Stock Market Prediction via Twitter Sentiment Analysis

Bobby Swingler, Isai Mercado Oliveros, Isaac Kurth
CS 478, Winter 2016
Department of Computer Science
Brigham Young University

Abstract

Stock market prediction is one of the holy grails of the business world, and is considered to be an unpredictable problem. For the scope of this course, an experiment has been performed to determine if Twitter sentiment analysis can help predict stock prices. Thousands of tweets have been manually and automatically collected to produce a dataset for ML training. A daily percent-change history for the Apple, Inc. portfolio has also been collected. After training with several models, the backpropagation MLP provided us with the best results of a RMSE within eight cents of the target stock price. More data and testing will hopefully prove the validity of these results.

1 Introduction

Stock prediction analysis is divided into three methodologies. They are fundamental analysis, technical analysis, and technological analysis. Fundamental analysis is often performed by financial experts who evaluate the economy as well as the given corporation, and is very much a qualitative approach toward stock prediction. Technical analysis relies solely on financial and portfolio history in determining prediction. Lastly, technological analysis, a subset of technical analysis, uses machine learning to discover insights and trends in financial data that may not be apparent to any human.

The Twitter experiment takes a different approach toward technical analysis. Simply put, we looked to social media and public opinion to determine stock prediction. To simplify the experiment, we have focused on the portfolio for Apple, Inc. (AAPL). This corporation came under fire during March 2016, as the FBI issued a court hearing over back-door access to a suspected terrorist's iPhone. The heated public debate over privacy and encryption made Apple, Inc. a prime choice for our study.

2 Methods

A description of our approach toward sentiment analysis and stock prediction.

2.1 Data Sources

We decided to use Twitter as our primary data source after a brief look into its API and documentation. Initially, we began by using the Twitter streaming API with the hopes that we could perform sentiment analysis on each tweet, as well as use tweet metadata to produce several ML features. However, we discovered the streaming APIs have no access to Twitter historical data, nor do they provide insight into tweet metadata, as the incoming tweets are so new that they have zero re-tweets, likes, etc. After discovering this issue, we turned our focus to the legacy APIs that give access to historical data. We quickly discovered new issues: Twitter imposes heavy limitations on queries against their API, and they only keep seven days of history. This came as a shock, as we wanted at least one year's worth of tweets for several specific queries. Along these lines, we found third-party companies selling historical Twitter data, as Twitter does not keep an archive. In some cases, said companies were selling historical Twitter data at $500 per query. Overcoming this potentially expensive issue proved to be laborious. Fortunately, we discovered Twitter's Advanced Search, a somewhat hidden feature, which allows for historical queries from the front end webpage; there is no API support whatsoever. So we began manually entering queries into Twitter, and screen scraping the results into spreadsheets. Luckily, we found a screen-scraping tool that allowed us to identify specific DOM elements by JQuery tag, and automatically put the results into CSV format. In this manner we were able to create a dataset spanning just over 30 days. We also used the Yahoo Finance API to gain access to historical stock prices for Apple, Inc. which went very smoothly.

2.2 Data Sets

After discovering several difficulties with the Twitter APIs, we minimized our prediction domain. Initially, we planned on mining one year's worth of data, but had to move forward with one month of manually-collected data. The data set is created in the following manner.

First, we collect tweets containing references to Apple as well as a few related entities for a specific date. Second, we pre-process the tweets by removing URLs and non-Latin characters in preparation for sentiment analysis. Third, we compute each tweet's sentiment using the open-source Stanford NLP Java Library. This natural language processing library returns sentiment as a double value between zero and four, zero being "very negative" and four being "very positive". Fourth, we find the mean sentiment for the given query for the given day, and place this mean value in the corresponding cell in our data set. Therefore, entries in the data set represent the mean sentiment for the specified company (column) and the specified date (row).

