Using Tweets for single stock price prediction

Hongshan Chu, Ye Tian, Hongyuan Yuan ***(By Alphabet)***

Abstract

Stock price, been studied for hundreds of years, is one of the most versatile thus hardly predictable things that is deeply rooted in the modern economy. With the trading frequency reaching sub-second and beyond, more advanced real-time stock price prediction tools would be highly demanded in addition to traditional financial analysis. In this work, we applied SVM and Naïve Bayes algorithms to this field and successfully built the connection between stock price and selected keywords frequency from Twitter Tweets. We firstly used TF-IDF method to filter out the candidates of keywords that is most relevant to stock price. Then based on these keywords, we systematically studied the testing error using cross validation for different number of keywords and unit time length. Finally, with the optimized keywords and unit time length, we plotted out the predicted stock price compared with the real stock price. We found that by using Gaussian kernel, SVM gives the lowest testing error, while Naïve Bayes catches pretty accurately the trend of stock price change over a longer period of time. Though limited by the available data size, the optimistic outcome opens a new approaching to real-time stock price prediction.

1. Introduction

Social media, as the collective form of individual opinions and emotions, has very profound though maybe subtle relationship with social events. This is particularly true when it comes to public Tweets and stock trading. In fact, research has shown that when it comes to financial decisions, people are significantly driven by emotions [1]. These emotions, together with people's opinions, are in real-time reflected by tweets. As a result, by analyzing relevant tweets using proper machine learning algorithms, one could grasp the public's sentiment as well as attitude towards the stock's price of interest, which could intuitively predict the next move of it.

Some previous work has been done to show that tweets can indeed reflect stock price change. Bollen. Etc (2010) randomly selected three months' tweets, and pointed out that, surprisingly, the 'Calmness' score of these tweets is able to resemble some of the key features on Dow Jones Industrial Average(DJIA) price change within the same period of time [2] (figure 1). Other work focused on one single stock and used particular person's tweets to predict that stock's price change. [3] However, there is yet any published work aimed to predict any single stock's price using machine learning through social media, like tweets.

In order to expand the scope of prediction from stock market index or some particular stock, we use keywords frequency (KF) of stock-relevant tweets to predict any single stock's price change. By assuming that these keywords are non-sensitive to any particular stock, we could in theory predict any stock's price change in real time. Note that Google has applied similar methodology for flu trend prediction using its gigantic search queries, and found out that certain query keywords are highly correlated with the current level of flu activity. This work has been publish on Nature [4]

In this work, we divided our selected tweets data and stock price data from the same period of time into different time slots featured by 'unit time length', and used SVM and Naïve Bayes algorithms to train the keywords frequency from time slot i and stock price percentage change from time slot i+1. To be exact, our work contains four main steps. 1. Data collection and wrangling from Twitter and NASDAQ official website. 2. Keywords selection by TF-IDF method.

3. Optimization of the number of keywords from 2, and unit time length through systematic study. 4. Testing prediction using both learning algorithms.

2. Data Collection and Wrangling

Twitter has collected gigantic amount of tweets. It is both impractical and unnecessary to even try to get all of them. Aside from technical barrier (in fact, twitter only opens the most recent one week's tweet data to public) most of the tweets are totally non-relevant to stock price, and are thus merely background noise. In order to increase the relevance to stock price, and overcome the lack of amount of available tweets, we fetched each weeks' tweets data filtered by stock symbol for 99 stocks with the most trading volume during the time of interest, using Python. By now, we have collected three weeks' tweets data, including 313030 tweets for analysis. For each tweets, we have its posted time in GMT, and the text content of the tweets, as shown on in figure 1 (a) and (b).
NASDAQ's official website has all the stocks' price. Though it provides historical data back years ago, like Twitter, only most recent one week has stocks' price posted for each trading minute. As a result, we fetched the stock price data with minute precision within the same period of time as tweets data. We then calculated the percentage change of each for each unit time slot. An example of 1day unit time length is given in figure 1 (d).

