Predict Effect of President Trump's Tweets on Stock Market Movements

TONG YANG, YUXIN YANG
tongy@stanford.edu, yuxiny@stanford.edu

Introduction

Background: Big high frequency finance corporates trade on Trump's Tweets in large quantities with algorithms, which can affect the market price. We want to understand the impact and implication of President Trump's Tweets on S&P 500 Movement, and predict how the market fluctuates after their reactions.

Goals: Given President Trump's Tweets (each tweet as a single data sample), here are our objectives:

- ❖ Predict stock market price change (rise/remain same/drop) **5** minutes after President Trump's Tweet release ;
- ❖ Understand the impact of time interval on prediction by comparing prediction accuracy of market price change **1-21** minutes after tweet release.

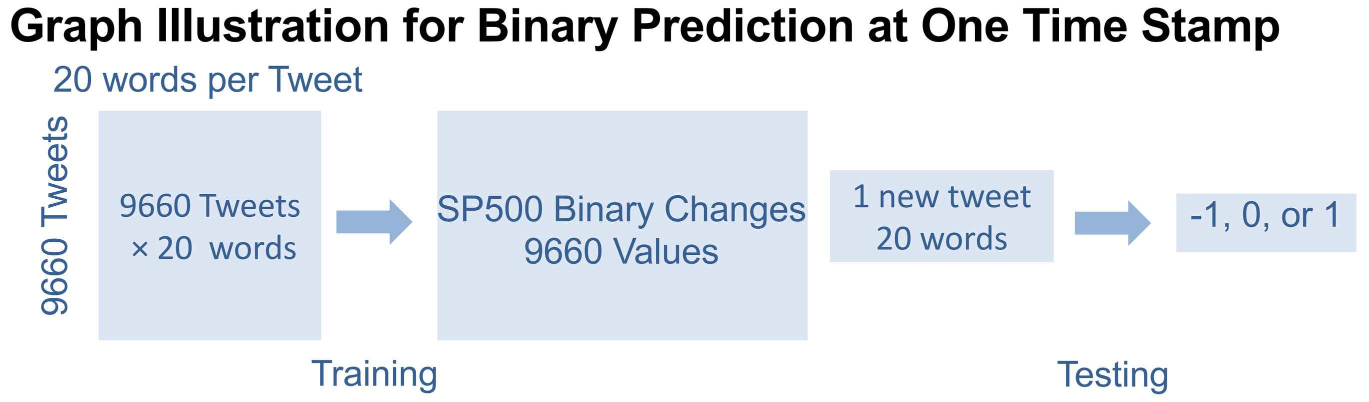
Results: Analyzing the problem using different models, here is a brief summary of the prediction accuracy:

Baseline Random Prediction:	**33.3%**	Naïve Bayes:	**42.5%**
SVM classifier:	**46.5%**	LSTM (RNN):	**48.2%**

Moreover, experimental result shows that longer time interval leads to higher prediction accuracy, which indicates that market price change make take some time after tweet release to take effect.

Problem Statement

Graph Illustration for Binary Prediction at One Time Stamp



Datasets

1. President Trump's Twitter Archive from 2009 till now:
http://www.trumptwitterarchive.com/
including 11,330 tweets with average length 20 words (**6 MB**)

2. Wharton Research Data Services (WRDS)
TAQ (Consolidated Trades) – SPY (S&P 500 ETF Trust)
470,159,618 trade entries containing raw SPY price. (**20 GB**)

3. GloVe: Global Vectors for Word Representation
We downloaded GloVe, a pre-trained model for word embedding. It contains matrices representations of words (400,000 words as 50, 100, 200 and 300 dimensions respectively).

Data Parsing

1. Millisecond Resolution S&P500 Data to Minute Resolution
In order to map the price change to the release timestamp of Trump's Twitter, we parsed our millisecond SP500 market data to minute resolution by taking the first reading of a particular minute. For minutes within which no trade happened, we applied forward fill.

2. Sentence Processing, Removal of Stop Words for Naïve Bayes and SVM
We used SKLearn NLTK library in Python for Words Pre-Processing. We transformed each word to lower case regular form, and removed punctuations and stop words with high frequency. A typical final sample is shown below: (Final Input Data for Baseline Models)


```
2017-10-20 11:31:00,"['report', 'united', 'kingdom', 'crime', 'rise', '13', 'annually', 'amid', 'spread', 'radical', 'islamic', 'terror',
'good', 'must', 'keep', 'america', 'safe']",0.1364216600732408,0.08185522932786317,0.1364216600732408,0.0779575131948468,0.155908949489345
```


3. Words Embedding and the Skip Gram Model for RNN
Trump's Tweets have ~9000 unique words. We downloaded pre-trained vector representation of words from GloVe, and converted each word from 9000 dimensional one hot representation to a 300 dimensional vector that captures the most information for that word to feed into RNN model.

