Prediction of Stock Trend with Media Sentiment Analysis

project-ID: 201812-30

Jiliang Ma (jm4750), Long Jiao (lj2463), Qinyuan Wei (qw2264)

Abstract—**In this paper, we apply sentiment analysis on twitter data and machine learning techniques to make prediction on stock prices. We use twitter data to analysis public sentiment on a certain company, and combine this result with the previous stock price to carry out the prediction about the company's future stock price. We implement three machine learning prediction methods: linear regression, random forest and multilayer perceptron (MLP). After the testing part, we find that MLP method has the best performance with MSE=0.071 and RMSE=0.27.**

Keywords-sentiment analysis; prediction; stock price; twitter

I. INTRODUCTION

Stock price changes every day. There are many factors may influence the price of stock price such as news releases on earnings and profits, introduction of a new product, or a change of management. From the prospect of behavioral economics, that the emotions and moods of individuals affect their decision-making process, thus, leading to a direct correlation between "public sentiment" and "market sentiment". So, if we know the "public sentiment", we can estimate the "market sentiment", and then predict the stock market.

Twitter, as known as social media, is a place where people talk publicly about their sentiment. With the now-a-days technology tools, we can do sentiment analysis based on the tweets text, extract people's sentiment about a certain thing. So, the problem we would like to solve is to build a model that can analyze the twitter data and try to predict stock price with the sentiment data.

II. RELATED WORKS

In "Stock Market Forecasting Using Machine Learning Algorithms", authors propose a prediction algorithm to predict the next-day stock trend using SVM and reach an accuracy of more than 70% [1]. Though the paper focuses on how to choose good indicators to predict stock price, it confirms that it is possible to predict stock price using machine learning method and receives a comparatively good result.

In "Stock prediction using twitter sentiment analysis", the authors aim to find the correlation between public sentiment and market sentiment using twitter data and finally reach accuracy more than 75% with the help of neural network [2]. This paper firstly proposes the idea of using sentiment as the indicator of model and truly promote the accuracy of prediction though the techniques adopted in the paper is somehow old. We will implement different kinds of techniques in order to improve the model's accuracy as well as interpretability.

In "Stock Price Forecasting via Sentiment Analysis on Twitter", the authors use SVM and naive Bayes to predict stock price using sentiment analysis on twitter data in 2016 and both methods reach a striking accuracy of about 80% [3]. Though the prediction level seems unreachable high, it is our goal to meet that level of accuracy by using different kinds of machine learning techniques.

III. SYSTEM OVERVIEW

Our project can be divided into 4 parts:

First, we do the data collection job with Hadoop. The dataset we use is the twitter7 dataset which is a collection of 476 million tweets collected between June to December in 2009. The size of this dataset is about 25GB and comes from Stanford Large Network Data Collection (SNAP) [4]. It includes 17,069,982 users, 476,553,560 tweets, 181,611,080 URLs, 49,293,684 Hashtags and 71,835,017 retweets. Each set of data is consisted of three parts: time, user and tweets. Since this dataset is large, we download it from SNAP dataset and save it in HDFS. We also download the stock prices data from Yahoo finance. In our project, we use Microsoft and Nasdaq stock prices from 2009/06/01 to 2009/12/31.

Then we use Scala to pre-process the data, which includes data cleaning and data filtering. Specific description is stated in the next part. The pre-processed data are saved as csv files.

After that we use python to do some data analysis work with the pre-processed data and stock prices data. The algorithms we use includes linear regression, random forest and multilayer perceptron (MLP). The aim of this step is to use machine learning techniques to predict the stock price based on the sentiment data and its history prices.

In the end, we use the techniques we learned in class to present our results. Furthermore, we use D3 and Node.js to present our analysis in a website, such as stock prediction, word clouds and so on.

IV. ALGORITHM

A. Data Pre-processing:

There are four steps in data pre-processing.

First, we load the data into HDFS. Then we use Scala to read it in spark as Figure 1 shows below.

```
T     2009-11-17 21:14:24
U     http://twitter.com/julianapanek
W     Today's weather is good
```

Figure 1. read data with Scala

Then we use Scala to parse and filter the data. We only keep the tweets and dates in each tweet data and filter all data by target trademarks, such as Microsoft and Xbox. The result of this step is shown in Figure 2.

```
scala> dfTweets.show
+--------+--------------------+----------+---------+
|    Date|               Tweet|Stock code|trademark|
+--------+--------------------+----------+---------+
|20090611|Microsoft Outlook...|      MSFT|Microsoft|
|20090611|#Java Interoperab...|      MSFT|Microsoft|
|20090611|"Morro" é o nome ...|      MSFT|Microsoft|
|20090611|@methedivine well...|      MSFT|     xbox|
```

Figure 2. Data parsing and filtering

Then we combine each data by its date, which means clustering the tweets from the same day into one set like Figure 3 displays.

