Stock Market Prediction of the Short-term Impact of President Trump's Tweet

Jiaxuan Ren
Computer Science
University of California, San Diego
jiaxuan.ren@jacobs.ucsd.edu

Kaihan Zhu
Computer Science, Economics
University of California, San Diego
kaihan.zhu@jacobs.ucsd.edu

Word cloud of Trump's tweets

1 Problem Description

Social media makes the spreading of news faster and broader, but at the same time, brings volatility to the stock market. Making predictions on the news events and react quickly gives comparative advantage in making more profits or reduce losses.

In this project, we try to understand and predict the impact of a very special kind of news, tweets from President Trump, which has become the most uncertainty in the market. We try to model the market's reaction to his tweets in the short term using supervised learning.

We tried two different models, Linear regression and Long Short Term Memory Recurrent Neural Network (LSTM), to predict the reaction of the market (rise / fall / stays the same) over a certain time after Trump makes a tweet. Each model have different training samples. Training samples for the Linear Regression are bags of words with each entry representing a tweet. Samples for LSTM are a transformation of his tweets into a sequence of vectors that represents each word (or subword). Labels are variations in the trading price of the ETF after a certain time frame.

2 Data

2.1 Datasets

We utilize two datasets. First part of our dataset is training samples. We obtained all of Trump's tweets from Jan.1, 2015 to Dec.31, 2017 from http://www.trumptwitterarchive.com/. It is a total of 14334 tweets. Table below shows some of the most frequent words in trump's tweets and their frequency. Figure 2.1(a) shows the length of Trump's Tweets.

Word	Frequency	Word (cont'd)	Frequency
will	1902	America	582
Trump	1874	just	578
great	1139	Great	560
Thank	1023	Trump2016	542
Donald	709	President	490
people	663	get	489
Hillary	611	like	460

Second part of our dataset is the labels. We obtained all of the trading records of the most popular S&P 500 ETF—SPY—from the NYSE database through Wharton Research Data Services. Figure 2.1(b) shows how absolute value of volatility varies as time elapses.

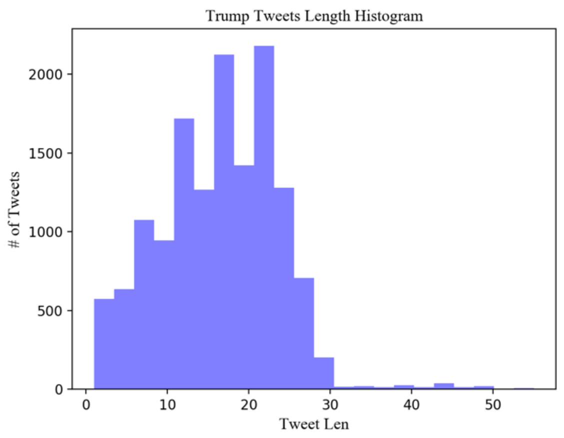


Figure 2.1(a) Trump's tweet lengths

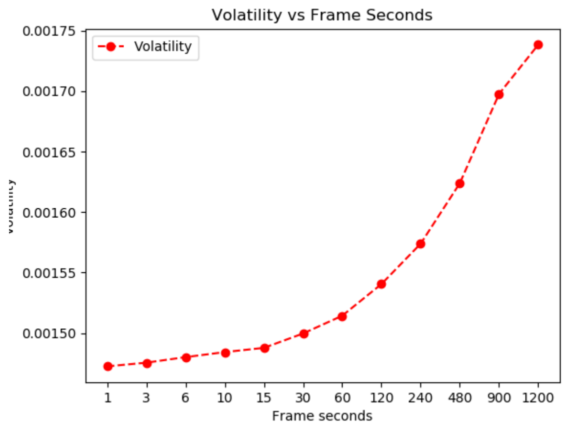


Figure 2.1(b) Relative volatility as time window increases

2.2 Data Processing

We need to process the ticks data from NYSE and the Trump's tweet data into those that we can use as imports into the database. We randomly choose 1/6 as validation data, 1/6 as test data, and the rest as train data.

2.2.1 Trump's Tweets Processing

Trump's tweets are processed differently for the LSTM and Linear Regression.

For LSTM, we used a subword encoding to encode the all of Trump's tweets into sequences of vectors where each sequence represents a tweet. Each vector represents a word, a part of a word, or a character.

This process will retain as much as information of the tweet while allowing LSTM to learn the relationships between words. We used SubwordTextEncoder from Tensorflow Datasets to achieve it.