Making the data set was difficult, and each row represents thousands of tweets. Also, the issues with the Twitter APIs caused us to re-think our features and refactor our code several times in order to build the data set in a timely manner. In addition, we discovered the Stanford NLP uses a recursive neural network for determining sentiment, which is quite slow. To facilitate the dataset creation process, we wrote a multi-threaded dataset builder which reduced parsing from one hour to about fifteen minutes.

2.3 Selected Models

The multi-layer perceptron has been our model of choice since starting the project. Our neural network has the following parameters: One hidden layer with 20 nodes. Learning rate of 0.3, and momentum of 0.2. For a complete description of model usage, see section 7.1 'Other Models'.

2.4 Dataset Example

See Table 1

3 Initial Results

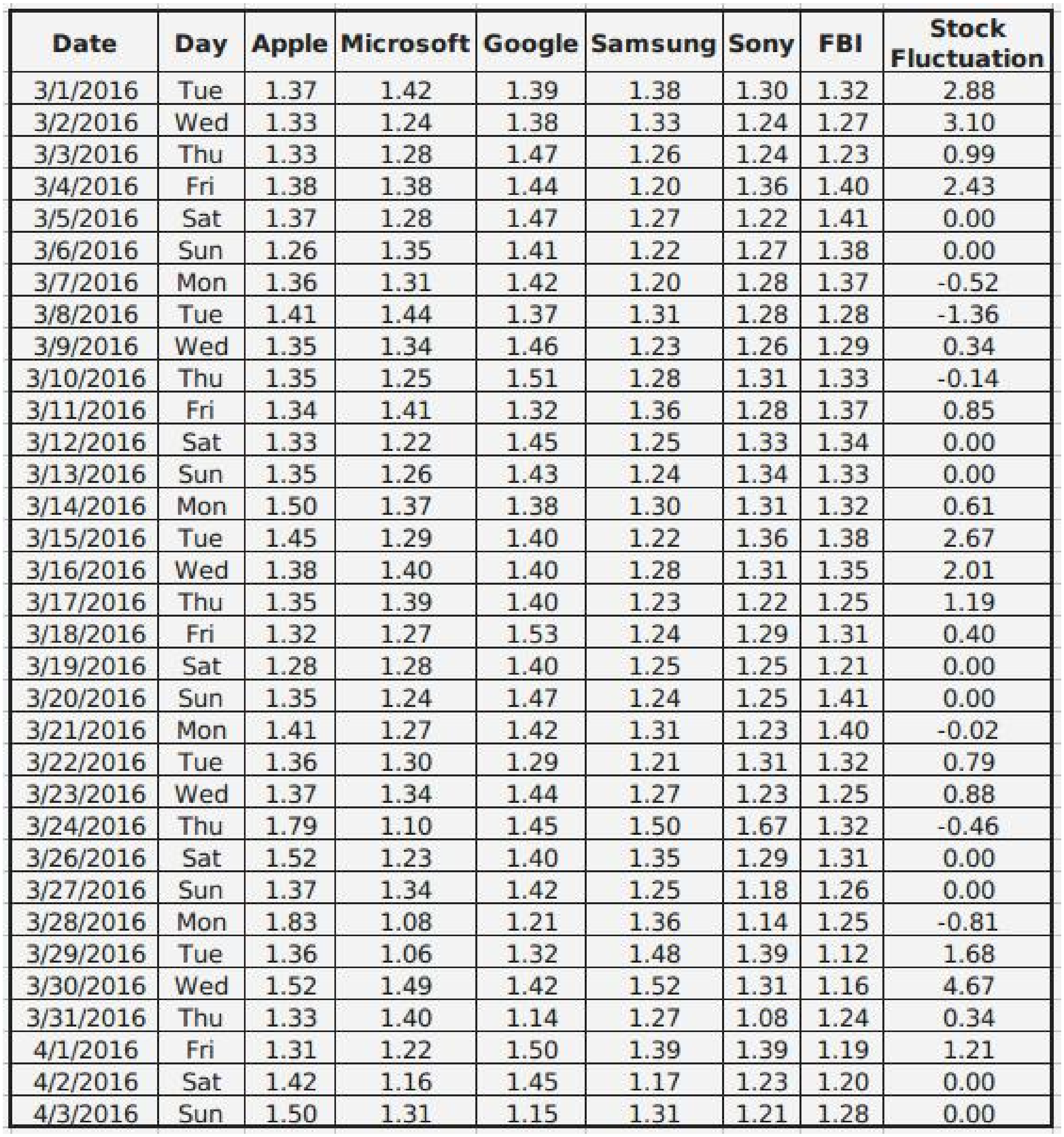
Our neural network was fairly small but it was able to overfit the training data set with 99% accuracy. We then used 10-fold cross validation to avoid overfitting, and found that the model was accurate within 80 cents of the actual label. However, this fact resulted to be not very informative when we wanted to predict unseen data because we had to tweak the columns of the training data set a little bit in order to get more accurate on unseen data.

