BDIF_C Assignment C

Date: Mar.20 2016

Student Name: Yue Wang

This document designs to outline the summary and methodology being used in Big Data Assignment C that intended to test the hands-on experience of utilizing Apache Spark to predict stock price through learning the given tweets data from Twitter.

Tools used: AWS and PySpark through command line and logging

Assumption: This exercise I assumed to analyze a single stock – to make it feasible and robust, I I tend to analyze the tweets impact of popular stocks that I can get enough sample data from a specific tweet data set, top notched technology firms came to mind in the first place such as apple, google, facebook etc. Taking apple as an example, then any other stock should roughly follow the similar procedure.

Step1: Data Scrubbing

Fit the given data to a hash-map data structure for robustness and quick access by leveraging pySpark SQLcontext utility

1. First layer of filtering - language, we have to assume the twitter tweets we will be analyzing are solely based on English instead of any other languages as we don't have expertise of it, despite the fact that it would be great to do so as Apple is a multi-national firms that operates worldwide where everyone enjoys using the apple products.
2. Second layer of filtering – keyword associated with the stock being analyzed, like “Apple”, or apple’s executive such as his CEO “Tim Cook”, or apple’s product line such as ‘macbook’, ‘itouch’, ‘iphone’ etc. every tweets that come across with above mentioned keywords assume to be a relevant tweet. – anything that not with such keywords association will be discarded.
3. The third layer of filtering is associated with counting the positive and negative effect of some keywords like the ones given from the given input file. Given the time constraint, I will just test for 10 most representative positive words and 10 most negative representative words.
4. List of the top 10 positive words and negative words as following (based on the file provided by Andrew – in homework specification): - I filter out the ones I think would do the best job
    - positive: achieve, reward, profit, positive, innovat (intentionally leave this, as innovate or innovation both works), improve, favor, enhance, enjoy, great
    - negative: accuse, accident, risk, refuse, recall, problem, plea, penal, negative, mis(pre-fix)

Step2: Data Programming and analysis:

Initial cleaned data was created and given by Andrew (thanks to Andrew), I used one single un-tar piece of data (about 330MB Json file) and use function of SQLcontext from PySpark to read data into a Data Frame object and I created a four different-scale input files aside from the sample ones for robust testing purpose (20, 200, 2000, 20000 and 200000 tweets respectively). The SQL utility provided by Pyspark is heavily used by constructing the predefined keywords targeting the information to be analyzed. The methodology is very simple by analyze any given day data, if there are considerably more positive keywords (70% tweets or above) associated tweets it will be classified as positive, if there are negative keywords (30% tweets or less) associated tweets it will be classified as negative, if there are between then I will consider it as a neutral signal with 50% going up or 50% going down.

Step 3: Data insight and limitation

- The limitation of existing methodology is it could make the analysis more effective by branching more keywords searching (100s instead of 10s) and combine the learning text with the actual historical stock performance to testify the underlying model assumption which could turn out to be more effective and efficient.
- Data insight is that interestingly enough the conventional keywords such as not is not necessarily bad, cause using single keyword search couldn't do a good job to identify the whole context which is more important, it might be the tweets was mentioning apple's competitor sumsung indicating they are not good as opposed with apple.
- Another potential improvement of existing procedure is to collaborate with sentence pattern as well as potentially analyzing the competitor information, if the competitor's tweets turned out to be bad, then its more naturally to assume this could be a good impact for apple.

Summary and Takeaway

This is an interesting project and I wish I could be given more time to explore other powerful functionality spark embed. The model is bit toyed but fun to test and see.