management (IØT), http://blog.andersen.im/wp-content/uploads/2012/12/ANovelAlgorithmicTradingFramework.pdf, Dec. 2012.

[3] J. Bollen, H. Mao, and X.-J. Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1–8, Oct. 2010.

[4] R. E. Chris Umbel. *NaturalNode/natural*. https://github.com/NaturalNode/natural.

[5] C. Dunis, J. Laws, and J. Rudy. Profitable mean reversion after large price drops: A story of day and night in the S&P 500, 400 mid cap and 600 small cap indices. *Social Science Research Network Working Paper Series*, June 2013.

[6] M. Farrell. High speed trading puts investors on losing end - aug. 15, 2013. Aug. 2013.

[7] V. Paruchuri. Natural language processing tutorial. Technical report.

[8] G. Rogow. Colocation: The root of all High-Frequency trading evil? - MarketBeat - WSJ. Sept. 2012.

[9] T. Schaul, J. Bayer, D. Wierstra, S. Yi, M. Felder, F. Sehnke, T. Rückstieß, and J. Schmidhuber. *PyBrain*, 2010.

[10] J. J. Zhai, N. N. Cohen, and A. Atreya. Sentiment analysis of news articles for financial signal prediction. 2011.