4 Data and Feature Improvements

Initially, we had the following columns in our training data set: (Week Day, Apple, Microsoft, Google, Samsung, Sony, FBI, iPhone, Android, Mac, Windows, iTunes, Spotify). We included these related companies with the assumption that related opinions would help the MLP to learn more effectively. However, we found out that the trend of the sentiment between Apple product users against competing product users followed the company vs company trend closely. For example, the pattern that we found between tweets containing Apple vs tweets containing Google, were very similar to tweets containing iPhone vs tweets containing Android. Therefore, adding more columns did not improve the amount of information that tweets were already giving.

In addition, when running experiments with 10-fold cross validation, we trained and predicted by leaving the column "day". Our assumption was that there was a possibility that opinions were related to the day of the week. For example, people would share more negative tweets on Mondays, and more happy tweets on Friday. However, when trying to predict unseen days, the model predicted better without that column. We think that this was the case because the relation between sentiment and day of the week was not strong enough to be useful in a training set of 30 rows. On the contrary, it limited the learning of the model because it could not compare opinions with great liberty if it had to take into consideration the day of the week. Therefore, it was removed and the model was able to get more accuracy predicting new data.
Apple vs tweets containing Google, were very similar to tweets containing iPhone vs tweets containing Android. Therefore, adding more columns did not improve the amount of information that the aggregation of tweets was already giving us.

As mentioned earlier, making the data set took a long time, since calculating sentiment for several thousand tweets by the Stanford NLP library is very slow. Therefore, for the purposes of this experiment we selected a subset of our original columns and predicted Apple's stock fluctuation based on the subset "Week Day, Apple, Microsoft, Google, Samsung, Sony", and "FBI". By having fewer columns, we are also able to retrain our model in a timely manner. Retraining the model is necessary since it needs to know the new opinions of the twitter users in order to predict the stock price for the following day.