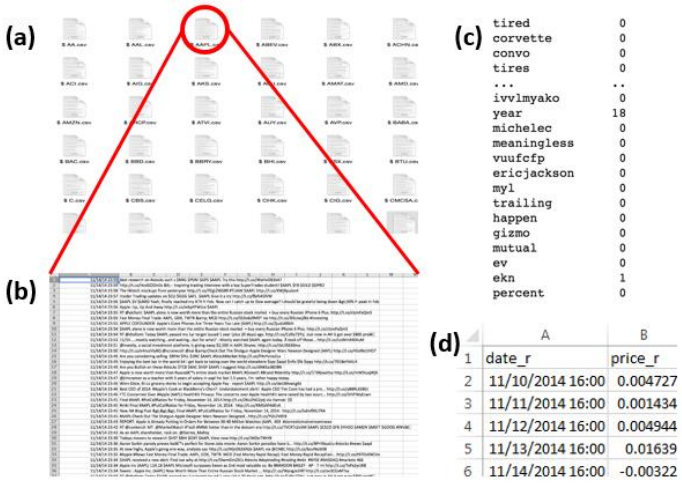

Figure 1. Tweets data and stock price data. (a) Tweets files searched by each stock's trading symbol. In total there are 99 files. (b) Demonstration of the raw tweets data with time column (in GMT) and tweets content column search by $AAPL. (c) Post-processing dictionary for $AAPL tweets on Nov. 13th. (d) Post processing stock price with 1day unit time length.

In order to find the most relevant feature words from tweets that can best predict the change of stock price, work that has been done is as below:

1. Create training sample by divided both the tweets data and stock price data by specified interval time.
2. For each tweets sample, convert the text into single words that's mentioned. Remove all the links, and none-English words.
3. Build a dictionary for all the mentioned words. Count the frequency of each word from each tweets sample.
4. For each stock price sample, take the difference between the start and end price for the given period of time.

3. TF-IDF for keywords selection

TF-IDF, short for term frequency-inverse document frequency, is a numerical statistics that reflect how important a word is to a document in a collection or corpus [5]. This method suits especially well to our purpose as our training tweets are purposely chosen that are related to stock price. By comparing our training tweets data with randomly chosen tweets, we could filter out those words that most likely connects to stock price.

To implement TF-IDF to find out the feature words, we searched the tweets that contain letter 'e' in their contents. And divided them equally to the size of the training sample. Then based on the same dictionary built from training sample, we created the random tweets' word frequency matrix to compare with training samples.
For a given feature size n, we denote the frequency of the word with i^{th} highest TF-IDF value as x_i, and use $\mathbf{x} = (x_1, x_2, \dots, x_n)$ as our feature.
To give a taste of these filtered out words, below are the top 20 words found with sample interval time of 15min, 30min, 60min, 90min, 120min, 180min, and 390min.

4. Results

4.1. SVM

We run Support Vector Regression with the feature we selected against the return of the corresponding stock for every time unit and single stocks.

Hypothesis	$f(x) =< w, x > +b$
Feature	x is the feature vector, the raw feature is the frequency of the selected words during corresponding unit time slot
Target	$f(x)$ is the stock's return in this unit time slot

The method do make sense as we can see the plot below suggest some time period our prediction is effective. In order to test the different combination of feature size and length of time unit we take into consideration we calculate the test error for all the cases and plot the heat map for better data visualization. (Figure 2(a))

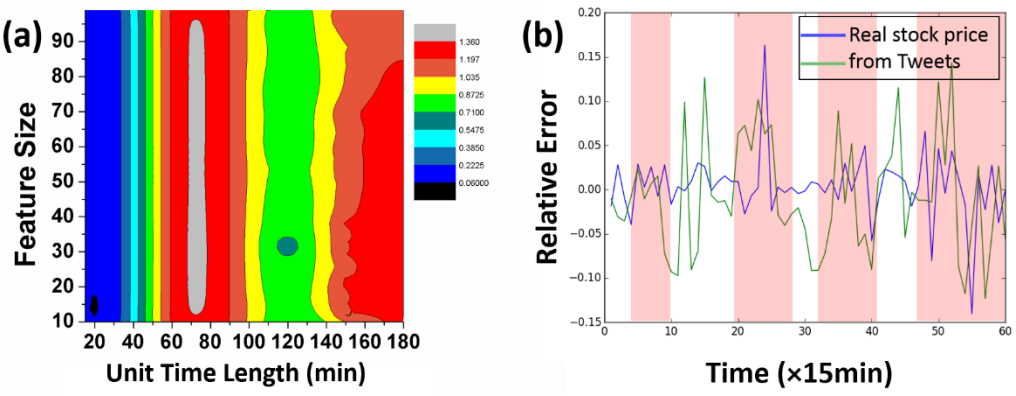

Figure2. SVM result. (a) Contour plot shown the testing error with different feature size and unit time length. (b) Comparison between SVM's testing result and real stock price. Highlighted area show great similarity

The test error was calculated via cross validation with 80% of all samples to train, and the remaining 20% to test. It should be pointed out that we have exclude some outlier (which take up about 5% of the raw data) to made our estimation more robust. Figure 2(a) indicates that the smaller the sample interval time is, the lower the error. Notably, when the sample interval time is about 15, the testing error reaches the minimum, less than 20%. The testing result with optimized unit time length and feature size is plotted as shown in figure 2(b)

4.1. Naïve Bayes

Naïve Bayes classifier is a powerful way to deal with text-concerning tasks. Now we adapt a similar strategy as spam classifier, as discussed below.

