Trading Tips from Trump's Twitter

Yeonjoo Jung and Paul Petit
W266: Natural Language Processing and Deep Learning | Fall 2019
UC Berkeley School of Information
{jungy06, paulrpetit}@ischool.berkeley.edu

Abstract

President Donald Trump's unprecedented use of Twitter, especially to publicize his praise for or concern about the economy, has generated significant attention from market analysts. This project walks through a practical testing and application of two Named Entity Recognition (NER) models, 3 sentiment analysis models, and a basic trading strategy to gauge the usefulness of buying or selling the stock of Amazon, Google, and Facebook according to sentiment of Trump's tweets mentioning the companies. We find that, despite modest precision in the entity recognition task, strong recall allows us to recognize 82% of the organizations in our test set. Our best sentiment analysis model, a CNN, scored an 81% accuracy rate. Applying our trading strategy, we netted a 5% and 2% gain on Amazon and Google stock, respectively, and lost 3% on Facebook trading Trump's sentiment as compared with a buy-and-hold strategy.

1 Introduction

President Trump has picked up his iPhone 11.2 times per day on average since his inauguration to share his musings or promote his agenda on the popular social media platform, Twitter. One topic he's keen on is the economy, and his comments about it and their impact on investment has garnered significant attention from media outlets and analysts alike (google "trump twitter stock market" to see).

While much of the focus of the business media has sought to understand the impact of Trump's tweets on the stock market at large[1], the more granular question about the impact of Trump's tweets on the stock price of an entity he directly names has generated less investigation. Given that factors affecting the price of a single stock on a given day, while certainly still numerous, are fewer than those affecting the market at large, we hypothesize that there may be unrealized opportunity in trading the stocks of specific companies that Trump names according to the sentiment of the tweets in which he names them.

In an effort to test this hypothesis, we gathered all of Trump's tweets since his inauguration, chose the *Stanford Named Entity Recognition* algorithm to identify companies named in his tweets, and built a convolutional neural network sentiment model to execute a simple retroactive trading strategy with the three companies that President Trump most often named: Amazon, Google, and Facebook.

Compared to the baseline portfolios holding same shares of stock constantly without any trading, we found that two of the three experiment portfolios trading on Trump tweet sentiment earned higher profit than the baseline portfolios. The profit ranged from 2 to 5% of the final portfolio values. One of the three companies lost its portfolio value by trading on sentiment by about 3%.

2 Approach

To outline our approach, we'll start with an end-to-end example of a tweet that we'd intend to trade trigger a trade. We'll then discuss the data and algorithms we used to automate this process

[1] (Soergel)

2.1 An example without automation

On August 19, 2019 at 8:52 AM, President Trump tweeted, "Wow, Report Just Out! Google manipulated from 2.6 million to 16 million votes for Hillary Clinton in 2016 Election! This was put out by a Clinton supporter, not a Trump Supporter! Google should be sued. My victory was even bigger than thought! @JudicialWatch." He named the company "Google," and his sentiment was negative. According to our hypothesis, this tweet might inspire concern from President Trump's 67 million Twitter followers, moving them to sell Google stock in the wake of the angry tweet.

The closing price of Google stock just after this tweet was posted was $1,198. The closing price of the stock the next day was $1,182. To evaluate our hypothesis, we compare the difference in the value of two portfolios: (S^T) one that sold 100% of this stock at the day's closing price and then bought it again at the closing price the following day and (S^C)one holding the same share of stock without any trading over the same period.

For this example, without considering fees and taxes related to transactions, the difference in the value of S^T and S^C is $15,760(1%).

The goal of this project is to automate each part of the process outlined, from identification of the corporation and sentiment in the tweet to the triggering of a trade dependent on the timestamp the tweet was sent. We evaluated the effectiveness of our trading strategy by aggregating and comparing the resulting performance of S^T and S^C each time Trump named a corporation in a tweet between 1/20/2017 and 10/25/2019 during his presidency.

