CS535 Term Project Proposal

Sentiment Analysis of Twitter Data
for Predicting Stock Price

Wei Xi: xiwei26@rams.colostate.edu
Xiang Li: ravno@rams.colostate.edu

Department of Computer Science
Colorado State Univerisy
Fort Collins, CO
2017-12-04

Section 1. Introduction

What is the problem we are trying to solve?

Stock price changes everyday. There are many factors that may influence the price of the stock such as news releases on earnings and profits, introduction of a new product, or a change of management. Many professional buy-side analysts can predict company's stock price and its trends. However, is it possible to let computers and big data do these kind of predictions based on some kinds of dataset? Twitter, as one of the largest social network services, "is a good source of information in the sense of snapshots of moods and feelings as well as for up-to-date events and current situation commenting."(Skuza). Every month, nearly 3.5 millions of users are active in twitter, they discussed about everything they saw and shared their opinions with others. According to Venkata, "Asur and Huberman have predicted box office collections for a movie prior to its release based on public sentiment related to movies, as expressed on Twitter" We believe those public opinions that were sent out by users' tweets is also deeply related to the change of market price. Therefore, the problem we would like to solve is to build a model that will analyze the twitter data and get the relation between the stock price change and people's tweets about that stock. According to Bing Liu, "Sentiment analysis, also called opinion mining, is the field of study that analyze people's opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes." Thus, our goal of this term project is to use big data and some machine learning approaches to build an application that will take a company's corresponding stock price and related tweets as inputs by running the sentiment analysis it will return the next day's stock price as long as its trends. For this project, we will mainly focus on the four leading technology company, that is Apple, Amazon, Intel and Microsoft. They not only build many hardwares which people are very familiar with such as iphone, macbook, xbox and processors but also provide online services such as office 365, AWS and icloud storage.

Why is this interesting as a Big Data problem? Who would use this if it was solved?

People used to think that stock price is unpredictable and do not follow any pattern. However, according to Jasmina, "recent research indicates that analysis of online texts such as blogs, web pages and social networks can be useful for predicting different economic trends" Everyone wants to minimize the risk and at the same time maximize the profit. In the era of big data, if we can find the relation between user's tweets and the stock price changing trend, we would take the lead in the financial market. As we all know, the key aspect of big data includes three Vs: Volume, Velocity and Variety. As Michael says in his paper, "Social media platforms aggregate constantly information about user activities and interactions" We have nearly 3.5 million active users commenting on everything every month. Though we are not doing real-time analysis, the data we are collecting still comes in at a high velocity and has a relatively large volume. Also, people comments on a large variety of different things. We need to consider it as a big data thing in order to get all the useful information we need.

If the problem is solved and shows that our model have a good accuracy in predicting the stock price, for the people who are interested in investigating some money on some stocks, it will benefit them a lot since they need to see if that stock is worth investigating or not. No one would like to purchase a stock that has bad reputation or has a trend of decreasing. Furthermore, in a company's view, the policy makers can treat this tool as a

reference or benchmark. They can decide when to release the new product to make sure they have the most benefits. In addition, it is always good to have different tools to know company's condition. For those buy-side stock analysts, it can be the tool to help them make the prediction more precisely and efficiently.

Section 2. Our Strategy to Solve the Problem

First, we decide to focus on four technology companies which are Apple, Microsoft, Intel and Amazon. Based on most of the research paper we've read, they just filter the tweets by the company name or stock code such as "AAPL" for Apple Inc.,but after consulting with our professor, we think that only filtering by company name or stock code may lose a lot of important messages. For example, people usually would discuss about their opinion on the newest released iphone rather than comment on Apple Inc. directly, so we went to the official page of those four companies and get their trademark list and created a trademark csv list file which has a format of (Stock code, trademark). That ensures us to have sufficient data for filtering purpose.

