Project Report: Super Stock Prediction

Yinghao Li, Xinyu Wang, Zhiqing Yang, Yiquan Li, André Haddad

1. Introduction

With the rise of machine learning technologies, predicting stock movements and automating investment decisions have been actively studied. The stock price is often determined by investors' behaviors, and the investors make their decisions based on public information to predict how the stock market will react.

Last year, there was a huge Gamestop "scandal" in which many hedge funds - based on legitimate market analysis - decided to short the company by selling options contracts. The Reddit community identified this and engaged people to buy the stock to drive an increase in prices, leading to huge losses in billions by many hedge funds. Celebrities like Elon Musk and other influential famous people suggested to their followers to also purchase shares of the company.

This phenomenon led us to take a closer look at how news articles and social media play a role in influencing the stock market. Previous studies have shown a correlation between the contents of financial news articles and stock prices [1]. In this project, we aim to predict market price movements using texts of news and chat communities. We hope to measure information and news value contributing to the market price. The quantitative financial industry can model these chatrooms and help them make better investment decisions.

2. Data

In this part, we will briefly introduce the data we use, the cleaning methods and the splitting methods.

Data description

We are using data from Reddit news on Reddit WorldNews Channel (worldnews), as well as tweets about top companies. The news data consists of the top 25 news headlines each day (ranked by users' votes), and the tweets data are about 5 companies of interest: Amazon, Apple, Google, Microsoft, and Tesla. We also have stock price information for each company, as well as Dow Jones Industrial Average (DJIA). Due to the running time limit, the time range of our analysis is 2015-01-01 to 2015-8-11. We plan to use this information to predict our target, which is whether the DJIA stock price will increase or decrease (1 for increase and 0 for decrease).

Data preprocessing

To clean and preprocess the data, we first remove HTML characters, ASCII in the tweets body and news headlines. Then transform all texts to lowercase. Next, we removed the punctuation and the stop words. Finally, we tokenized and lemmatized the text.

Data splitting

We split the days into 80% development data and 20% testing data, and the development data is split into 80% training and 20% validation. We then define a few parameters related to how we train the data, which are window_length and forward_predict_length. For now, we set window_length to be 7 and forward_predict_length to be 3, which means that we are using data from the previous 7 natural days (only data on trade days will be collected) to predict the price movement of 3 days later.

3. Models

For text embedding, we are using Doc2vec from the Gensim package. Sent2vec costs much more time to train, so we choose Doc2vec for embedding. Doc2vec is an extension to Word2vec, which represents each document of various lengths as a feature vector [2]. Since we have many tweets each day, which would take a long time to run, we randomly select 7 tweets per day in the training data. After obtaining feature vectors for news headlines and tweets for each day, we stack them together along with the stock price for the day. After flattening this matrix, we will have a feature vector for each day as well as a target variable of 0 (down) and 1 (up). We also normalize the vectors because the prices have higher values.

To predict the movement of the prices, we deploy three classification models, which are Random Forest (RF), Time Series Forest (TSF) and LSTM [4]. Random Forest often performs well on classification problems. Since our data is a time series, we choose TSF from the PYTS (A Python Package for Time Series Classification) Package. It takes time as a consideration in the training process and employs a new measure of entropy and distance gain to identify high-quality splits [3]. We choose LSTM because it is an advanced recurrent neural network that can learn order dependence in sequence, which is helpful for time series. It can connect previous information to the present task. We add a sigmoid function to get values between 0 and 1.

We use Grid Search to do hyperparameter tuning for RF and TSF, and use the accuracy rate to evaluate the model performances.

Results

Table 1. Accuracy for different models

Model	RF	TSF	LSTM
Accuracy	0.375	0.625	0.500

From Table 1, we can find that the accuracy of Random Forest on the test is only 0.375, which is even worse than a random classifier. LSTM only achieves an accuracy of 0.5. Time Series Forest achieves a much higher accuracy score of 0.625, which is our best result. The performances may be limited by the small training set and the large feature numbers. We also experiment on different parts of the features with LSTM (Table 2). We can find that it performs the best only with price information. Possibly because we are to predict the price movement, so it definitely contributes most to the performance. And we stack news and tweets together, which means we may absorb more noise into the model, especially for the tweets data. Sometimes, more data doesn't mean better results.

Table 2. Accuracy for different training features with LSTM

Feature Parts	News	Tweets	Prices
Accuracy	0.500	0.375	0.541

4. Conclusion

We use 1 NLP method and 3 classification methods in this project. We find that using Doc2vec along with TSF achieves the highest accuracy score. We can conclude that news and tweets do have a contribution to the movement of stocks, but there are still various different factors that can affect the stock market that are not reflected in this public information.

For future improvement, we can try different sets of parameters for our training dataset, such as time window and prediction days, to find out the time-sensitivity of the news and tweets. In other words, how long does it take for the stock market to react, and how long does the effect stay. Furthermore, we can also try TF-IDF for text embedding, because it takes a shorter amount of time to run compared to Doc2vec. This could be better if we have larger datasets. Finally, instead of directly using the text embedding as features, we may apply sentiment analysis on the text and use the extracted sentiment as the inputs of the models.

References

[1] Kaya, Mesut and Mine Elif Karsligil. “Stock price prediction using financial news articles.” 2010 2nd IEEE International Conference on Information and Financial Engineering (2010): 478-482.

[2] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14). JMLR.org.

[3] Houtao Deng, George Runger, Eugene Tuv, Martyanov Vladimir, A time series forest for classification and feature extraction, Information Sciences, Volume 239, 2013.

[4] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.