2.2 Datasets

Our primary dataset consists of President Trump's tweets. We learned that Twitter's API restricts queries to the most recent 3,200 tweets of a specific user. Luckily, the Trump Twitter Archive[2] does not. So, we downloaded all tweets from the handle @realDonaldTrump to analyze.

[2](Brown)

The datasets we used for training will be addressed in the Models section of the paper.

Data on stock prices are from Yahoo Finance. For the sake of simplicity, we constrained the granularity of our price data to daily opening and closing prices for three stocks: Amazon, Google, and Facebook.

2.3 Data preparation

Twitter data is notoriously noisy with substandard grammar, misspellings, emojis, user handle references, and hyperlinks throughout. Unfortunately for data cleanliness, this is especially true of Trump's tweeting style. To clean the tweets in our Trump tweet dataset, we followed the example of Ritter et al. (2011) and largely applied the same cleaning strategy for named entity recognition and sentiment analysis, removing stop words, including special characters and hyperlinks.

In order to properly test our models, we also annotated our own test set of 150 Trump tweets. To do so, we deliberately sampled the full set of tweets for 76 tweets that named at least one organization—an effort to somewhat balance the distribution of entity classes in the tokenized test set— that we'd want our entity recognition models to flag. We then added another 74 tweets without entities and assessed the sentiment (positive, negative, or neutral) of the all of the tweets. To test the entity recognition models, we created a tokenized version of this test set with two classes ("ORG", "NO_ORG") per token.

3 Models

3.1.1 Named Entity Recognition (NER)

For the NER portion of our task, we utilized two off-the-shelf NER algorithms, following the work of Ream (2017), that were featured in a Twitter NER competition hosted in 2016 by the Workshop on Noisy User-Generated Text (W-NUT) for Twitter NER: Stanford CoreNLP and spaCy. These models return by default 3 and 18 entity classes, respectively, but we modified our program to ignore all but a single class, "organization," the only relevant class for our NER task.

The Stanford CoreNLP algorithm is a general implementation of linear chain Conditional Random Field (CRF) sequence models[3]. The algorithm was trained on the CoNLL 2003 training data, Message Understanding Conference (MUC) 6 and 7 training sets, and Automatic Content Extraction (ACE) 2002 training data[4].

The spaCy algorithm encodes context-independent embedding vectors for words into context-sensitive matrices which are classified into predictions using long short-term memory (LSTM) convolutional neural network (CNN) units[5]. The algorithm was trained on the Onto Notes 5 training set[6].

A component of this analysis that we'll leave for future work is re-training both the Stanford CoreNLP and spaCy models on Twitter data exclusively. This may result in improved performance for our task given Twitter data's uniqueness, as highlighted in the Datasets section.

3.1.2 NER model results

Before we present the model performance table, it's worth reflecting on what would constitute a proper baseline model. For our NER task, we chose to use a model that categorized capitalized words as "ORG"s and non-capitalized words as "NO_ORG"s. The intuition behind the rule is that entities are likely to be capitalized if they're proper nouns.

The following table summarizes our NER model results on the test set of Trump tweets that we annotated:

	Precision	Recall	F1
Stanford CoreNLP	**0.327**	**0.823**	**0.469**
spaCy	0.190	0.300	0.232
capitalized words - baseline	0.123	0.930	0.217

Precision suffers across all models, which is unsurprising given President Trump's propensity for rampant capitalization, likely resulting in the models over-identifying organizations. The spaCy model seems to particularly suffer from the terse Twitter messages' uniqueness.

Recall, on the other hand, is fairly strong for Stanford CoreNLP, and most notably, the baseline model's recall score is the highest of all the models.

To understand this result, it's important to acknowledge that it's due in part to a Trumpian quirk and an accident: the quirk is that President Trump is excellent at capitalizing organizations' names—in fact, he's excellent at capitalizing words in general in his tweets—and the accident is that our annotated test set of Trump tweets has few examples of organizations that were not capitalized, likely because there aren't many.

