Assignment C : Using Twitter to Forecast Stock Prices

BARUCH COLLEGE, MASTERS IN FINANCIAL ENGINEERING

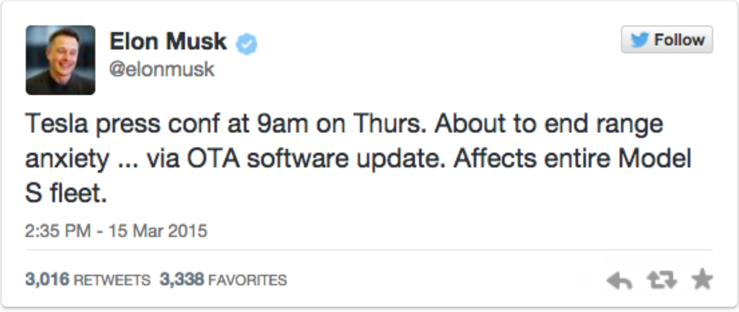
MTH 9898: Big Data in Finance
March 11, 2016

Issued: Friday, March 11, 2016.

Sunday, March 20, 2016, at 11:59PM EST — code & training set error due
Sunday night, March 20th — test set issued
Monday, March 20, 2016, at 11:59PM EST — test set predictions and error due.

1 Introduction

One year ago, Elon Musk sparked off a heated discussion on the internet, and trading on Wall Street, with this tweet:



Elon Musk
@elonmusk

Tesla press conf at 9am on Thurs. About to end range anxiety ... via OTA software update. Affects entire Model S fleet.

2:35 PM - 15 Mar 2015

3,016 RETWEETS 3,338 FAVORITES

At the time of the tweet, Tesla's stock was trading at $191.10 a share, but by the close of trading that day, it had reached $195.70 at the closing bell, a little more than a 2% increase.

This event was not an isolated incident. Other research has suggested the utility of

Twitter data in predicting overall market sentiment, the effect of product recalls,[1] box office returns,[2] and drug approvals. One clear example of why this may work is that some news is spread on Twitter before it reaches other media, and thus other investors, as was the case with an outbreak of norovirus on a Royal Caribbean cruise ship, that caused share prices to drop 2.9%[3]. In other cases, share prices changed in response to tweets by people considered market experts such as Carl Icahn.[4]

2 Problem Description

Interest in using social media to forecast sales and stock variation goes back to at least the late 2000s. While some studies have claimed significant predictive ability[5], however, critics have suggested that such models suffer from difficulties in predicting over time as conditions change.

In this assignment, you will build a simple model to use Twitter data to predict variations in stock prices. To simplify, your job will be to predict one-day ahead closing values of the stock of one or more companies in the S&P 500, using previous closing prices and tweets up until closing time of that day. Please do not include other economic data, since this is not an exercise in gathering and incorporating other additional data. As a baseline, your predictions should do better than simply predicting that the price does not move from the previous day's close.

The full "firehose" Twitter data feed is on the order of 500 million tweets per day, so sorting through it and finding tradeable insights requires more than your grandmother's laptop. This makes it a good opportunity to employ Apache Spark, which has the capacity to deal with even petabytes of data.[6] Twitter restricts this full data to paying customers, so the data you will be working with is considerably smaller. However, with the scalability of Amazon AWS and Spark, your code should be written in such a way that it could be used with full data stream by simply spinning up more resources from Amazon and turning on the firehose.

[1] In March 2015, Kraft recalled 242,000 cases of Macaroni & Cheese due to the possibility that boxes contained shards of metal, causing about a 2% drop in share price. Time.com, March 18, 2015, How Kraft's Mac and Cheese Recall Will Affect Its Stock Price

[2] Asur & Huberman, Predicting the Future with Social Media. http://arxiv.org/abs/1003.5699

[3] Alcorn, 2013. Twitter can predict the stock market... http://www.fastcoexist.com/1681873/twitter-can-predict-the-stock-market-if-youre-reading-the-right-tweets

[4] Deaton. 2016. 5 Giant Stock Moves Predicted by Twitter...
http://thefinancialledger.com/2016/01/02/5-giant-stock-moves-predicted-by-twitter/