For the twitter data, we used 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) 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 as shown below and ideally we did some parsing the three attributes into one line instead of three. For the performance purpose, we only use 8GB which is one month of it.

```
T     2009-11-30 21:03:12
U     http://twitter.com/janeadarby
W     Gettin my good good perm!

T     2009-11-30 21:03:13
U     http://twitter.com/philuponem
W     @LovelyLittleLey lmaooo that's a compliment in my eyes thnx
```

Since the twitter data comes with different languages and it does not have the format we want, we put the dataset into HDFS and parse the data using spark and the build-in "newAPIHadoopRDD". It is quite convenient to read those kind of data and do some modification. We wrote a custom input reader so we can wrap every four lines. Then, we crossed join of the result dataframe with our trademark list.

This causes a little problem since originally we have around 500 trademarks so after union two table, the data became extremely large and due to the memory and time limitation, we decided to manually select some most popular trademarks to improve the performance.

Next step, we filtered by the trademark to see if we the tweets contain the trademark or not to decide if that row of data is needed or not. And then we ran the sentiment analysis on the filtered twitter data. The package we use is called Stanford NLP which is released by The Natural Language Processing Group at Stanford University. It looks at words in isolation, giving positive points for positive words and negative points for negative words and then summing up these values. In scala version of the package, 0 represents very negative, 1 represents negative, 2 represents neutral, 3 represents positive and 4 represents very positive. Our main purpose is to get the polarity of each tweets which shows how negative or positive this tweet content is. In addition, we somehow modified the function so that we only

that the value of the longest sentence instead of the value calculated by each words. Because we store all the data in dataframe in scala, we found it not easy to extract the value of a column. It onlys gives you a columnType but we want to have each row as a string in order to run the sentiment analysis. What we do now is to create a user defined function (UDF) which is a feature of Spark SQL to define new column-based functions. Finally, we group by the stock code and normalize the polarity value. In the end, we will have a data format as “Date, stock code, polarity value”

Our second dataset is the stock price data set which contains the stock price for Apple, Amazon, Intel and microsoft. We used a package called “panda” in python to get all the historical data we need and store them as a csv file. We collected all the stock price for those four companies in the year of 2009 from June to December. The size of data is relatively small. It’s will be about several megabytes. There are some restriction of data, the format is more like a table than queries and we did some parsing to have it meet our needs. The screenshot of part of the source dataset is attached below:

```
Date,Open,High,Low,Close,Adj Close,Volume
2009-06-01,19.495714,19.998571,19.428572,19.907143,17.84543,113124900
2009-06-02,19.855715,20.191429,19.764286,19.927143,17.863356,114055900
2009-06-03,20.0,20.158571,19.867144,20.135714,18.050327,141299900
2009-06-04,20.018572,20.597143,20.005714,20.534286,18.407618,137658500
2009-06-05,20.758572,20.914286,20.45857,20.667143,18.526722,158179000
2009-06-08,20.545713,20.604286,19.918571,20.549999,18.421707,232913100
```

And then, we did some modification of our stock price data to have a format of we want and after parsing, the modified version of this dataset now has of format like this:

```
Date,Close,Offset,Code
20090601,19.907143,,AAPL
20090602,19.927143,0.02,AAPL
20090603,20.135714,0.208571,AAPL
20090604,20.534286,0.398572,AAPL
20090605,20.667143,0.132857,AAPL
```

The four columns represent the date, the close market value, the offset(the percent increment or decrement of the close market value compared to previous day’s)

Final step will be combining this dataset with the previous output and implementing the linear regression in Apache Spark. The Spark MLlib library is very common for fitting models to data. We have used linear regression to fit the prediction model based on the stock price data and the related data. It usually takes a set of features x and a label y. In our case, we have the offset value of stocks as the label and the features include total comments and sentiment value of the stock. Since we have the linear function as y=mx+b, the model is trained until it converged on stable values for m and b.

The above is the offline part and our final version software will take a stock company name as an input from the user, and the output will be a graph that contains both our predicted stock price and the actual stock price for that company within the date of our twitter data.

Section 3. Functions of your software

Our software provides the user the functions of parsing the twitter data, filtering the data with trademarks, calculating the sentiment value for the four stocks everyday and finally providing a predict-price graph.