These NER model results indicate that implementing a rule-based matching algorithm first to (1) cross-check capitalized words against a list of corporations, categorizing those that match properly as "ORG"s, and (2) implementing NER models for the remaining non-capitalized words to augment the set of entities identified in (1) would likely improve recall and precision.

For our trading analysis, we'll proceed with Stanford CoreNLP, which displayed the best F1 score on the NER task.

3.2.1 Sentiment analysis

Before we build our own sentiment model, we first examined an existing sentiment model

[3]("The Stanford Natural Language Processing Group")
[4]("The Stanford Natural Language Processing Group")
[5]("Models · SpaCy Models Documentation")
[6]("Models · SpaCy Models Documentation")

called TextBob in NLTK. The model was trained on a movie sentiment corpus, not with twitter data. To see general accuracy of the model, we applied the model to about 150 test Trump tweets. The model was 80% accurate, which served as a good baseline for further development.

Therefore, we decided to build our own sentiment model using corpus composed of training examples specifically generated by twitter.

3.2.1.1 Data

The two main twitter corpora we looked at for model building were Sentiment140[7] created by Stanford and Sanders twitter sentiment corpus[8].

We chose the Sentiment140 data set over the Sanders data set because of its size: 1.6M versus 5K records. The training data set in Sentiment140 has two sentiment classes (positive and negative) while the Sanders corpus has four (positive, negative, neutral, and irrelevant). However, two thirds of the Sanders text is labeled neutral or irrelevant, so we decided to build our model with Sentiment140 data.

One thing we noted with Sentiment140 is that the training data set had only positive and negative labels, but the test data set had positive, negative, and neutral labels. Since the test data set is small and contains labels not in the training set, we decided to use the training data set to train, validate, and test.

3.2.1.2 Models

We built three sentiment models. The baseline mode is a Naive Bayes model built with simple bag of words method. We also built a CNN and LSTM models to improve accuracy. 25% of the Sentiment140 training data set was used as test data set to assess model accuracy for all three models.

The CNN and LSTM models share many of the common features in the model building process including the data cleaning, embedding, and padding layers that utilize the max length and vocabulary size of the Sentiment140 corpus. Both of the models have 2 hidden layers and 1 dense layer with softmax function to predict 5-class classification. It's 5 class because the training data set returns sentiment in numbers, and it ranges from 0 to 4 (0 for negative, and 4 for positive sentiment).

CNN model has 2 hidden layers with kernel size 3 in both layers, and 32 filters in the first then 64 filters in the second layer. LSTM model uses first 64 then 32 hidden nodes. Dropout rates of 0.4 ~ 0.5 were added to avoid overfitting in each hidden layer.

3.2.2 Sentiment analysis model results

Our CNN model is the best performing model with highest test accuracy of 81%. LSTM model has close accuracy of 79%. The deep learning models have 3 ~ 5% accuracy improvement from the baseline Naive Bayes model.

Model	train	validation	test
N.B.			76%
CNN	**82%**	**81%**	**81%**
LSTM	78%	79%	79%

4 Application of models to Trump tweets

With our models trained, we applied them to the full set of President Trump's tweets. The output of these models was then tokenized such that classes are cut token-by-token, with sentiment classes consistent across tokens within the same tweet. By filtering, counting, and sorting for tweets classified as "ORGs" and pulling their timestamps, we distill a set of organizations commonly mentioned by Trump that need only be validated against a list of corporations in order to be candidates for application in our retroactive trading strategy to assess how lucrative it would have been to trade on Trump's sentiment.

Once the best sentiment model was trained and selected, we applied the model to all of Trump's

[7] ("For Academics - Sentiment140 - A Twitter Sentiment Analysis Tool")
[8] https://github.com/zfz/twitter_corpus

tweets since his inauguration in January 2017. We ran our NER model over the sentiment-labeled tweets and then tokenized the tweets. To narrow down the set of organizations we'd consider buying or selling, we counted distinct entities.