	Date	Tweet
0	20090701	Yahoo CEO: We Have Nothing To Say About Micros...
1	20090702	Microsoft's \Pink\" smartphone to be Microsof...
2	20090703	RT @Bob_do: Microsoft Changing Users' Default...

Figure 3. Data combination

And then we merge the tweets with the stock prices. Since there are opening and closing days in stock market, we fill the blank data with the last day's price which is not in a closing day. After this step, we get the stock price data like Figure 4 shows.

	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Adj Close	Volume
2	2009/6/1	21	21.5	20.86	21.4	16.92261	57317100
3	2009/6/2	21.36	21.98	21.2	21.4	16.92261	48935700
4	2009/6/3	21.31	21.76	21.29	21.73	17.18356	56039600
5	2009/6/4	21.77	21.9	21.58	21.83	17.26263	42330000

Figure 4. Fill the blank data

As is displayed in Figure 5, we finish our data pre-processing work after the merging step.

	Date	Tweet	Prices
0	20091001	Very funny Microsoft! Now can I use the Windo...	24.88
1	20091002	Played against getonyourkneesJR and he had a ...	24.96
2	20091003	Comment on Windows 7 RC ISO Official Microsof...	24.96

Figure 5. Result of data merging

B. Sentiment Analysis

We use vader_lexicon in NLTK package in Python to execute the sentiment analysis, get the polarity for the Tweets of each day, which is how much positive, negative, neutral the Tweets are [6]. This algorithm can analysis the sentiment polarity of each word in the text, and add them up under specific rules, getting the sentiment polarity of the whole sentence or article. Figure 6 shows the result of sentiment analysis.

	Date	Tweet	Prices	Negative	Neutral	Positive
0	20090701	I only use my credit card online, recent trans...	24.040001	0.058	0.798	0.144
1	20090702	Microsoft's \Pink\" smartphone to be Microsof...	23.370001	0.042	0.842	0.116
2	20090703	RT @Bob_do: Microsoft Changing Users' Default...	23.370001	0.071	0.825	0.104
3	20090704	RT @arturogoga La publicidad de Microsoft, ca...	23.370001	0.038	0.811	0.151
4	20090705	Get Rich on Microsoft Search engine Bing http...	23.370001	0.044	0.836	0.119

Figure 6. Result of sentiment analysis

We also use Doc2Vec in Gensim package in Python to execute sentiment analysis to find the most related words in tweets data, and that result will be used in the data visualization part in our website using d3 [5].

C. Data processing and analysis

For data processing, we use the "sklearn" package in Python to perform machine learning. Specifically, we use linear regression, random forest and MLP to train the model [7]. As for parameters, we use polarity of tweets of each day, the close stock price of previous days, the close Nasdaq composite index of previous days.

1) Linear regression

Linear regression is one of the simplest machine learning models. In linear regression, the output is fitted by linear relationship with features, which is:

$$y = a_1 x_1 + a_2 x_2 + \cdots + a_m x_m + \varepsilon$$

where y is the output, x_i is the i_{th} feature of the data set, a_i is the the parameter of the i_{th} feature of the data set, and ε is the noise value.

2) Random Forest

Random forest is an integrated learning method for classification and regression that operated by building multiple decision tree when train the model and get the average output when perform the prediction. Random forest can overcome the over-fitting problem by create multiple

random decision trees, so it can promote the accuracy of decision tree.

3) MLP

Multilayer perceptron (MLP) is a type of artificial neural network. It has at least three layers of nodes: input layer, hidden layer and output layer. Figure 7 just shows an example of MLP model with one hidden layer. In our algorithm, we use 4 hidden layers to train the model. Each node exclude input nodes is a neuron that use a nonlinear activation function.

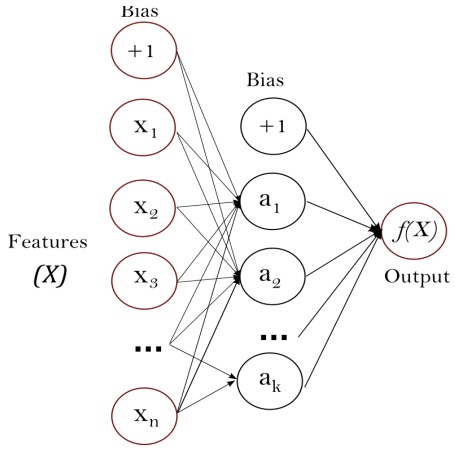

Figure 7. MLP model with one hidden layer

Using different techniques of tuning, we get our best model with least RMSE using parameters as 4 hidden layers with sizes are (60,60,60,60), max_iteration is 150000000 and solver is "sgd".