The first function in our program is to parse the twitter data. Before parsing, the format of the twitter data contains sets of three rows:

```
T     2009-11-18 21:14:24
U     http://twitter.com/emonemo92
W     iPhone is extremely good

T     2009-11-18 21:14:24
U     http://twitter.com/hollagraphics
W     I bought an iPhone

T     2009-11-18 21:14:24
U     http://twitter.com/imeaglesmith
W     Intel is extremely good

T     2009-11-18 21:14:24
U     http://twitter.com/isgangster
W     I bought an iPhone too

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

After running some mapreduce job, we got the data format like this:

```
20091118,iPhone is extremely good
20091118,I bought an iPhone
20091118,Intel is extremely good
20091118,I bought an iPhone too
20091118,Today's weather is good
```

The second function takes the previous data and did a cross join of the result with our trademark list which returns the data like this:

```
20091118,iPhone is extremely good,APPL,Mac
20091118,iPhone is extremely good,APPL,iPhone
…
20091118,Today's weather is good,APPL,Mac
20091118,Today's weather is good,APPL,iPhone
```

The third function filters the previous data to check if the tweets mentioned the trademarks or not. For example, we will keep the "`20091130,iPhone is extremely good,AAPL,iphone`" If the input is "`20091118,Today's weather is good,AAPL,iphone`" We would like to drop that row of data.

Our fourth function takes the previous input and calculate the sentiment value and append it to the previous data. So the output will look like this:

```
20091118,iPhone is extremely good,APPL,iPhone,4
20091118,I bought an iPhone,AAPL,iphone,2
20091118,Intel is extremely good,INTL,Intel,4
```

```
20091118,I bought an iPhone too,AAPL,iPhone,2
```

Then we take the previous data as input, and group them by date and stock code and update our sentiment value by normalizing it.

So then we will have the output like this:

```
20091118,AAPL,2.67
20091118,INTL,4
```

Next,we union the previous output with the stock price and offset on that date. So we will have