Date	Day	Apple	Microsoft	Google	Samsung	Sony	FBI	Stock Fluctuation
3/1/2016	Tue	1.37	1.42	1.39	1.38	1.30	1.32	2.88
3/2/2016	Wed	1.33	1.24	1.38	1.33	1.24	1.27	3.10
3/3/2016	Thu	1.33	1.28	1.47	1.26	1.24	1.23	0.99
3/4/2016	Fri	1.38	1.38	1.44	1.20	1.36	1.40	2.43
3/5/2016	Sat	1.37	1.28	1.47	1.27	1.22	1.41	0.00
3/6/2016	Sun	1.26	1.35	1.41	1.22	1.27	1.38	0.00
3/7/2016	Mon	1.36	1.31	1.42	1.20	1.28	1.37	-0.52
3/8/2016	Tue	1.41	1.44	1.37	1.31	1.28	1.28	-1.36
3/9/2016	Wed	1.35	1.34	1.46	1.23	1.26	1.29	0.34
3/10/2016	Thu	1.35	1.25	1.51	1.28	1.31	1.33	-0.14
3/11/2016	Fri	1.34	1.41	1.32	1.36	1.28	1.37	0.85
3/12/2016	Sat	1.33	1.22	1.45	1.25	1.33	1.34	0.00
3/13/2016	Sun	1.35	1.26	1.43	1.24	1.34	1.33	0.00
3/14/2016	Mon	1.50	1.37	1.38	1.30	1.31	1.32	0.61
3/15/2016	Tue	1.45	1.29	1.40	1.22	1.36	1.38	2.67
3/16/2016	Wed	1.38	1.40	1.40	1.28	1.31	1.35	2.01
3/17/2016	Thu	1.35	1.39	1.40	1.23	1.22	1.25	1.19
3/18/2016	Fri	1.32	1.27	1.53	1.24	1.29	1.31	0.40
3/19/2016	Sat	1.28	1.28	1.40	1.25	1.25	1.21	0.00
3/20/2016	Sun	1.35	1.24	1.47	1.24	1.25	1.41	0.00
3/21/2016	Mon	1.41	1.27	1.42	1.31	1.23	1.40	-0.02
3/22/2016	Tue	1.36	1.30	1.29	1.21	1.31	1.32	0.79
3/23/2016	Wed	1.37	1.34	1.44	1.27	1.23	1.25	0.88
3/24/2016	Thu	1.79	1.10	1.45	1.50	1.67	1.32	-0.46
3/26/2016	Sat	1.52	1.23	1.40	1.35	1.29	1.31	0.00
3/27/2016	Sun	1.37	1.34	1.42	1.25	1.18	1.26	0.00
3/28/2016	Mon	1.83	1.08	1.21	1.36	1.14	1.25	-0.81
3/29/2016	Tue	1.36	1.06	1.32	1.48	1.39	1.12	1.68
3/30/2016	Wed	1.52	1.49	1.42	1.52	1.31	1.16	4.67
3/31/2016	Thu	1.33	1.40	1.14	1.27	1.08	1.24	0.34
4/1/2016	Fri	1.31	1.22	1.50	1.39	1.39	1.19	1.21
4/2/2016	Sat	1.42	1.16	1.45	1.17	1.23	1.20	0.00
4/3/2016	Sun	1.50	1.31	1.15	1.31	1.21	1.28	0.00

Table 1: Example dataset featuring mean sentiment analysis for given queries.

5 Final Results

We had the notion that opinions were related to stock prices, but seeing accurate predictions to the real stock price based on what people said in twitter is very surprising. To test our model we trained it with data from this past month, and then computed opinions and stock prices for the next three days. As mentioned before, we anticipated the relation between opinions and stock fluctuations to be very complicated, and because of that, we chose a neural network to learn those relationships. We found that for the first day after training, the model predicted the stock fluctuation almost perfectly. The target fluctuation was +0.70 cents, and the model predicted +0.60 cents. For the second day, the prediction's magnitude returned a bit off, but the direction was still correct. The target was -0.60 cents and the model predicted -1.10 dollars. The third day, the prediction's magnitude was very off, but the direction was once again correct. The target was +1.20 dollars and the prediction was +4.80 dollars.

We think that the model was able to learn the curve by the 30 rows of the monthly training data set, but it did not have enough training data to nail down the magnitudes. That explains why all predictions were correct on direction, but magnitude became worse and worse the next days. It looked like if uncertainty was increasing as we dived more into the future. We think that if we could create a training data set worth one year of information we could predict more days in advance not just the very next two or three days; however, even predicting one day with this amount of accuracy is very impressive if we consider that it is all based on opinions for one month.

6 Conclusions

Gathering large quantities of historical Twitter data has proven to be difficult, but we have been able to pivot our project to create a successful experiment. The progression of selected features, as well as selected models has shown that MLP backpropagation has the most promise in solving this problem. Current results have been exciting for such a small dataset, and could also be accurate as a Boolean value - will the stock price increase or decrease today? We are hopeful that future testing and experimentation will help improve the model.