V. Software Package Description

A. Data filtering

The "data_filtering.spark" file contains all Scala codes used for data filtering. To execute this file, you need to start spark and spark shell first. After execution, you should have a new csv file that contain the filtered data in your path.

B. Data pre-processing and sentiment analysis

The "Data_pre-processing&Sentiment_analysis.ipynb" execute the data pre-processing and sentiment analysis. The result is shown in Figure 8.

	Comp	Date	Negative	Neutral	Positive	Prices	Tweet
1	1	20090612	0.076	0.829	0.095	23.33	[@CNETNews]EUは、 Microsoftのブラウザに移動反応 http://ti...
2	1	20090613	0.049	0.826	0.125	23.33	Intuit Did Not Kill MS Money. Microsoft Did. ...
3	1	20090614	0.07	0.81	0.119	23.33	DRINGEND: VAST contract - Leuke Microsoft .Ne...
4	1	20090615	0.058	0.856	0.085	23.42	XBOX CONSOLE FOR SALE - Microsoft xbox 360 sy...
5	0.9999	20090616	0.075	0.835	0.091	23.45	The comic Little Gamers on Microsoft's Natal:...
6	1	20090617	0.065	0.832	0.102	23.68	Vendo xbox 360 mas info aki... RT @chrispiril...
7	1	20090618	0.072	0.8	0.128	23.5	Microsoft makes gains with Bing, clashes with...
8	1	20090619	0.053	0.799	0.148	24.07	Microsoft's free anti-malware beta to arrive ...
9	1	20090620	0.05	0.792	0.158	24.07	Microsoft to revamp MSN, include stronger Bin...
10	1	20090621	0.064	0.786	0.15	24.07	if any one has xbox live message me with your...

Figure 8. Result of data pre-processing and sentiment analysis

C. Data processing

The "Data_Processing.ipynb" execute the data processing. In this file, we train the model with 3 different algorithms, you can implement any of the three algorithms as what it shows in Figure 9.

```
In [20]: from treeinterpreter import treeinterpreter as ti
         from sklearn.tree import DecisionTreeRegressor
         from sklearn.ensemble import RandomForestRegressor
         from sklearn.metrics import classification_report,confusion_matrix
         from sklearn.model_selection import train_test_split
         from sklearn.preprocessing import StandardScaler
         scaler = StandardScaler()
         from sklearn.neural_network import MLPClassifier,MLPRegressor
         from sklearn.linear_model import LinearRegression

         #try MLP method
         # scaler.fit(numpy_df_train)
         # numpy_df_train = scaler.transform(numpy_df_train)
         # numpy_df_test = scaler.transform(numpy_df_test)
         # mlp = MLPRegressor(hidden_layer_sizes=(60,60,60,60), max_iter=150000000, solver='sgd')
         # print(numpy_df_train[:,3:5])


         #mlp.fit(numpy_df_train, y_train)



         # mlp.fit(numpy_df_train[:,0:9], y_train)
         # prediction = mlp.predict(numpy_df_test[:,0:9])
         # print(numpy_df_test[:,3:5])

         #prediction = mlp.predict(numpy_df_test)

         # try Random Forest
         # rf = RandomForestRegressor()
         # rf.fit(numpy_df_train, y_train)

         # prediction, bias, contributions = ti.predict(rf, numpy_df_test)
         # print(prediction)

         #try Linear Regression
         linreg = LinearRegression()
         linreg.fit(numpy_df_train, y_train)
         prediction = linreg.predict(numpy_df_test)
```

Figure 9. Notebook of data processing

After running this file, we can get the plot of predict price with actual price, and the MSE and RMSE of the model.

D. Data visualization

The "dataclean4doc2vec.ipynb" does data clean on raw data and output processed data file. The "doc2vec_sentiment" makes sentiment on the processed data and hereby we get the most related words to the objective words.

Then we use "wordcloud.ipynb" to display the word cloud for each month to help users find keywords. Besides, with multiple files in the website folder, we implement data visualization using d3 to let user better understand the analysis result.

