Sentiment Analysis of Twitter Feeds and Correlation with Actual Stock Prices

Project Report

CS 6350

Section: 0U1

Aug 08, 2016

Presented By

Anusha Kowdeed (axk150431)

Bharat Bhavsar (bmb140330)

Manaswi Reddy (mxk154430)

Sai Chakravarthy (sxa158530)

Shakti Shivaputra (sxs159231)

Table of Contents

1. Design

1.1. Problem Definition

Extract sentiment from real time twitter data and correlate it with the stock prices from the yahoo finance API. Train a classifier using the data from both sources to sketch the correlation between them and then use the same to predict future trends based on the sentiment identified at that time.

1.2. Description of Input data

We use data from tweets about a company to decipher the public opinion/sentiment towards a company. We then obtain actual stock prices and compare the price movement with the prediction made.

To extract a sentiment from text, we train a machine learning algorithm to make the prediction. A variety of training datasets are available to extract sentiment from text. In this project we use the twitter dataset for sentiment analysis. We use the Twitter4j API to extract both live and past tweets in text format. Using this API we define the query that we are searching for. In this case we are searching tweets related to company name/ company stock registered name.

Yahoo Finance, Google Finance, MSN Money and CNN Money are the most popular sources to obtain stock prices. Since Yahoo Finance is the most popular (in terms of the number of unique visitors) among the above mentioned sites, we choose Yahoo Finance as the data source to obtain stock prices. Yahoo Finance data is retrieved in the form of JSON and close and open prices are retrieved.

1.3. Algorithm and Pseudocode

The project can be divided into 2 main stages - creating the data set, obtaining the prediction.

The first stage involves collecting tweets for a particular company (e.g. “AAPL”) using the search API for Twitter for the past 7 days in intervals of 1 day. We also use the streaming API to collect live stream tweets for a few days. Next, we collect open and close prices for the same dates using Yahoo Finance API.

Once the two sets of data are created, we assign labels to the tweets based on the change in open and close prices for that day and consolidate the results in a single file. This file is uploaded on HDFS and is used as the input dataset for the next stage.

We use Spark MLlib to train a model and classify test tweets. The chosen classifier in this project is the Logistic Regression classifier. We first tokenize the tweets and split the tweets into a list of words. Using these words, we use hashing to obtain term frequencies for each

word. To avoid overfitting, we perform dimensionality reduction to reduce the number of features. We use pipelines to perform the classification.

Once the predictions are obtained, we output the final prediction based on the majority of predicted labels.

Pseudocode:

a. Download tweets in text format using twitter4j API.
b. Download finance data using yahoo finance API.
c. Create custom labels for tweets based on stock price changes observed from finance data.
   i. if(price increases) then label = 1 (positive sentiment)
   ii. if(price decreases) then label = 0 (negative sentiment)
d. Logistic Regression classifier using Spark MLlib
e. Train the algorithm with the dataset created above
f. Obtain predictions on the test data
g. Obtain label counts of predictions
   i. if (number of 0 label predictions > number of 1 label prediction) then predict "Price is expected to decrease"
   ii. else predict "Price is expected to increase"

Assumptions/ requirements:

To predict good result from trained model, we need to train the model with as much data as possible. This prediction model will produce better result as the train data increases. We have worked on available data from twitter4j API. This API allows to extract data from at most past 7 days. Therefore, as we add data to the train set, the accuracy is expected to increase compared to the initial outcome. Therefore, we periodically add past data and streaming data to train the model for better results.