Hypothesis	$y = h(\mathbf{x})$
Feature	$\mathbf{x} = (x_1, x_2, \dots, x_n)$, where n is feature size. x_i is the frequency during corresponding unit time slot of the word with i^{th} highest TF-IDF value.
Target	y=I{stock price raises in the next time slot of x}

To find the optimized feature size and unit time length, we utilized cross validation and calculated the testing error. 75% of the data is used as the training set. The result is shown in figure3 (a)

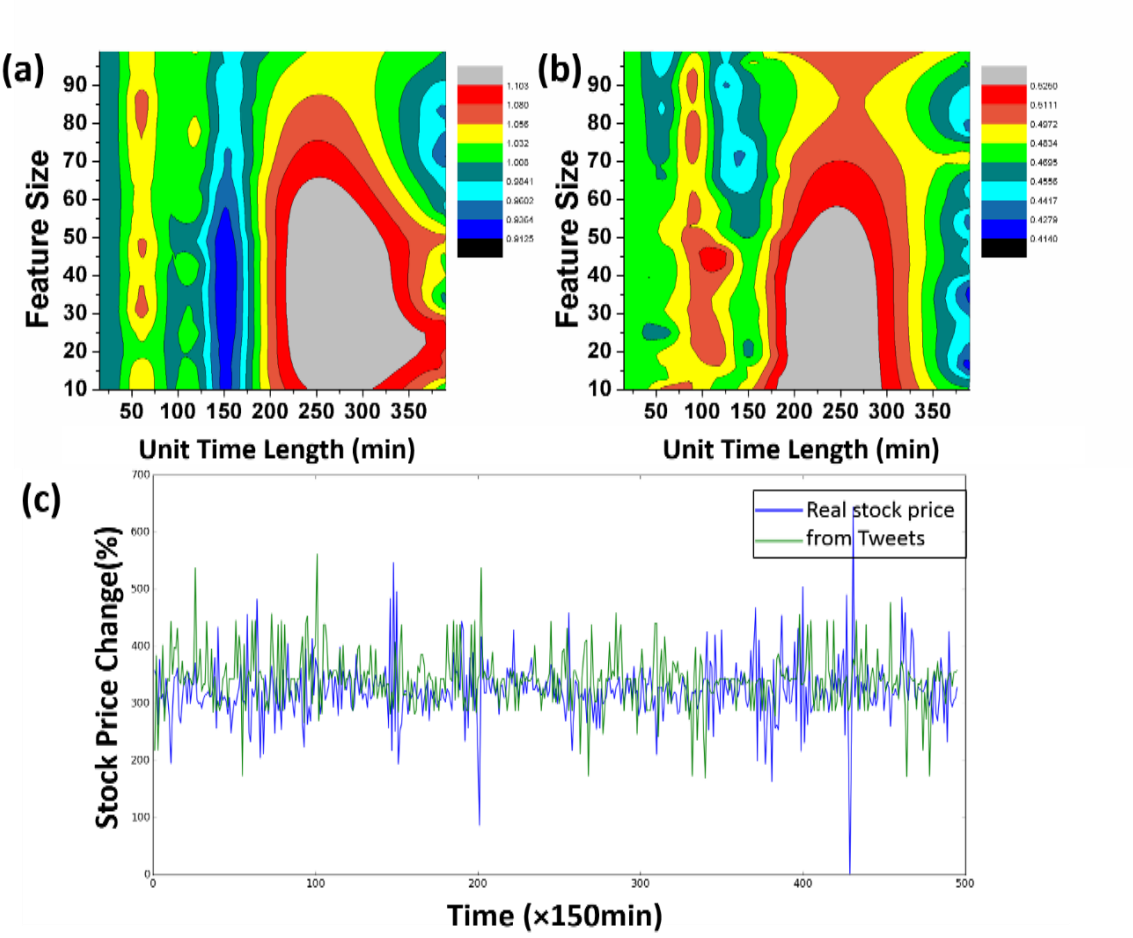

Figure 3. Naïve Bayes result. Contour plot shown the testing error with different feature size and unit time length for 1000 discrete stock price change (b) and 2 discrete stock price change (a). (c) Comparison between Naïve Bayes' testing result and real stock price.