7 Future Work

Collecting our own Twitter archive would be the first step in making this problem more user friendly, as Twitter does not offer API access to tweets older than seven days. To do this properly, we would need to create several Streaming API Twitter bots (one for each search term) and have them catalog tweets 24 hours a day for the foreseeable future. We would also focus on improving sentiment analysis, and attempt to glean multiple features from this area. Further experiments with the dataset could be performed to determine the optimum range of opinion, and if an opinion/stock price delayed effect exists. Today's opinions may have an effect on stock price at some point in the future. We do not know the range or if a delay exists, so the question is open for research. Perhaps today's stock price is affected by what people thought last week, or maybe last year. If we can identify an optimal range, we could be more confident in our prediction. This would improve the accuracy of the social mining algorithm to predict the stock price, and looking at a bigger picture, it would be a good technique to predict social phenomena based on past opinions.

7.1 Other Models

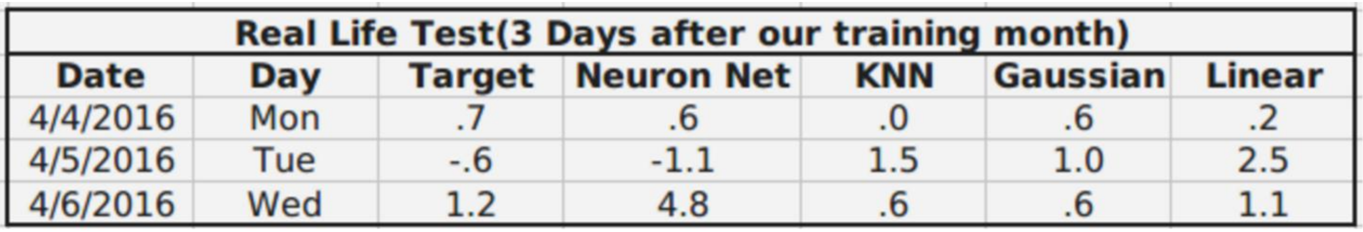
We used four models on this data set. Neural networks were our first choice for prediction because we assumed that relations between opinions and stock fluctuation were very complicated, but we also ran other models based on different assumptions.

First, to model history repeating, we chose knn. The prediction model looks at historical data and finds k closest records and see what the fluctuation of the stock was in those records. For example, if in the past we have seen 3 times that positive sentiments for Apple and negative sentiments for all other companies means that Apple's stock price increased that day, then we can be confident that today's stock price is going to go up by a similar magnitude, if we saw the same pattern. However, knn did not work very well with our 3 test days, because the training data set only contained 30 rows, but we suspect that if we had a training data set worth several years, this method would be very accurate.

To model a distribution of sentiments, and not just one average sentiment, we used a gaussian process. We can simulate this fact by treating every entry in the training set as a mean with some variance that forms a gaussian distribution per entry. Regression returns the expected value of the unseen data point. We chose a gaussian kernel because it is likely that the majority of sentiments group together in a bell curve shape. The results were very accurate for the first and third day of the test days, but our confidence dropped because the second day vas very inaccurate.

Finally, we used linear regression because there might be situations in which we want to know the direction of the overall social behavior but not the specifics. For example, in the case of predicting Apple's stock price, we might just want to know if the overall price trend is simply going up or down for this month, in which case someone might want to buy stock for long term investment.

See Table 2 for multiple model results.



Real Life Test(3 Days after our training month)						
Date	**Day**	**Target**	**Neuron Net**	**KNN**	**Gaussian**	**Linear**
4/4/2016	Mon	.7	.6	.0	.6	.2
4/5/2016	Tue	-.6	-1.1	1.5	1.0	2.5
4/6/2016	Wed	1.2	4.8	.6	.6	1.1

Table 2 – Multiple Model Results