Models and Algorithms

1. Naïve Bayes Classifier Model
Naïve Bayes classifier only considers each word in Trump's Tweet to contribute independently to the probability that the market goes up or down.

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots x_n \mid y)}{P(x_1, \ldots, x_n)}$$

Where x_i shows the existence of each word, and y represents classification result (up, down, or no change). We picked this model as the simplest baseline for our purpose to predict binary change.

2. Support Vector Machine Classifier Model with Non Linear Kernels
SVM model is an ideally more robust model than Naïve Bayes, for it can capture some inter-correlation between two words to some degree and optimize the decision boundaries. We eventually ended up using Poly Kernels observing it outperforms other kernels.

3. Recurrent Neural Network (LSTM) Classifier
LSTM would theoretically outperform, because it has the ability capture long term messages in a sentence. It can also remember "state" information that can capture some latent features hidden in a sentence. Our model used Softmax Cross Entropy between Logits and Labels as loss function.

Experiments and Toolsets

- ❖ We used Python **SKLearn** Naïve Bayes, and SVM library respectively to train our baselines. After fine tuning learning rate, loss, number of iterations and so on, we found optimized learning rate for SVM classifier using stochastic gradient descent to be 0.00001 and max iteration to be1000.
- ❖ We used Python **Tensorflow** Library to train our RNN LSTM model. Due to the limited size of dataset, we used batch size of 24 with 5 LSTM units for model training, and we decided not use a stacked LSTM model to prevent over-fitting.

Results and Analysis

Naïve Bayes: Accuracy on 2000 Test Data **42.5%**
SVM Classifier: Accuracy on 2000 Test Data **46.5%**

15.3715	division	-14.8110	imagination
14.0969	worst	-14.2299	james
13.9708	stupidly	-13.6232	reporter
13.6392	candidacy	-13.4340	puttster71
12.8453	jobless	-12.7844	diligence
12.5191	unaffordable	-12.5848	leibowitz
12.1724	96	-11.8904	rest
11.8494	lawrence	-11.8467	nhgop
11.8477	joelivan2	-11.6939	laura
11.7986	spectacle	-11.5271	understand
11.4673	disgusting	-11.3132	induct
11.4451	storm	-11.2103	refer
11.4271	misstetu	-10.8856	newspaper
11.4171	teacher	-10.5132	turnout
11.3710	bet22325450ste	-10.5052	skylerdeckard
11.1441	thehill	-10.4223	williebosshog
11.0000	brainpower	-10.4075	codyalliecats
10.9364	rave	-10.2830	core
10.8221	dude	-10.2624	brother
10.7374	2nd	-10.2522	ursulacurtin

14.3487	joenbc	-13.1766	division
12.8589	laura	-12.7904	accept
12.3803	leibowitz	-12.7456	arseniohall
12.2734	accrue	-12.7360	teacher
12.2476	lesleyclark	-11.9163	rolling
12.0598	politicalwire	-11.6819	disgusting
11.5967	impact	-10.9578	96
11.5886	skylerdeckard	-10.7496	thehill
11.4413	treated	-10.6875	idle
11.4013	wasted	-10.5661	drink
11.3998	divide	-10.4486	38000
11.2942	speculation	-10.2812	advice
11.2124	james	-10.1755	unique
11.1753	publicity	-10.1024	oct
11.0744	graphic	-10.0409	church
10.9842	ursulacurtin	-10.0392	assault
10.8414	watched	-10.0255	ryalsflair
10.8059	filing	-10.0247	20k
10.5621	slow	-9.9343	santa
10.5153	billingsley29	-9.9255	pleasurable

Fig. 1 SVM (a) Most informative words for negative labels (b) Most Informative words for positive labels.

Recurrent Neural Network (LSTM):
Our best LSTM model (30 words, 300 dimensional embedding) converged to have an final test accuracy of **48.2%**.