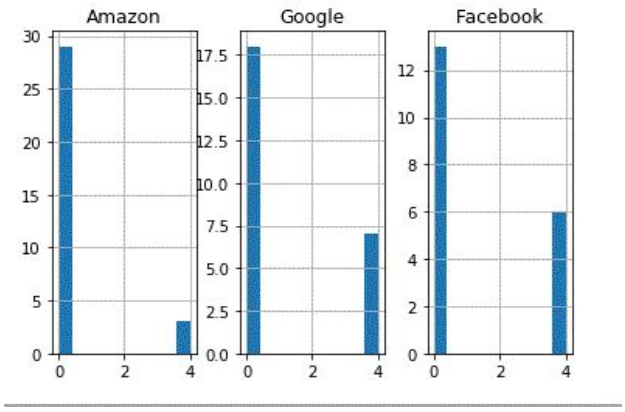
5 Stock Trading with Trump Sentiment

5.1 Identifying Company

Some of the identified entities were misclassified as organizations, and many others were federal or government agencies, which are obviously not corporations and are naturally excluded. Because of their strong political affiliations, we decided to exclude media organizations (ie. Fox, CNN, the New York Times, etc.) as well though they're mentioned frequently. This left us with 3 tech companies that are also frequently mentioned: Amazon, Google, and Facebook.

5.2 Initial EDA

Amazon, Google, and Facebook are mentioned 32, 25, and 19 times, respectively. Trump tweeted more negative comments than positive as seen in the chart below. Amazon received the highest ratio of negative tweets among the three.



5.3 Trading Strategy

To build an automated trading execution algorithm driven by tweet sentiment, we defined these trading rules and assumptions:

1) Even though a positive or negative sentiment may impact stock price immediately after the tweet is released, for simplicity's sake in our experiment, we assume that we trade daily without any intra-day trading.

2) For daily trading, we buy/sell at the last available closing price for the day when the sentiment was captured. We hold the state for a day and reverse the trade at the next day's close price.

3) For the sentiment captured on the day when the stock market is closed (weekend, holidays), we execute the trade on the next trading day's open price and reverse the trade at the next trading day's close price.

4) We assume that we buy stock with positive sentiment and sell with negative sentiment. We do nothing with neutral sentiment.

5) Definition of positive/neutral/negative sentiment when tweets are aggregated daily:

- positive: sentiment range from 2.5 to 4
- neutral: sentiment range from 1.5 to 2.5
- negative: sentiment range from 0 to 1.5

6) Even though in reality, very few people will trade like this, we execute 100% of total share and reverse 100% of the traded share to make our portfolio effect from the sentiment more visible with simpler calculations.

7) The daily value of the baseline portfolio is determined by the daily close price.

8) We do not consider trading fee or other expenses that may occur with stock transactions. We only compare nominal portfolio value by simply multiplying the share with the appropriate stock price for simplicity.

6 Results of Trading Strategy

The results from the automated trading execution is below. Amazon and Google stock trading performed better than our baseline portfolio by 2 to 5%. Facebook stock trading yielded 2% lower net value at the end of the test period.

		Amazon			
		Baseline	Experiment	NetTradeProfit	NetProfit%
begin	1/20/2017	808330	808330	0	
end	10/25/2019	1761329	1848120	86790	**4.70**

