Measuring the Reliability of Twitter Sentiment Analysis Applied to Stock Market Prediction

Nicholas Cowles

Project URL: https://github.com/NCowles15/cse482\_project

**ABSTRACT**

This project is an examination of the application of sentiment analysis to Twitter data and using the results to try to predict stock market trends. The stock market is a notoriously difficult system to predict, but it was my hope that gauging popular perception of a stock via social media would provide a good predictive measure for that stocks performance. I approached this as a normalized regression problem where I aimed to compare the daily change in sentiment intensity with the daily change in a stock price. For this task I chose the stock for Apple Inc. due to its sizeable cultural presence. The results of this analysis showed little predictive value in using sentiment analysis in this way. However, I was able to identify several factors that

# INTRODUCTION

This project is an application of stock market prediction. This type of problem has captivated analysts for as long as the stock market has existed. This is due to the promise of immense fortune to anyone who could succeed in reliably predicting future stock prices. But this problem has proven quite difficult and complex due to the sheer number of variables at play. What makes the problem of stock market prediction so compelling is that one can become rich in the process. If one knows when a stock will go up, they can buy the stock at the lower price, sell it once it reaches peak value and profit off the difference. But trying to do this purely through mathematics and history has proven to be quite unreliable. This is where sentiment analysis comes in. Sentiment analysis is a new technique that has become available to analysts in the age of social media. The idea is to evaluate text for positive or negative sentiments on a topic to gauge the overall perception of it. This could be useful in stock market analysis, as in theory the value of a stock is heavily linked to how its value is perceived by the world at large. It follows then that if one could reliably monitor shifts in this perception that one could predict the subsequent change in stock value.

The goal of this project is to use user sentiment in twitter data and evaluate if it provides useful information to predict changes in the price of a stock. To do this I plan to apply regression methods on a combination of historical price and twitter sentiment data and evaluate whether the prediction results are better than using historical data alone.

There are two main types of data I collected for this project. The first dataset I collected was historical twitter data from 2015-2016 from MSU research archives. The second was historical stock prices for every major US company for as long as they have existed. Both of these datasets required trimming to be useful.

The largest challenge in collecting and preprocessing data came from the Twitter data. The raw dataset is 2TB and stored on a remote server. This presented several issues. The first is that I could not store it locally, since I do not have that much memory available. To mitigate this I decided to write a python script to trim the data and only keep tweets with mentions of Apple or its stock. This is where the second issue came in. My connection to the remote server did not have a lot of bandwidth, so I could not quickly read the file remotely store the results locally while reading. My solution to this was to trim the archive from my student account on the MSU server and then download the trimmed results all at once. This presented a third problem in that my student account only had 3GB of space left yet I had no idea how much space would be needed for the trimmed data. I solved this problem by running the trimming script in batches and then moving the trimmed data to my local machine before proceeding to the next batch.

I would have preferred to work with the data on an AWS cluster that could handle the entire dataset, but the time needed to transfer that data would likely have far exceeded how long I could afford to keep such a large server running. Even with all the mitigations I went through it still took several days to trim the whole dataset, and that does not include debugging the script I used and occasionally having to restart the whole process. At the end of the trimming I was able to get the data from 2 TB to nearly 4GB, which was much more manageable but did present a limitation for the analysis.

I had hoped to find evidence supporting the reliability of sentiment analysis in stock market prediction, but in the end the results pointed toward the opposite. What I found is that sentiment analysis on its own is outperformed by mere historical analysis of stock price changes. I cannot say whether these findings are due to flaws in sentiment analysis in general, the nature of stock market prediction or various limitations of my experimental design.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*CSE881-2015*, Month 1–2, 2004, City, State, Country.

Copyright 2004 ACM 1-58113-000-0/00/0004…$5.00.

# DATA

I used 2 primary sources of data.

* 1. Twitter data from: /user/research/ptan/data/Twitter/ hosted at black.cse.msu.edu
  2. Historical stock price data from Kaggle at: https://www.kaggle.com/borismarjanovic/price-volume-data-for-all-us-stocks-etfs?