We can see that with feature size=18 and unit time length=390 min, we can get highest accuracy. With these parameters, we utilized 3 weeks' data to train and predict. We trained the classifier using 90% samples of the 3 weeks and make prediction according to x corresponding to the rest 10% sample, and compare the predicted y with real y. The error is 45%.

Now we are going further: using Naïve Bayes to predict the stock price. We divided the return range of training set into several intervals (1000, for example) and devote these intervals as $\alpha_1, \alpha_2, \dots, \alpha_{1000}$. Then $y = \sum_{i=1}^{1000} I\{y \in \alpha_i\}$. Similarly, we do cross validation and draw the picture shown in figure 3 (b).

And we find that with feature size=30, unit time length=150 min, we can get highest accuracy. We again train the classifier with 90% samples of the 3 weeks and make prediction according to x corresponding to the rest 10% sample, the results is shown in figure3 (c)

5. Conclusion

This work has successfully demonstrated that using keywords frequency, SVM and Naïve Bayes algorithm can predict any single stock's price in real time. With our given data size, SVM gives the best test error, while Naive Bayes catches the general trend of stock change pretty well. For each of these two algorithm, the conclusion is discussed as below.

Due to limited data size, in the Support Vector Regression context, the testing error seems to be more sensitive to the length of time unit than feature size. This may be due to relatively even distributed word frequency for most, if not all, samples and the fact that we do not have enough tweets data to select the most effective feature. The words may not be effective themselves which result in insensitivity to feature size. In sum, with only three week's data, though there is for sure still quite some mis-fit, but this model does suggest some useful information.

For Naïve Bayes classifier, error is relatively high. However, regardless of the detailed features like local peaks, Naïve Bayes captured the general trend pretty well. It is because tweets is a delayed and smoothed version of public sentiment. For example, when news is released, not everyone receive the information immediately and simultaneously. Even if they receive the information, they will not necessarily talk about it or talk about it immediately on twitter.

6. Future work

To improve the predicting accuracy and decrease the testing error, future efforts should be devoted to the following four aspects.

First, data from longer period of time will be highly helpful. This is true not only because this can increase the sample size, but more fundamentally, as stock market shows periodic behavior switching between bullish and bearish, data from longer period of time would catch the public emotion and opinions in both market situations. Based on the first order estimation, half a year to a year's data would be preferable.

Second, better keywords selection algorithm is needed. This is maybe equally as important as or even more important than the first one. Though TF-IDF can tell us the stock-relevant keywords, it does not tell whether these words has anything to do with rising or dropping of stock price. As shown in figure 2, a certain amount of filtered keywords by TF-IDF seems to be neutral to stock price change. A better solution would be to look at the words frequency change for all the mentioned words from selected tweets when stock price rise, compared to those when the stock price flat out and drop. The different response of these words frequency would tell which words are relevant to each stock price behavior, very likely leading to better performance. Though this is a more accurate approach, it is doomed to have much longer run-time due to significant increase of computing work.

Third, because twitter is a delayed and smoothed version of public sentiment, we can try to predict y using x several time slots ago and combine x from several time slots. For example, we can use $x^{(i)}, x^{(i+2)}, x^{(i+3)}$ to predict $y^{(i+j)}$.

Last but not least, improvement on learning algorithm could be further explored. For example, we have only tried linear and Gaussian kernel for SVM. Some other kernel maybe more suitable to this problem.

Focusing on the above four aspects for improvement, this project can be continued developed into an online real-time stock price prediction tool. Data can be kept collected and stored in some online storage database like My SQL.

7. Reference

[1]. Nofsinger, J. (2005) Journal of Behaviour Finance. 6, 144–160.

[2]. Bollen, J., Mao, H. and Zeng, X.-J. 2010. Journal of Computational Science 2(1):1–8

[3]. Hu, Z. Jiao, J. Zhu, J. http://cs229.stanford.edu/projects2013.html

[4]. Ginsberg, J., Mohebbi, M.Patel, R., Brammer, L., Smolinski, M., Brilliant, L., Nature 457, 1012-1014 (2009)

[5]. Http://en.wikipedia.org/wiki/Tf%E2%80%93idf