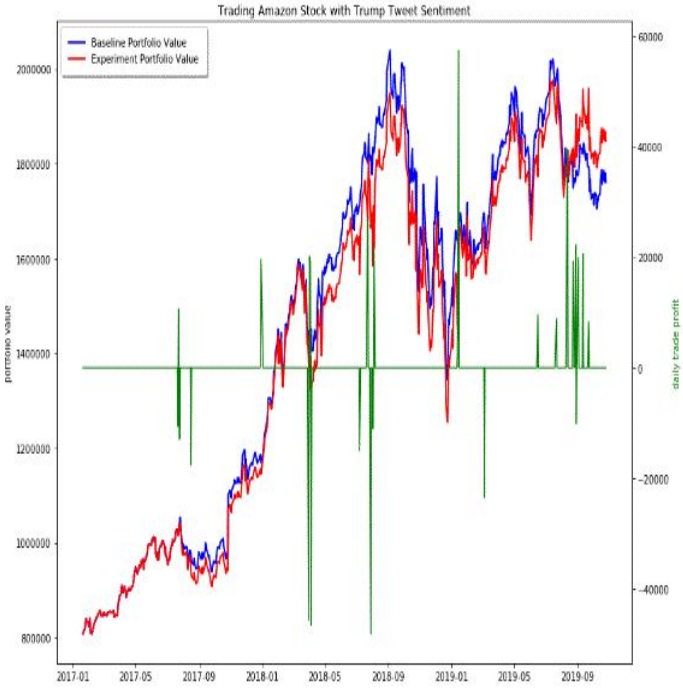
		Google			
		Baseline	Experiment	NetTradeProfit	NetProfit%
begin	1/20/2017	805020	805020	0	
end	10/25/2019	1265130	1287090	21960	**1.71**

		Facebook			
		Baseline	Experiment	NetTradeProfit	NetProfit%
begin	1/20/2017	127040	127040	0	
end	10/25/2019	187890	182930	-4960	-2.71

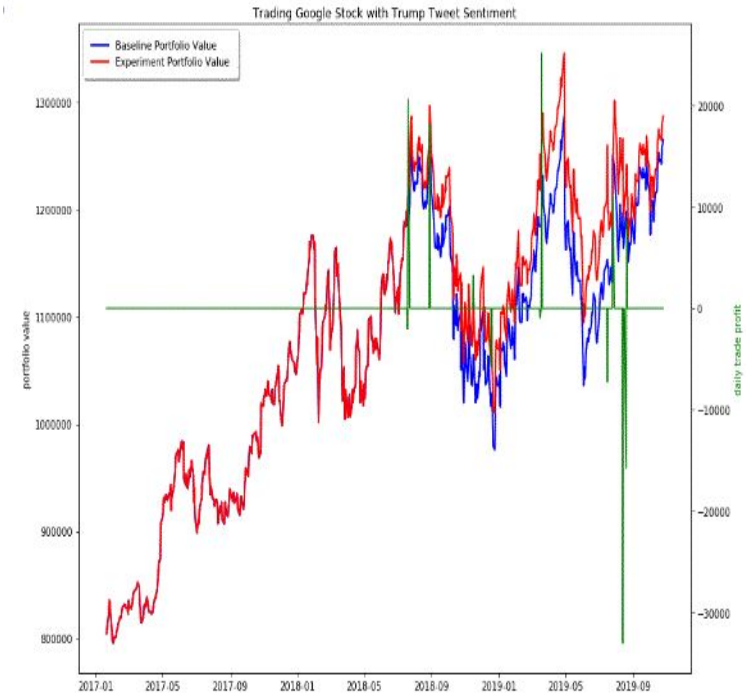
6.1 Amazon

The blue line represents our baseline portfolio value when we hold 1000 shares steady. The red line represents our trading portfolio value. The green line represents daily trading profit triggered by Trump's tweet sentiment.

Trump had more recent tweets in late 2019 , and trading on company sentiment turned our cumulative trading profit from negative to positive.