For Linear Regression, we used bags of words. From previous experiments, we saw the improvement from using bigram is minimal. We used TF-IDF (Term Frequency–Inverse Document Frequency) of each unigram to build a vector of TF-IDF where each entry in the vector is the TF-IDF of a word, and train in a linear regression model.

2.2.2 S&P 500 Tick Data Processing

First step of data processing is to clean up the millisecond trade and quote data from NYSE database. Since fluctuations within a second is negligible, we use the first trading price of that second to be the price of that second. Some sample data look like below:

```
[[datetime.datetime(2015, 1, 2, 12, 32, 19, 805000, tzinfo=<DstTzInfo 'Ame
rica/New_York' EST-1 day, 19:00:00 STD>)
  205.12]
 [datetime.datetime(2015, 1, 2, 12, 32, 20, 853000, tzinfo=<DstTzInfo 'Ame
rica/New_York' EST-1 day, 19:00:00 STD>)
  205.16]
```

In the second step, we perform some processing on the tweets and pair them up with the corresponding market movement. We removed all URLs, punctuations, @'s, and empty tweets after removing these contents, which yields 13174 tweets. For the market reaction, we select the market price of the second just before the tweet as the "baseline", and the first price after a certain time frame to calculate the fluctuation. Thus, during market closed, we consider the price of the previous day as the baseline and use opening price of the next day to calculate the fluctuation.

$$Fluctuation = \left(P_{(T+10)} - P_T\right)/P_T$$

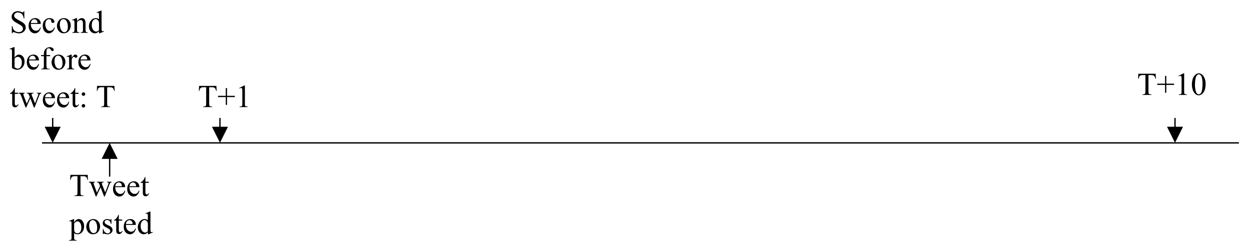


Figure 2.2 Illustration of time window with 1 and 10 seconds

Some training samples look like below:

```
['i love it when trump calls out msm for what they really are cams startin
g to show crowds at rallies',
 'i love mexico but not the unfair trade deals that the us so stupidly mak
es with them really bad for us jobs only good for mexico']
```

While their corresponding labels look like below:

```
[0.0003175013817189546,
 -0.0001645742229745455]
```

3 Models

We choose popularity predictor as our baseline model. The popularity predictor simply predicts the most popular choice, usually "rise". The idea behind choosing popularity predictor as the baseline is that in a chaotic system like the stock market, a popularity predictor would be hard enough to beat.

We first implemented LSTM (Long Short-Term Memory), a type of RNN (Recurrent Neural Network). We built a vector of words using all of Trump's Tweets.

The first layer is an embedding layer. This layer allows the model to preprocess the words into a smaller vector. It can also deduce some understanding of the words and feed those understandings into LSTM. We trained our own embedding layer to fit Trump's language.

The second layer we used is LSTM. This layer allows the model to understand the context of the tweet, namely, understanding the relationship between a sequence of words. This layer is crucial in understanding the tweet.

The third layer we used is a Dense layer (or Fully Connected layer). This layer allows the model to transform the output of LSTM into the predicted labels (the change in S&P 500) we want. It is connected to the last output from LSTM.

With these layers as foundations, we also experimented with adding more LSTM layer, having more Dense layer, and different sizes of LSTM and Dense.

We also implemented a linear regression model that predicts the fluctuation.

$$X\Theta = y$$

where X is the vector of words, and y is the fluctuation.

During our implementation, we are constantly monitoring the results to ensure that we are not overfitting and not generating a non-trivial predictor. Common error-measuring methods such as balanced error rate is not suitable for our problem, and the measurement of MSE is not a good choice to compare the error rate across the two models. We monitor the number of correct predictions in different buckets to ensure that it is not shifter to one side of the balance.