Collection of the stock data was as simple as downloading a zip file from Kaggle. Collection of the twitter data was far more complicated, largely due to resource limitations on my end. The size of the dataset meant that I had could not store the whole dataset locally and needed to either trim it or host it on a separate server. The bandwidth of the MSU server connection meant that I could not afford to host a server for it, so I wrote several scripts to read trim, and reformat the data to only include tweets relevant to Apple.

The stock price data was stored in .txt files but was formatted like a csv which made converting it straightforward. The Twitter data was stored in files without extensions with each line being a JSON formatted tweet. This format was not easy to work with so I reformatted the data into .JSON files that stored the tweets in an array of JSONs that could be easily loaded later. In order to perform the linear regression on the tweet data with stock prices I needed to merge the two datasets. This meant converting the JSON tweet to a csv with more compatible formatting. I could then load the new datasets into pandas dataframes and merge them along a common date attribute.

The raw stock data goes back decades, for some companies there is over a centuries worth of historical data. For Apple in particular the data spans from September 9th 1984 to October 11th 2017. The stock data has missing values caused by days when the stock market was closed, though because of this they occur predictably and are easier to work around. The raw Apple stock data consists of 8365 observations which each consist of 6 features. The only features of interest in this data were the closing price and the date, though the other features could be used in a more sophisticated analysis.

The twitter data was taken from a span of April 2015 to April 2016. This data has many unnecessary feature that needed to be trimmed out at several stages after initial collection. The raw twitter data for just Apple related tweets consists of which each contain around 70 features.

I chose to discard dates with missing stock values. I decided to calculate the percent daily change in both sentiment and stock price to use in my regression. In order to do this meaningfully I also needed to normalize the daily change in sentiment intensity using a Z-score transformation. I was able to spot some outliers in the twitter sentiment data, but chose not to discard them as I was curious how they correlated to stock price changes.

The final merged dataframe consisted of 227 rows, 7 columns. From this dataframe 2 features were extracted into their own series objects to be fed into the regression. This final dataframe is 13KB.

|  |  |
| --- | --- |
| Stock Data Features | |
| Feature | Description |
| Date | YYYY-MM-DD date value |
| Open | Opening price of the stock |
| High | Peak price of the stock that day |
| Low | lowest price for the stock that day |
| Close | closing price of the stock |
| Volume | amount of this stock traded |
| OpenInt | Value for if the stock was open |

The final dataframe has 7 features: date, Date, SentimentSum, Open, High, Low, Close, Volume and OpenInt. The predictor attribute of the merged data is the calculated change in sentiment intensity. The target value is the change in stock price.

# METHODOLOGY

The twitter data flow starts on the MSU remote server. From there it is read and trimmed on the server. The trimmed data is sent to my local machine in batches to make room for more twitter data on the server. Once this is complete I reformat the trimmed data into more compatible files on my local machine. Then I perform the sentiment analysis of all tweets. The analysis produces a .csv that maps the date to the sum of all sentiments on that day and a .json which contains an array of all the dates found. These dates are then used to trim the apple data into a more manageable .csv. Finally the data is combined, normalized and regressed on, then compared to the baseline regression.

I used sklearn to perform the linear regression and provide metrics for the produced model.

Script descriptions:

* trim\_stocks.py: Takes the raw Apple stock data a produces a file with only data from only the dates that have tweets
* remote\_gather.py: Reads the raw twitter data on the remote server and creates a trimmed version of the data to be stored locally.
* reformat\_data.py: Reformats the raw twitter data into an array of JSON objects in a .JSON file.
* reformat\_data2.py: Creates a single JSON file by aggregating the other JSON files produced by reformat\_data1.py
* analyze\_sentiment.py: Analyses the sentiment of the text of all tweets in the local data set. Mainly Produces a list of dates and a csv of sentiment for a given date
* baseline\_data.py: Performs linear regression of Apple Stock prices uses previous 4 days of price change as the predictor.
* combine\_data.py: Combines stock and twitter data into final data frame. Normalizes the sentiment data and performs a linear regression.

Most of this project was done entirely in python and all libraries used come with Anaconda.

