Predicting Stock Prices through Textual Analysis of Web News

Daniel Gallegos, Alice Hau

December 11, 2015

1 Introduction

Investors have access to a wealth of information through a variety of news channels to inform them on their decisions to buy and sell stocks when managing their portfolios. Our project's goal was to simulate and improve on this process and predict stock price fluctuations of specific companies through a supervised learning approach to textual analysis of recently published and relevant articles on the web.

2 Data

We limited our scope to examine four prominent tech companies: Apple, Microsoft, Amazon and Tesla. Our assumption was that these companies, because they operate in similar spaces, would react similarly to the same text features. Using python, we scraped the sites of Bloomberg Businessweek, TechCrunch, and the Motley Fool for a month's worth of relevant news articles and press releases based on searches by the company name and ticker symbol. We also computed the stock fluctuation for each company for each day of news collected. We collected a total of 1,123 articles over 30 days. After collecting the articles, we wrote scripts to parse their content, convert all words to lower case, remove punctuation, and stem words to their base form using the Porter stemming algorithm.

Raw Text	**Processed Text**
HTCs One A9 has been criticized as an Apple iPhone knockoff because of the resemblance in the exterior design. However, the ambitious A9 will hit the store shelves today.	htcs one a9 has been critic as an appl iphon knockoff becaus of the resembl in the exterior design howev the ambiti a9 will hit the store shelv today

3 Learning Algorithms

We treated this as a natural language processing and binary classification problem where stock price either improved or did not regardless of the percent change. To predict these two classes we had two types of features: word frequencies and numerical measures derived from sentiment analysis. For each method, we trained our models with holdout cross-validation by training on the data scraped for Microsoft, Tesla, and Apple and testing on the data scraped for Amazon. This problem lends itself to supervised learning algorithms such as SVMs and logistic regression.

SVMs are among the best "off-the-shelf" supervised learning algorithms. SVM's excel at efficiently handling high dimensional feature spaces because of their use of kernels. Also, their margin maximization usually creates very robust predictions. SVMs are a good starting point for almost any type of classification problem.

Logistic regression is the other natural choice for a classification problem. It assumes very little information about the training data, and it tends to do well even with small amounts of data. In addition, it is a very simple learning algorithm to implement, so it is considered a good first step.

4 Feature Selection

The primary focus of our project was feature selection–filtering out the most influential features in text articles on stock prices. We incrementally tweaked our selection with two approaches to choosing subsets of features. In our second approach, we also experimented with aggregating content.

4.1 First Approach

The first was a "bag of words" approach similar to how we implemented spam detection in class. We created a lexicon of words using frequency tables, and then ran our supervised learning algorithms. Limiting our features – in this case tokens – was our primary concern in this approach. Below are different methods we used to select features, which we then used holdout cross-validation to find test error estimates.

- Created a lexicon of the 250 most frequently used stemmed tokens across all articles, ignoring numeric values and "stop" words such as "the," "a," "as," etc.,
- Created a lexicon with sentiment analysis in mind by using an existing sentiment lexicon to include only the 100 most frequently used "positive" and "negative" tokens (ignoring "neutral" ones). We sought to test our hypothesis that some words were more consequential than others. Examples of positive and negative tokens

(stemmed to be consistent with our processed data) are: “accomplish, amaz, matur, rich. . . ” (positive) and “abolish, accus, lose, poor” (negative)

4.2 Second Approach

Our second approach used as features the following:

- Percentage of positive tokens in an article
- Percentage of negative tokens in an article
- Percentage of the relevant company’s name or ticker symbol mentions. This was to differentiate between more and less relevant articles. An article that mentions ″Microsoft″ 10 times might be more relevant than one that mentions it once.
- Percentages of positive and negative tokens in the sentences immediately surrounding a company name/symbol mention. This was to make our feature selection more contextually sensitive by giving weight to sentiment-charged statements in close proximity to the specific mention of a company.

In this approach, we also took into account bigrams such as “not profitable” or “no revenues” to make our sentiment analysis more sensitive to the meaning of pairs of words in context.