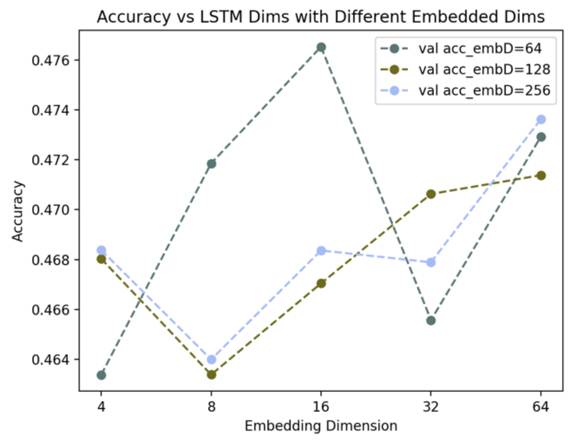
4 Results

4.1 LSTM

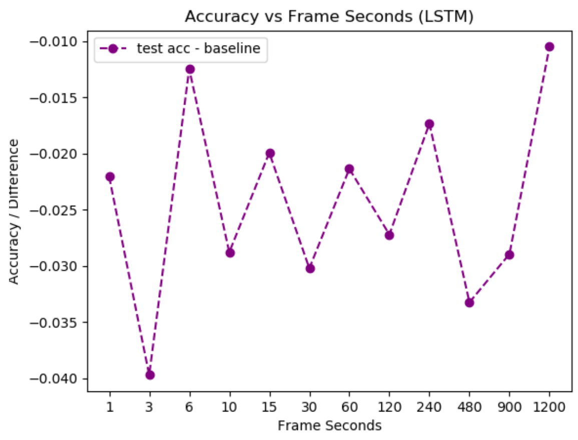
We build and tested the LSTM model on a variety of parameters.

(a) By running the model on different embedding dimensions of 64, 128, and 256, we found that there is little variation between 128 and 256, but the variation between 64 and 128 is significant. This can be due to that a dimension of 64 is not large enough to understand the relationship between Trump's words and a dimension of 128 is large enough to understand the relationship between Trump's words.

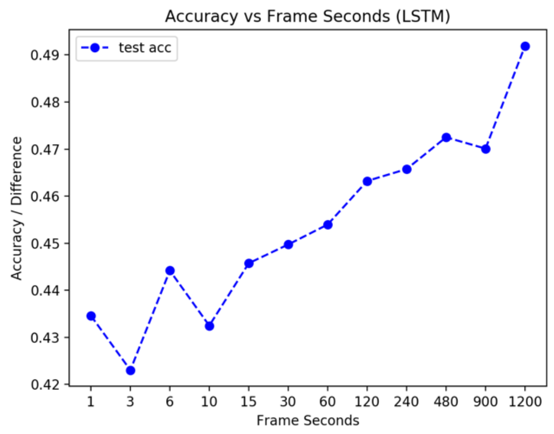
(b) By running the model on different LSTM dimensions of 4, 8, 16, 32, 64, we found that there are minimal variations on the accuracy. What we can inference from that is that the tweets from Trump is not significant in his wording. The context of his sentence does not matter. The significance is in the words he used. It can also be that tweets are meant to be short and punctuate, this level of context is minimal and is not worth the LSTM to learn.



(a) Accuracy vs LSTM dimensions vs imbedded dimensions



(b) Accuracy against baseline vs different frame seconds



(c) Accuracy vs different time fames

Figure 4.1 LSTM Model accuracy on different parameters. In figure (a), the x-axis corresponds to different embedding dimension values. There are minimal differences across different embedding dimension values. The different lines correspond to different embedding dimension values. With value=128 and 256, the differences are negligible. In figure (b) and figure (c), the absolute value of accuracy increases with time, but the relative accuracy against baseline does not show a clear trend, and is always below zero, i.e., the model does not outperform the baseline.