```
20091118,AAPL,2.67,16,0.83
20091118,INTL,4,4.56,-0.02
```

Finally, we make the offset as label, sentiment value ,offset value and the total mentions count as feature to run the linear regression. For example:

0.83,[2.67,16,2] and more for Apple and -0.02,[4,4.56,1] and more for Intel

We randomly select 80% of the previous data for training the linear regression model and 20% for test. We run the model on the testing data and so we will get the tuple of actual value and predicted value. In addition, we also stored number of training set and test set along with RMSE and R2 value in local for evaluating use.

Section 4. Testing proccess of your software

The final data we mentioned in the previous section is divided into training and testing data using a random split with 80% for training and 20 % reserved for testing.

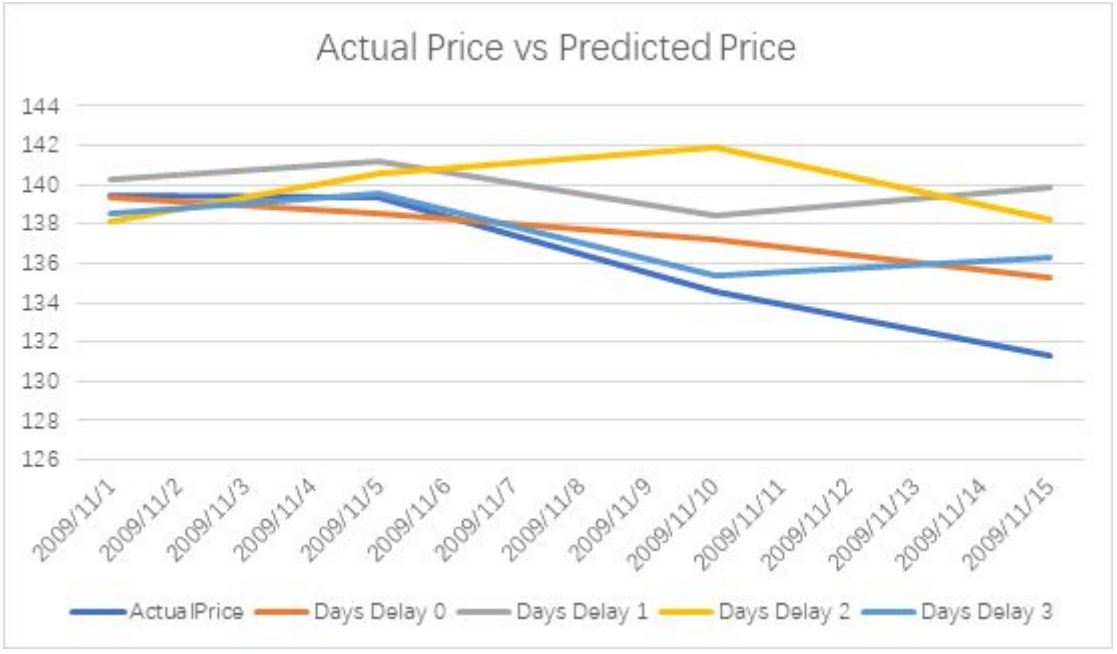
For the testing scenario, after we have generated the model using the training data, we run the model and make those testing data as our input and record for the predicted stock price. Then we will compare those predicted output with the actual stock price in the data. By using the root mean square error (RMSE), we can simply estimate the performance and accuracy of our model. Our initial target accuracy of the predicted output is within 10% and we will do that for several time to improve our model.

We will deploy our software in CS120 clusters. We will upload every data to HDFS and Spark in order to run the tasks successfully. In the data crawling and parsing phase for stock price and trademark list, we used our own laptops for convenient purpose.

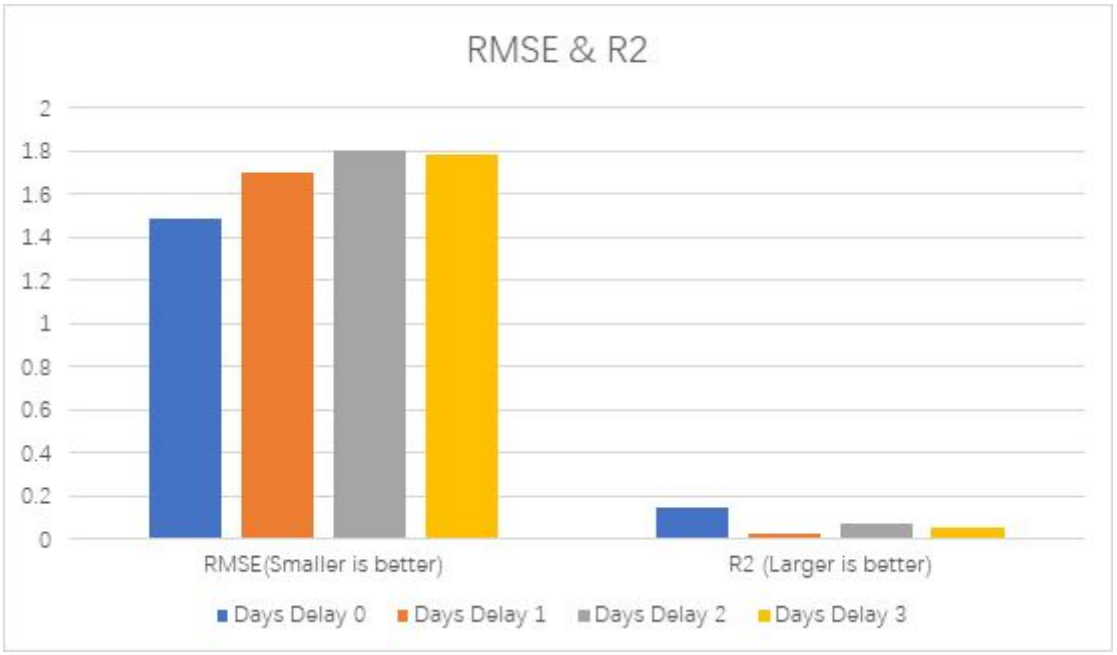
Section 5. Results and Evaluation

We choose the coefficient of determination(R2) and Root Mean Squared Error(RMSE) to evaluate our software. They measures the differences between values predicted by model estimator and the values actually observed. They are the most popular and accurate metric for linear models and provides the most intuitive view to judge if a model is good or not.The R2 usually have a value ranged from 0 to 1 where 0 represents the model does not improve the predict and 1 means the model is perfect for prediction. For RMSE, the smaller the value it, the better the model is.

Here is the table for SE and RMSE, along with a graph showing the actual stock price and our predicted stock price.



The dark blue line corresponds to actual stock prices and others line shows predicted price based on different days delay values of a stock in with 15 days. We can see from the graph that our model fits the actual stock price pretty well but with the days delay value as 0, it gives us the best result.



The above graph shows the RMSE and R2 value. Our RMSE is in a range of 1.4 to 1.8 which is pretty good but our R2 is near 0.2 which means that our model is not very ideal. We still found that we have the best result with days delay value of 0.

According to our R2 and RMSE and the result graph of prediction, it seems like that there is a weak correlation exist between the change in stock price of a company and the