4.3 Aggregating Content

We found that, on average, the length of any given article was approximately 600 words. Given that many of those words would be ignored by our feature selection of either most frequently occurring words, positive or negative words, and non-stop-words, we hypothesized that a single article did not have enough information to make an accurate prediction. So, for our second approach to feature selection, we took three approaches to aggregating content to make predictions not on the information of a single article, but on the information of the content of multiple articles aggregated together:

- Aggregate all article content for a company for each day. We did this by summing up all of the frequencies of positive words, negative words, and company mentions over all of the articles for each day and for each company. Make predictions for the next day given all content for the current day.
- Aggregate all article content for a company for each day and the previous day, then make each prediction for the next day given its previous two days of content.
- Aggregate all content for a company for each day and the previous two days, then make each prediction given three total days of news content.

5 Results

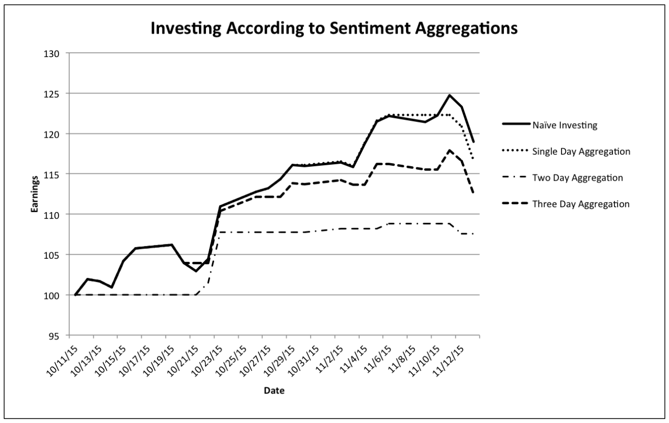
The following table contains holdout cross-validation errors for the models using the first approach:

Feature Selection	SVM	Logistic Regression
250 Most Frequently Used Words	0.4821	0.4502
100 Most Frequent Positive and Negative Words	0.5179	0.3865

The following table contains holdout cross-validation errors for the models using the second approach:

Feature Selection	SVM	Logistic Regression
Non-Aggregated Content	0.5817	0.5498
Single-Day Aggregated Content	0.4483	0.4138
Two-Day Aggregated Content	0.5769	0.4230
Three-Day Aggregated Content	0.4783	0.4783

To measure whether our classifier would fare well predicting the stock market, if the classifier predicted that the stock price would rise that day, we decided to invest money that day in the market and if the classifier predicted that the stock price would fall that day, then we didn't invest money in the stock. To put it in perspective, we decided to compare it to the ”naive” approach which was the case where we would have invested in the stock every single day.



6 Discussion

Through our sentiment analysis of news articles, we achieved error rates fluctuating between .4 and .6, and thus were not able to conclusively find a model that accurately predicted the fluctuations in stock prices. However, we at least found that our market return on investment using our best model was approximately 15 percent. Logistic Regression consistently out-performed SVMs as a learning model.

For our first approach to feature selection, using the bag of words approach, it is unclear whether or not the choice to look at only the most frequent positive and negative words rather than all words was actually improvement since the error rate improved by roughly 6 percent for logistic regression, but worsened by 4 percent for SVMs. For our second approach to feature selection, where we calculated measures for features,it appears that aggregating content over a day to make predictions performed the best. This could indicate that the biggest predictor for a day's stock price, if it affected by the news, is the previous day's content. Further, that might indicate that stock prices are more vulnerable to immediate rather than long-term changes in public sentiment about the company.

7 Future Work

Aggregating data greatly shrank our training data set by roughly a magnitude of 10 since it reduced our number of observations to the number of days over which we collected data. Hence, our results from the aggregated data had higher variance than the other methods, and it would be productive to scrape more data to train and test on.

It would have been interesting to see how effective our models for predicting sentiment actually were. For example, using the features we selected, could we predict whether an article was positive or negative? It would probably be beneficial to research and apply more advanced sentiment analysis techniques in order to be able to capture more of the nuances in the articles.

Further, our models were built on the large assumption that the stock prices for a single company could be predicted by models trained on other companies. While it might be the case that there is one model common to all tech companies, and that they are all impacted similarly by news, it's also highly likely that companies respond differently to the same words. For example, Amazon might be more affected by the word ”delivery” or ”ebay” than Microsoft, Tesla, or Apple would because those words are more relevant to its service. Thus, for future work, it might be fruitful to make separate models for each company, training only on news articles relevant to that company, and making predictions only from articles related to that company.

8 References

[1] Porter, M.F. An algorithm for suffix stripping. Computer Laboratory, 1980.
[2] Liu, Bing. Sentiment Analysis and Opinion Mining. Morgan and Claypool Publishers, May 2012.