[5] For example, Bollen et al, 2010. Twitter mood predicts the stock market.
http://arxiv.org/abs/1010.3003

[6] Xin. 2014. Spark the fastest open source engine for sorting a petabyte.
https://databricks.com/blog/2014/10/10/spark-petabyte-sort.html

3 Problem Specification

1. First, you will need to get the Twitter data into a format you can work with in Spark. The Twitter data is available to you in an Amazon s3 bucket at `s3://bdif-tweets/`, organized into folders by year. This data is monthly `.tar`balls that are composed of bzipped `json` files, which are 20-50GB each when zipped. To help with the import, we will also be providing cleaned data that may resolve problems with loading the data into Spark (bad characters), in the `cleaned` folder. For more information about loading data into Spark and working with it, see the materials provided in the workshop which are posted on forum.
2. Secondly, you will need a basic model to use the Twitter data in your predictions. You are not expected to develop a sophisticated machine learning algorithm, since that is the focus of the second part of this course. Instead, focus on a simple idea that is well executed. Some studies claim predictive ability just from using the volume of tweets that mention a specific company; this may be a good starting point, and once you have a framework for making that work you can consider more sophisticated models.
3. The training data we provide covers 2012 and 2013. The test set will be a six-month period that will begin no earlier than Jan 1, 2014. Please note that it will not help to try to build your model around *specific* knowledge about the period covered by the test data (that is, knowledge you have as a result of hindsight) since we will penalize you in your coding and you would likely do poorly in the in-sample predictions. The test data will be released when the project is due, after which you will run your code on the test data and submit your predictions within 24 hours. You should report the mean absolute error rate between your prediction and the actual closing value, as well as the mean absolute error between the closing price and the previous day's closing price (as a baseline). You can get daily closing stock price data in whatever way you find most convenient. For one example using the Yahoo API in python, see `https://github.com/aristotle-tek/cuny-bdif/yahoo_stock_data/`.

4 Some Hints to Help You Along

Please note that this is a challenging assignment, since stocks fluctuate for many reasons not easily captured by Twitter data and given that much of the Twitter data is noise that should be disregarded. You should focus your time on demonstrating the basic technical skills and approaching the problem in a sensible way, writing clear and concise code, and ensuring that your predictions are at least reasonable (i.e. not worse than guessing).

If you have the basics of working with Spark down, you can go beyond calculating simple volume of tweets by identifying when mentions of the company are associated with meaningful words. You could do this to focus on specific events, such as product releases mentioned above, or more general sentiment analysis. For the latter, you might find useful the Loughran and McDonald Sentiment word list, which provide positive and negative measures for a wide variety of words. These were developed by relating 10-K reports to stock performance (see the papers on their website for more information): `http://www3.nd.edu/~mcdonald/Word_Lists.html`

5 How This Assignment Will be Graded

Partial programs that do not meet all the requirements described above will receive credit. You will get marks for showing you were heading in the right direction.

Points will be allocated as follows:

- 30% for meeting the requirements of the program specification using Spark to analyze the text of Twitter data, e.g. by generating time series of Tweet information for specific firms.
- 20% for designing a model to use the tweet time-series data in a model for predicting closing price.
- 20% for programming style. This encompasses not just the clarity of your code, but also such things as efficient data structures, efficient algorithms and use of system APIs and resources. While use of libraries is generally encouraged, there is no substitute for understanding the underlying algorithms, so for this assignment implementing your own algorithms rather than relying completely on library functions will get you more points. However, get your programs working correctly first, and only substitute library functions with your own code if you find yourself with time left over.
- 15% for the in-sample mean absolute error between your predictions and the closing price (that is, on the training set). Full credit is only possible if your mean absolute error is less than the mean absolute error of using the previous day's close as the prediction.
- 15% for the out-of-sample mean absolute error between your predictions and the closing price (that is, on the test set). Full credit is only possible if your mean absolute error is less than the mean absolute error of using the previous day's close as the prediction.

This is a challenging exercise, so if you find yourself struggling, reach out to us sooner rather than later. And remember that this is not a pass/fail exercise and that you can still get a good grade with a less than perfect program.