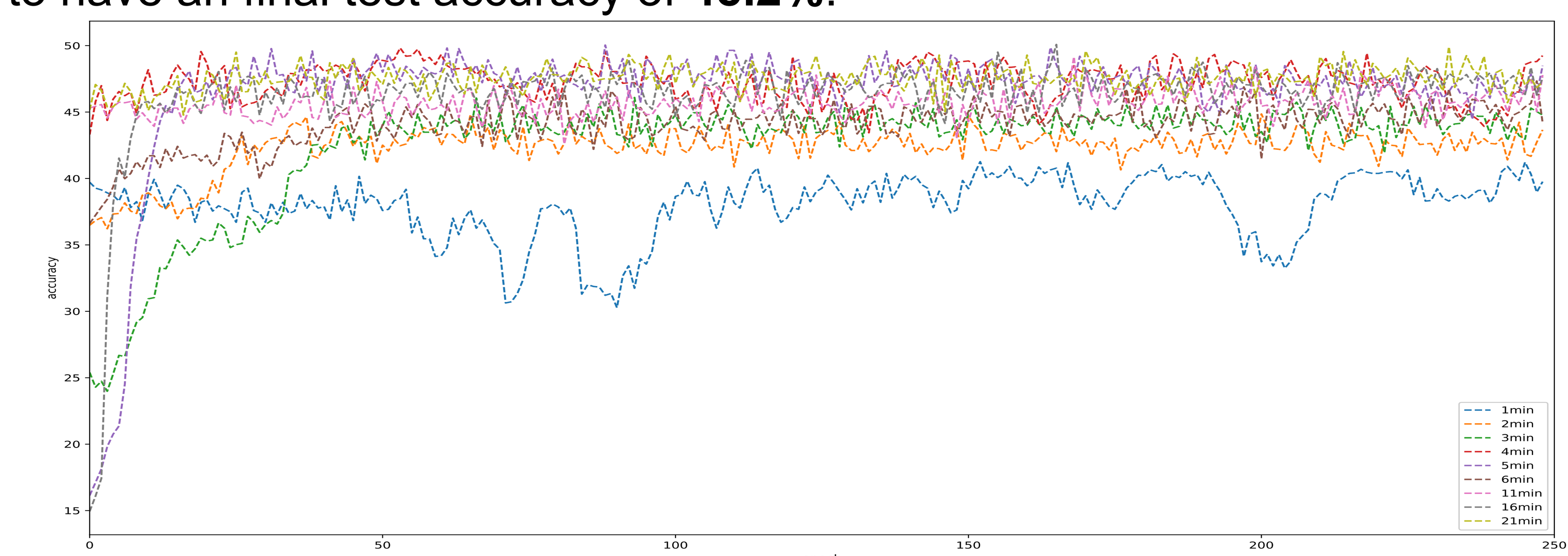


Fig. 2 LSTM Model Prediction Test Accuracy vs. number of Epochs, for different times interval

According to this plot, we observe that the longer time interval after the release, the better LSTM model perform in predicting binary outcome.

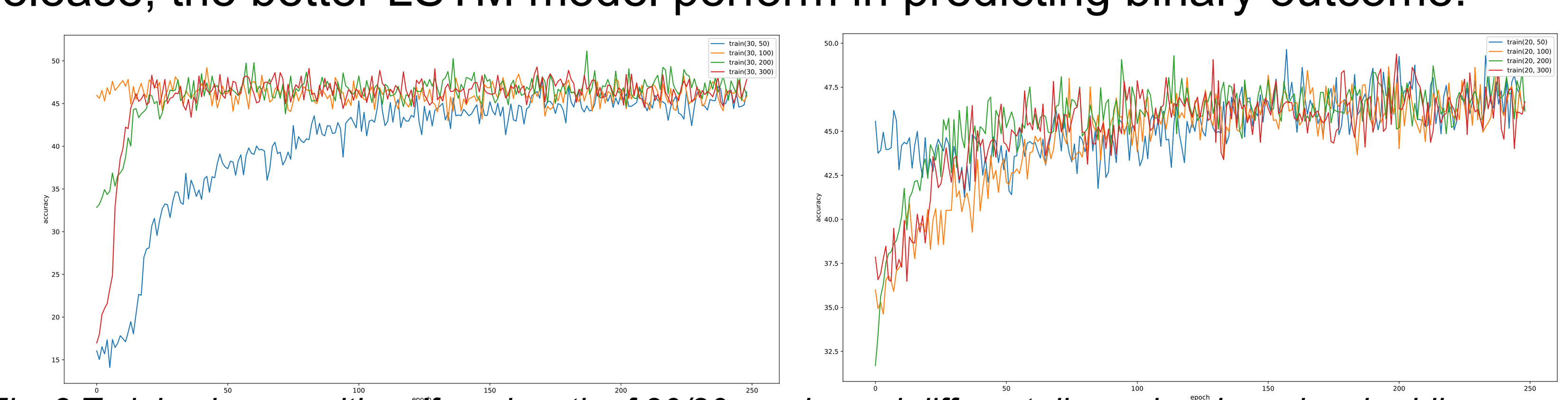


Fig. 3 Training losses with uniform length of 30/20 words, and different dimensional word embedding.

Conclusion & Future Work

Conclusion: All three models we used turns out to outperform the baseline random prediction, and LSTM is the out winning model because it models feature indicators and also performs semantic analysis, which can better capture tweets meaning. In addition, from the time interval testing, it seems that it takes some response time for market price to change after tweet release.

Future Work: 1. GPU training to finer tuned model; 2. Testing the robustness of model by testing on other news and Twitter releases.