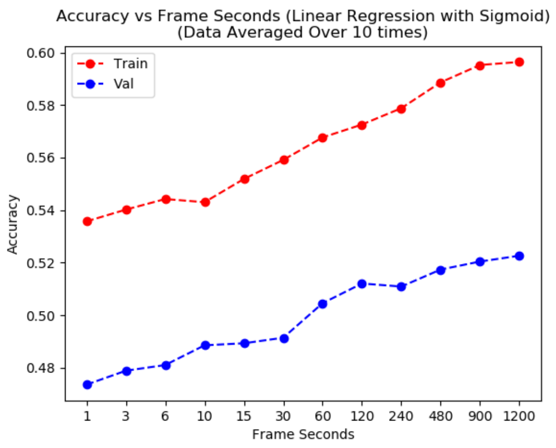
4.2 Linear Regression

To improve the accuracy, we build a linear regression model for sentimental analysis. The model tries to predict the price of the ETF after a certain time frame. There are several parameters that we've tested, including alpha, which is how the data should be regularized; different time frames; with or without sigmoid. After analyzing the results, we found the following facts:

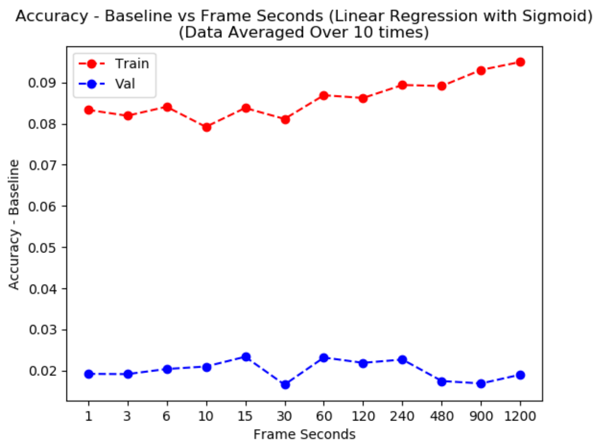
(a) With increasing time frame, the absolute value of volatility increases. This is because as time increases the market should be more diverted from its original starting point. But the relative value of how the model outperforms the baseline does not necessarily increases. And it decreases as the time window becomes very long. This is also reasonable since the longer the time frame, the more uncertainty in the market and thus harder to predict.

(b) The train accuracy always decreases as alpha increases. The validation accuracy first increase, and after reaching a maximum at alpha=10, then decreases. This is true for both absolute value of accuracy and relative accuracy against the baseline. At a very low alpha, the model overfits and makes the train accuracy significantly higher than the validation accuracy. At a very high alpha, the model underfits and makes both accuracy lower.

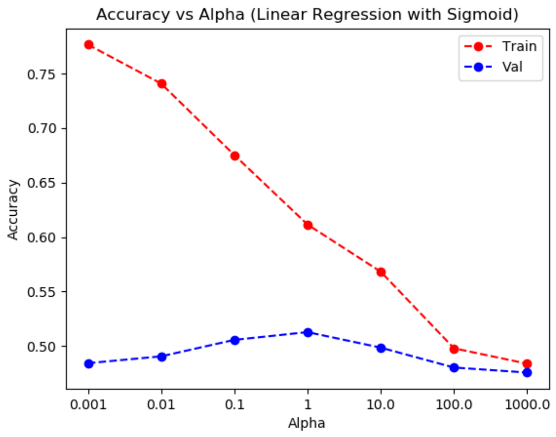
(c) The adding of sigmoid does not cause a significant change in accuracy. This is reasonable as the value of volatility does not shift towards one side of the balance; thus, it is not necessary to balance the data again.



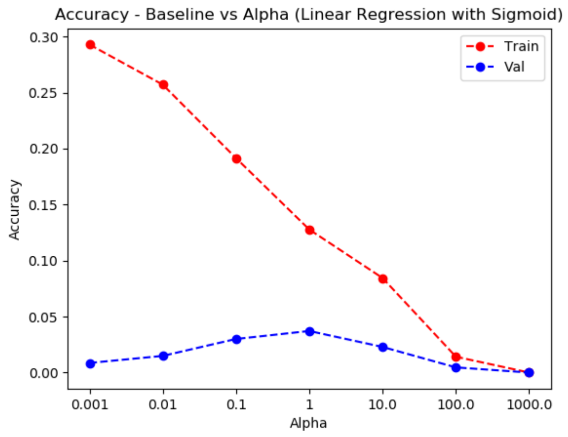
(a)The accuracy vs time window



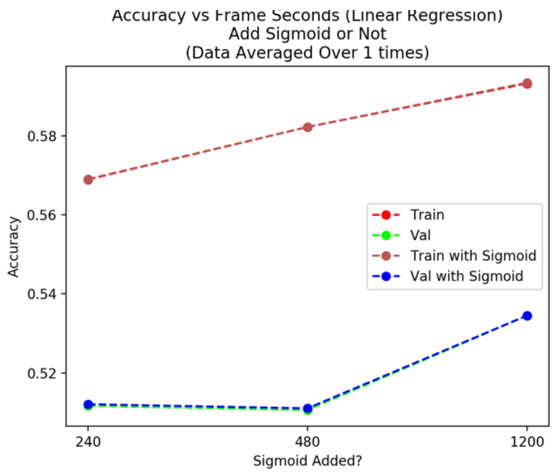
(b) The accuracy against baseline vs time window