public opinion of that company which are expressed by their twitter message. If the averaged polarity value of a company in one day is positive which means people are likely commending on that company or its product that day, there is a large possibility that its stock price would increase next day. However, there are still many improvement that we think is necessary for our project. First, due to the memory and time limitation, we have a small set of trademarks. We would like to find a more efficient and accurate way to have more trademarks and filter the data by those trademarks. Second, we found that the sentiment analysis in spark is relatively slow and it's not very accurate. We would like to change our methodology in the future. Last but not least, we are thinking to add more features in our training data like the open market price and the volume of the stock per day in order to get a more accurate model.

Section 6. Contribution

Xiang: Gathered twitter data and upload it to hadoop. Write the main scala program to parse the twitter data and build the linear regression model. Helped with writing the report and interpreting the results.

Wei: Collect the stock price data and trademark data. Help writing scala program. Writing the report and interpreting the results.

1. *Section 7. Bibliography*

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

Skuza, Micheal. Romanowski, Andrzej.(2015). *Sentiment Analysis of Twitter Data within Big Data Distributed Environment for Stock Prediction*
DOI: 10.15439/2015F230
http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7321604&tag=1

Pagolu, S.Venkata. Reddy, N.Kamal. Panda, Ganapati (2016). *Sentiment analysis of Twitter data for predicting stock market movements*
DOI: 10.1109/SCOPES.2016.7955659
http://ieeexplore.ieee.org/document/7955659/?part=1

Aich, Satyabrata. Kim, Hee-Cheol. Sain, Mangal.(2017) *Analyzing stock prioce changes using event related Twitter feeds.*
DOI: 10.23919/ICACT.2017.7890136
http://ieeexplore.ieee.org/document/7890136/

Twitter Sentiment Analysis Using Python
http://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/

Smailović J., Grčar M., Lavrač N., Žnidaršič M. (2013) *Predictive Sentiment Analysis of Tweets: A Stock Market Application*. In: Holzinger A., Pasi G. (eds) Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data. Lecture Notes in Computer Science, vol 7947. Springer, Berlin, Heidelberg
https://link.springer.com/content/pdf/10.1007%2F978-3-642-39146-0_8.pdf

Ranco G, Aleksovski D, Caldarelli G, Grčar M, Mozetič I (2015) The Effects of Twitter Sentiment on Stock Price Returns. PLOS ONE 10(9): e0138441.
DOI: 10.1371/journal.pone.0138441
http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0138441

Ballou, Kenny. Real Time Streaming with Apache Storm and Apache Kafka. ORPI.org
https://zdatainc.com/2014/07/real-time-streaming-apache-storm-apache-kafka/

B. Liu, Sentiment analysis and opinion mining, Morgan and Claypool publishers, 2012.
https://www.cs.uic.edu/~liub/FBS/SentimentAnalysis-and-OpinionMining.pdf