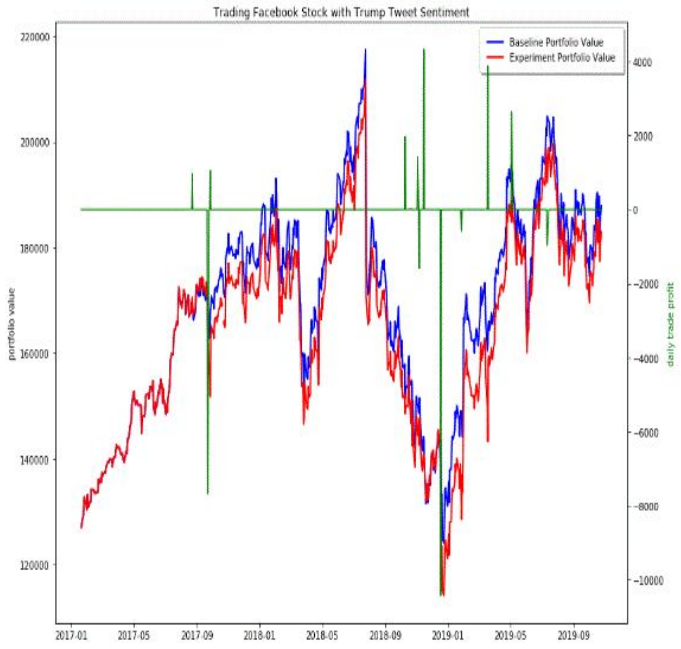


6.2 Google



Trading Google stock with Trump tweet sentiment was generally effective throughout the time. We had a couple of larger losses in late 2019, and narrowed the net profit to a smaller amount.

6.3 Facebook



Trading facebook stock on Trump tweet sentiment was overall less effective than trading the other two company's stock. However, the profit gain or loss scale is less due to generally lower stock price than the others.

With the three company's trading examples on tweet sentiment, the final net gain/loss of the stock trading over the years is not significant ranging from 3% loss to 5% gain. This is because by nature the transactional gain and loss offset each other over time. However, when we look at the daily trading gain or loss triggered by tweet sentiment, there is definitely an opportunity to utilize it to make short term profit (or loss) by the trade.

7 Conclusion

Do to the dust his Twitter antics have raised on larger market impact, we sought to investigate whether the sentiment of President Trump's tweets toward one company on Twitter would have an immediate impact on that company's stock price.

We identified 3 companies (Amazon, Google, Facebook) by applying our NER algorithm to Trump's tweet history since his inauguration. Then using a CNN, we built a model to infer the sentiment of tweets in which Trump mentioned these companies. Finally, based on the tweet sentiment, we retroactively executed trades of these company's stocks with a simple trading strategy (buy/sell based on tweet sentiment and reverse it in a day). The goal was to determine if we could profit from a series of short term trades of the company's stock instead of holding it over the same period.

Given our analysis was constrained to just three companies, we can only speculate about how our models would generalize. However, we saw that 2 of the 3 companies in our experiment had a final net value higher than our baseline portfolio's net worth. The net gain ranged from 3% to 5% of the latest portfolio value. Even though the trading gain or loss was not a massive percentage, there is some evidence to suggest that trading Trump's sentiment could be financially lucrative.

Works Cited

Brown, Brendan. “Trump Twitter Archive.” Trumptwitterarchive.Com, 2019, www.trumptwitterarchive.com/.

“Models · SpaCy Models Documentation.” Models, 2016, spacy.io/models. Accessed 8 Dec. 2019.

Ream, Shannon. “Named Entity Recognition for Twitter.” DigitalGlobe Blog, 2 Nov. 2017, blog.digitalglobe.com/developers/named-entity-recognition-for-twitter/. Accessed 8 Dec. 2019.

Ritter, Alan, et al. Named Entity Recognition in Tweets: An Experimental Study. 2011.

Soergel, Andrew. “Study: Trump’s Tweets Move the Markets.” US News & World Report, U.S. News & World Report, 2019, www.usnews.com/news/economy/articles/2019-10-08/study-trumps-tweets-move-the-investment-markets. Accessed 8 Dec. 2019.

“The Stanford Natural Language Processing Group.” Stanford.Edu, 2010, nlp.stanford.edu/software/CRF-NER.shtml. Accessed 8 Dec. 2019.

Alexander Pak, Patrick Paroubek, “Twitter as a Corpus for Sentiment Analysis and Opinion Mining” May, 2010

Shaunak Joshi, “Twitter Sentiment Analysis System”, Jun, 2018

Johan Bollen, Huina Mao, Xiao-Jun Zeng, “Twitter mood predicts the stock market”, Oct, 2010