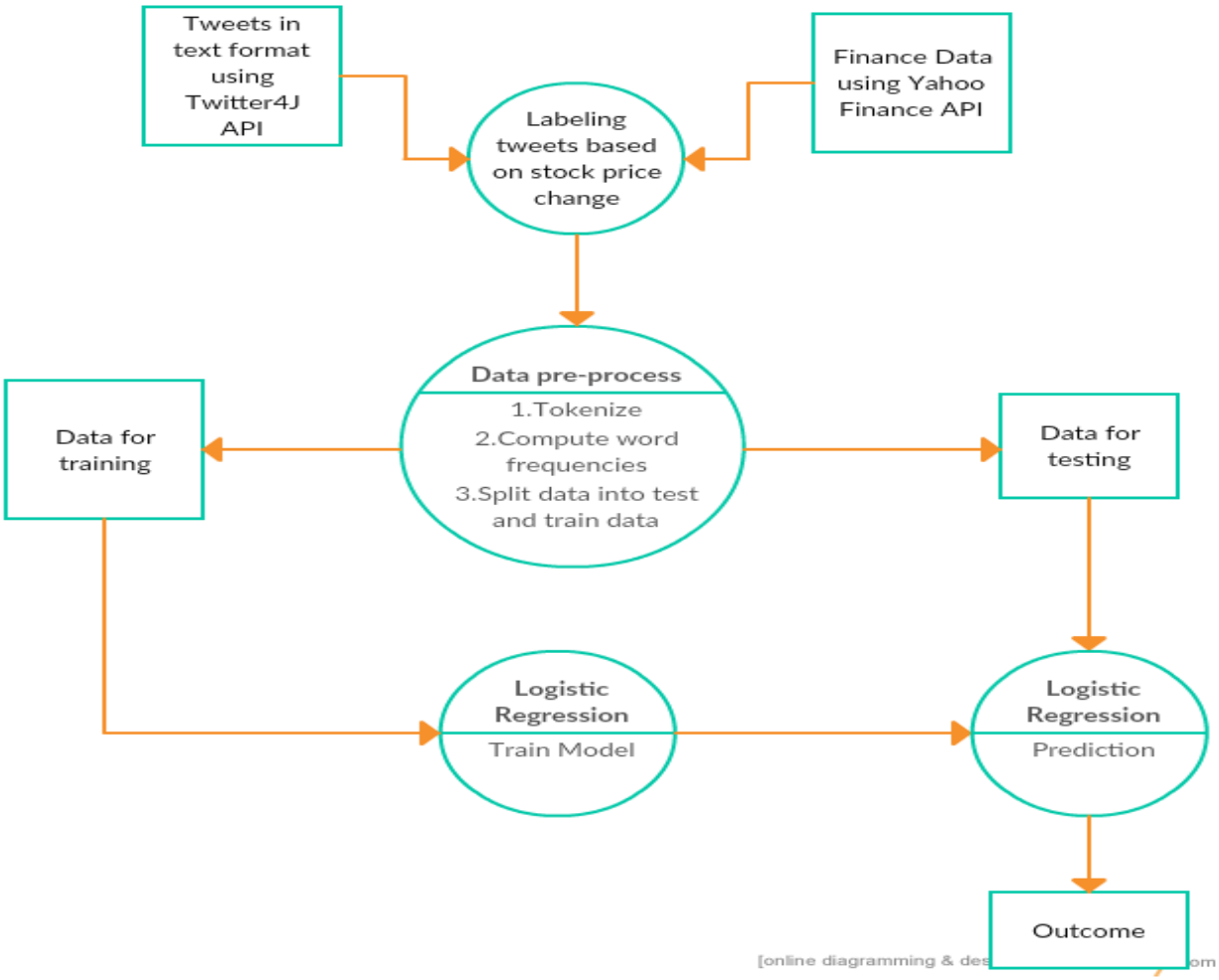
1.4. Big Data technologies

The project uses multiple big data technologies - HDFS, Spark MLlib.

HDFS: We use HDFS during the creation of the input dataset. The tweets are downloaded, assigned labels and uploaded to HDFS. This provides the ability to increase the size of the train and test datasets without affecting the speed of execution.

Spark MLlib: We use Spark MLlib to perform the classification. Since Spark MLlib is a scalable Machine Learning library it has the ability to train models and classify large datasets.

1.5. Data Flow Diagram



2. Analysis of Results

The following table contains results obtained on running the classification multiple times varying the number of instances, test to train split ratio, number of features retained after dimensionality reduction.