(c) The accuracy of different alphas



(d) The accuracy against baseline of different alphas



(e) The accuracy with or without sigmoid

Figure 4.2: The accuracy on the linear regression model with different parameters.

4.3 Discussion

By comparing different prediction results across different models, we found that linear regression has the best performance. The model can outperform the baseline by more than 2%. This is reasonable since few words have significant impact on the stock market, such as “tax”, “job”, and “interest rate”. However, limited by the number of text samples, it is hard to get a better result.

The LSTM model, which should work better for non-consecutive inputs, did not perform well when the time window is short. The reason behind this is probably because it is too easy to overfit, and with the parameters that does not overfit, it won’t produce decent results.

There are several issues that prevent us from further improving the accuracy. The most serious issue is the problem of overfitting. Limited by the length of tweets as well as the number of tweets we have, it is hard to run a multiclass classifier without overfitting. Second problem is the problem of bid-ask spread, which is the difference in trading price since the bid price and ask price has to differ by at least 1 cent. But we weren’t able to find a way that would incorporate the bid-ask spread into both models. Third problem is that for real life applications, we should penalize wrong labels to achieve higher profit. But that would also lead to the problem of overfitting. Forth problem is about evaluating the importance and influence of a tweet. For instance, “Happy New Year” is much less important than “Nice talk with President Xi”, but that would require more text mining in potentially the context of the tweets.

5 Literatures

There are limited literatures about this topic. The research by Tong Yang and Yuxin Yang at Stanford University tries to solve a similar problem. They run the tests on a higher granularity, from one minute to twenty minutes. They reached an accuracy of 48.2% using LSTM with a time window of 8 minutes. We replicated their methods and achieved a higher accuracy of 50.2%. They used Naïve Bayes classifier as their baseline, which they considered their method a great improvement over the baseline. However, we found that there are more percentage of rises in the data points as the time window increases. At 8 minutes, around 49% of labels are rise, which means that an accuracy of 48.2% is not enough to beat the popularity predictor. We think that they have not chosen a strong enough baseline to compare with. It is also not persuasive enough to directly compare the accuracy across different time window since the underlying data are different.

It is also worth noting that they improve their model accuracy by cutting the tweets to a maximum length of 30 words. But they did not give a reasonable explanation for this action. After cutting the tweets, we did see an increase in accuracy of more than 1%. But we did not take that into account when comparing our models.

There are also other literatures about this topic that analyze daily data instead of tick data. Most of them utilize similar approach such as RNN or Semantic analysis, and they are mostly around an accuracy of 50%. These results are common around stock market analysis since stock market as a chaotic system is hard to predict.

6 Conclusion

The Trump tweet and stock market dataset is both difficult and fun to work with. The dataset is not too small to cause overfitting, but yet not large enough to perform more interesting tricks. The stock market, as a chaotic system, is difficult to predict, yet Trump's tweets are strong enough for us to outperform the baseline by 2%.

Through machine learning, we build a model that could predict the movement of the stock market using Trump's tweets and could potentially generate profits. But Trump's tweets are not enough for a multiclass classifier. We plan to include breaking news as well as tweets from other key political figures into the model. There are also other aspects of the market, like trade volume and Volatility Index (CBOE: VIX), which would be fun to deal with.

References

[1] Trump Twitter Archive. Trump Twitter Archive, http://www.trumptwitterarchive.com

[2] Millisecond Trade and Quote of SPY. NYSE Trade and Quote, Wharton Research Data Services, https://wrds-www.wharton.upenn.edu

[3] Yang, Tong and Yuxin Yang. Predict Effect of President Trump's Tweets on Stock Market Movements. http://cs229.stanford.edu/proj2017/final-posters/5140843.pdf

[4] TensorFlow LSTM API. TensorFlow, https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM

[5] Sci-Kit Learn Ridge Regression API. Sci-Kit Learn, https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html