VI. Experiment Results

Since the algorithms we adopt are supervised learning methods, and we only have six months of data, even if it is already a big dataset, we choose to use the first 5 months of data as the training set and the last month of data as the test set. Figure 10, Figure 11 and Figure 12 shown below represent the prediction result of linear regression, random forest and MLP separately.

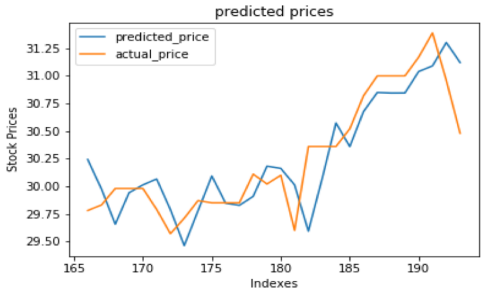

Figure 10. Prediction result of linear regression

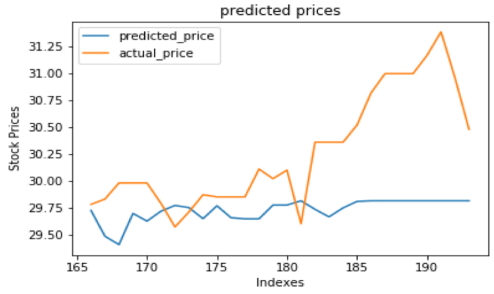

Figure 11. Prediction result of random forest

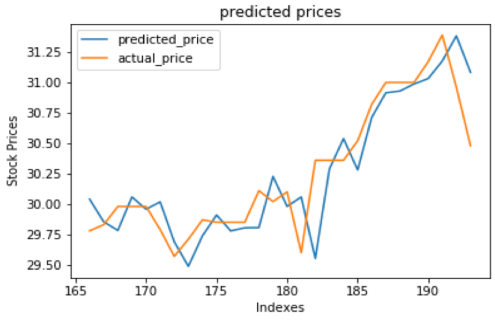

Figure 12. Prediction result of MLP

From the figures above, we can conclude that obviously MLP and linear regression have much better prediction accuracy than random forest does. Specifically, though the prediction seems good in linear regression result, the predicted line shows a pattern like imitating the last day's actual price, which in the real world may result in huge loss since the trend of stock price always changes abruptly. Comparatively, MLP prediction is closer to the actual price. In other words, though still has a pattern of delay, MLP's predicted price shows a pattern of catching up with the actual price better than linear regression does.

Though several related papers choose to use the proportion of accurately predict the trend as the criterion of a prediction model, here we do not use that as the criterion of evaluation, since stock investment is not simply defined by the qualitative trend of ups and downs but also includes how long should we hold the stock, fee of transaction, taxes and so on, which is very complicated to evaluate. So, in this project we choose to use Root Mean Square Error (RMSE) and Mean Square Error (MSE), two commonly used criteria of prediction model, as criteria to evaluate the prediction. Definitions of RMSE and MSE are shown as follows, and results of evaluation are shown in Table 1 and Table 2.

$$\mathrm{RMSD}(\mathbf{v},\mathbf{w}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\|v_i - w_i\|^2}$$

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

Table 1. The MSE result of three methods

	MSE
Linear Regression	0.082381695
Random Forest	0.5054099
MLP	0.071215228

Table 2. The RMSE result of three methods

	RMSE
Linear Regression	0.287022117
Random Forest	0.710921867
MLP	0.266861815

Result of evaluation in diagram is presented in Figure 13.

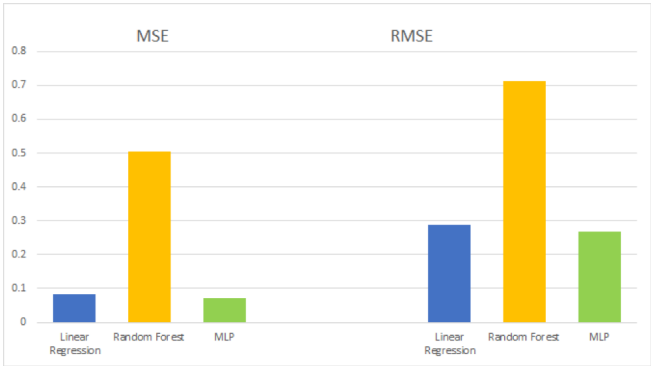

Figure 13. Evaluation of prediction using three models

Both linear regression and MLP have a comparatively low mean square error from the evaluation diagram. We can also conclude that MLP performs the best prediction in the three models which confirms the previous intuitive statement.