Sr. No.	Number of instances	Train: Test split	Predicted label counts	Actual labels counts	Area under ROC	Area under PR
Number of features = 500						
1	8144	60:40	(0): 2260 (1): 925	(0): 2036 (1): 1149	0.6608610288950331	0.6470447030312751
2	8144	65:35	(0): 1991 (1): 790	(0): 1768 (1): 1013	0.6624489666016	0.6526403036116627
3	8144	70:30	(0): 1732 (1):653	(0): 1503 (1): 882	0.6614831561367062	0.6576661799203003
4	1055	60:40	(0): 323 (1): 98	(0): 323 (1): 98	0.83374928918936	0.7745891705850987
5	1055	65:35	(0): 279 (1): 81	(0): 273 (1): 87	0.821523304281925	0.773760110685398
6	1055	70:30	(0): 251 (1): 75	(0): 246 (1): 80	0.8196646341463415	0.7715260736196319
Num features = 1000						
7	1055	60:40	(0): 318 (1): 103	(0): 323 (1): 98	0.819359322676439	0.7477403617397885
8	8144	60:40	(0): 2197 (1): 988	(0): 2036 (1): 1149	0.6814687667246311	0.6689632780628875
Num features = 100						
9	1055	60:40	(0): 337 (1): 84	(0): 328 (1): 98	0.788920831490491	0.7379962673905667
10	8144	60:40	(0): 2676 (1): 509	(0): 2036 (1): 1149	0.5662881022363343	0.5345754141385775

From the above results we notice that contrary to our initial expectation that the prediction accuracy would increase on increasing the number of instances in the train set, we notice a decrease in the accuracy on increasing the size of the train set. The reason for this is due to the bias caused by inclusion of streaming data. Although the number of tweets obtained using the streaming API is higher, the labels assigned to all tweets from a particular day are the same (close - open price from that day. This leads to a lot of noise in data and reduces the accuracy of prediction.

We also notice the effect of dimensionality reduction from the above results. We found that setting the number of features to 500 resulted in the best accuracy of predictions. On decreasing the number of features to 100, we noticed a decrease in accuracy in both cases due to inability of the classifier to classify using 100 features. On increasing the number of features to 1000, we noticed a slight increase in accuracy for the large dataset. However, it resulted in overfitting and a slight decrease in accuracy for the smaller dataset.

In conclusion, we obtained the best accuracy (Area under ROC = 0.833) using the Logistic Regression classifier with 500 features on a data set containing 1055 tweets split in a ratio of 0.6:0.4 as train and test data.

3. Conclusion

3.1. Use of Big Data

We have investigated the relationship between words used in tweets about a company and the variation of stock prices of the same company. The results obtained indicated that it is indeed possible to predict price trends with reasonable accuracy using tweets.

In this project, we use Spark and HDFS to handle the data. Since both technologies work with scalable distributed data, the system can be expanded to handle large amounts of train data and produce predictions in real time.

3.2. Learning outcome

During the course of this project, we understood the use of machine learning using spark and other big data technologies like Hadoop and HDFS.

3.3. Future scope

The efficiency of results obtained can be improved by using shorter time intervals for tracking stock prices and assigning labels. Also, by using natural language processing techniques to process data before performing the classification, we could improve accuracy of the prediction.

4. Role of each Team Member

Research and Design: Whole team.

Twitter Data Download: Bharat Bhavsar & Sai Chakravarthy

Finance Data Download: Anusha Kowdeed

Labeling tweets: Anusha Kowdeed

Preprocessing: Shakti Shivaputra

Classification and prediction: Shakti Shivaputra

Documentation: Manaswi Reddy

5. References

1. Sentiment analysis of twitter data for Prediction of Stock Market Movement:
   http://cs229.stanford.edu/proj2011/ChenLazer-SentimentAnalysisOfTwitterFeedsForThePredictionOfStockMarketMovement.pdf

2. " Stock Price Forecasting Using Information from Yahoo Finance and Google Trend"
   https://www.econ.berkeley.edu/sites/default/files/Selene%20Yue%20Xu.pdf

3. Analysis of Twitter Messages for Sentiment and Insight for use in Stock Market Decision Making
   https://tradethesentiment.com/dissertation

4. Logistic Regression using Spark MLlib:
   http://spark.apache.org/docs/latest/ml-classification-regression.html

5. Feature extraction and selection using spark:
   https://spark.apache.org/docs/1.2.1/mllib-feature-extraction.html

6. Twitter API documentation:
   https://dev.twitter.com/overview/documentation

7. Twitter4j API:
   http://twitter4j.org/en/index.html

8. Yahoo Finance API:
   https://developer.yahoo.com/yql/

   http://www.jarloo.com/yahoo_finance/