# EXPERIMENTAL EVALUATION

## Experimental Setup

This experiment was done on a Windows desktop machine with 1TB HDD storage and 8GB of RAM. To establish a baseline of comparison for the linear regression I did a regression on the same stock data but used the preceding 4 days of price change as the predictors instead of tweet sentiment.

Sentiment results:

* The sentiment based regression produced an RMSE of 1.4115 and a R-squared value of -0.0058. The small value of the R-square result implies that the generated model does not improve prediction over the mean. In addition, the RMSE value is quite large considering the range of the given data. These values are show that this model is quite inaccurate.

Baseline results:

* The historical based regression produced an RMSE of and 1.9771 a R-squared value of -0.1003. This R-Square value is still too low, though it is superior to the result of the sentiment regression. The lower RMSE also implies that it is superior to the sentiment based regression, though it is still large considering the size of the dataset. This is generally still an inaccurate as a predictor, but is more accurate than the sentiment based regression.

## Experimental Results

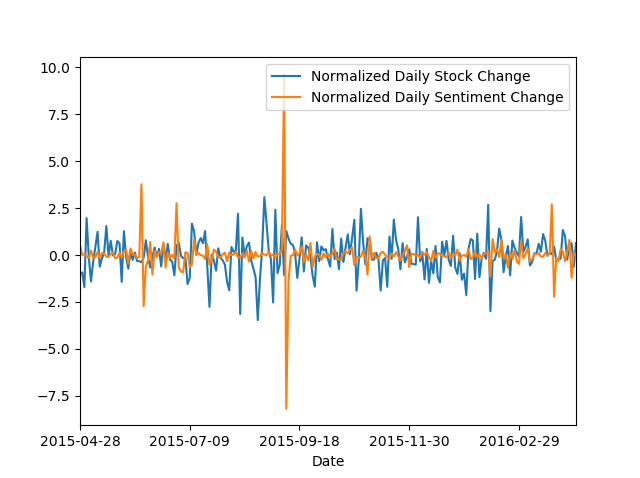


Figure 1

Figure 1 shows the correlation of daily stock price change and daily sentiment change. It is a good visual representation of why sentiment is not a good predictor of future stock price. There are occasions where price and sentiment seem to align, but there are far more where there don’t. This unreliability is means that this approach is not sufficient to predict stock price changes.

It is hard to gauge the significance of these results due to numerous confounding variables. It can be conclusively said that this approach to sentiment analysis is not sufficient as a predictor of stock market performance. But this cannot easily be generalized to sentiment analysis as a whole. It is possible that using a larger proportion of the twitter dataset might have produced more accurate predictions. It is also possible that other companies than Apple may be easier to reliably predict. There is also a question to be asked about how effectively the technology used in this project actually is at determining sentiment. It might be that with more sophisticated sentiment analysis of text the results would become more in line with predicted outcomes.

I would say that this project was successful in evaluating how effectively general twitter sentiments data can be used to predict stock market performance. In the case of Apple, the sentiments of a general twitter audience are unlikely to accurately reflect the perceived value of apple stocks. Though I would also say that it was not successful in fully tackling the proposed problem. This is partly due to the scope of the problem, but there are things I could have done to improve the odds of success. I could have used multiple approaches to sentiment analysis of tweet text to find the best one, though the size of the twitter data makes checking accurate sentiments difficult. I could have used a larger portion of the twitter dataset had I had more resources available. This also would have allowed me to evaluate more than one company and come to a more definitive conclusion on the approach.

# CONCLUSIONS

Summarize the overall findings and contributions of the project. If possible, provide suggestions for future work that could improve what you’ve done for the project.

# REFERENCES (at least 3 references)

1. Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, and Vipin Kumar *Introduction to Data Mining*, 2nd Edition, Addison Wesley, 2018.
2. Anand Rajaraman, Jeff Ullman *Mining of Massive Datasets* Cambridge University Press, 2011
3. Wes McKinney, *Python for Data Analysis: Data Wrangling with Pandas, Numpy, and iPython*, 2nd Edition, O’Reilly, 2017.