We also draw a word cloud and develop a website to present our analysis of Microsoft stock price as Figure 14, Figure 15, Figure 16, Figure 17 show.

Figure 14. Word cloud of Microsoft in December

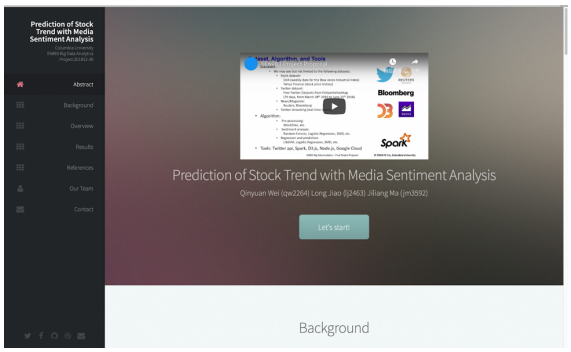

Figure 15. Website to display analysis

Figure 16. Candlestick chart of Microsoft stock price

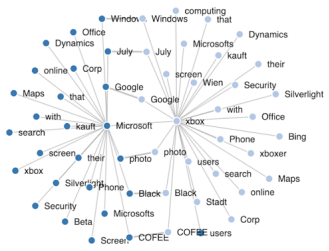

Figure 17. Most related words

Word cloud represents what people like to talk about Microsoft and its trademarks during a period of time. In the above figure we display the word cloud for December of Microsoft as an example. Then with candlestick chart, we can get the trend of Microsoft stock price in a direct way, and catch the important turning point in a easier way. We also draw the most related word using the result of Doc2vec sentiment analysis of tweets related to Microsoft. Here we take "Microsoft" and "Xbox" as example, note that nodes that connect to two topic words are important nodes that may represent important incidents [5].

With those plots we visualize, we can acquire what happen during the corresponding period of time, which will help us and potential user better understand how different factors influence the prediction model, for example how an event might cause changes in public sentiment and further causes fluctuation of stock price.

VII. CONCLUSION

In this project, we implement a sentiment analysis model using tweets data to predict stock price of a company. From the large raw dataset to website visualization, we put almost all the techniques we learn from this course to enhance the concept of "big data" project.

We firstly downloaded and processed the huge dataset （about 25 GB complete twitter data of 6 months in 2009) using Hadoop and HBase to accelerate our data access. Then with Scala, we finish data preprocessing quite efficiently, where we mainly do the data parsing work and data filtering work with the interested brand "Microsoft". On the other hand, we also implemented data merging and data combination in python with pandas for future use.

With the preprocessed data, we try different kinds of supervised learning methods including linear regression, random forest and MLP to get the model with best prediction. Using MSE and RMSE as criteria to evaluate prediction, we can conclude that MLP is the best model of three with very low mean square error.

To present our work more visualized and easier to get access, we also develop a website to display our analysis about a certain company's stock price.

For future work, there are three parts we will make more efforts on:

- Implement tests on more companies and markets.
- Use twitter API to get access to a large amount of data (limit of authority), and implement real-time analysis.
- Construct more criteria to evaluate our model in a more realistic way such as return rate when doing real-time analysis.

ACKNOWLEDGMENT

HERE WE WOULD EXPRESS OUR APPRECIATION FOR ALL THE SUPPORT AND HELP THAT PROFESSOR LIN AND ALL SELFLESS TAS OFFERED.

APPENDIX

Individual contribution:

Qinyuan Wei takes charge of data collection and preprocessing work, he also trains the linear regression model and implements NLTK sentiment analysis.

Jiliang Ma trains the MLP model and finishes part of the preprocessing work. He also implements Doc2vec sentiment，d3 visualization part in website and report compilation.

Long Jiao takes part in data preprocessing work including data merging and combination. Besides, he trains random forest model and builds the server and website.

REFERENCES

[1] Shunrong Shen, Haomiao Jiang and Tongda Zhang. Stock Market Forecasting Using Machine Learning Algorithms. 2012

[2] Anshul Mittal and Arpit Goel. Stock prediction using twitter sentiment analysis. 2012

[3] Kordonis John, Symeonidis Symeon, Arampatzis Avi, Stock Price Forecasting via Sentiment Analysis on Twitter. 2016

[4] 476 Million Twitter Tweets, J. Yang, J. Leskovec. Temporal Variation in Online Media. ACM International Conference on Web Search and Data Mining (WSDM '11), 2011. https://snap.stanford.edu/data/twitter7.html

[5] https://radimrehurek.com/gensim/models/doc2vec.html

[6] https://www.nltk.org/

[7] https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[8] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.