Spark Near Real-Time Sentiment Analysis

Summary: The goal of the project is to study the effect of social media tweets focused towards stock market on underlying stock prices by performing sentiment analysis. The data source for tweets is from StockTwits.com where users express their opinion in the form of tweets. There are three distinct phases to the project. The first phase provides a generic approach to download historical tweets for the chosen stocks for a chosen period. We then preprocess tweets using NLTK lemmatizer and stopwords. The second phase introduces couple of models to perform sentiment analysis using Bag-of-Words approach and Naive-Bayes classifier. The third phase performs regression analysis using the daily sentiment scores produced by previous phase to see if social sentiment can be a predictor of underlying stock price changes. We demonstrate the ability to sense the overall sentiment in real-time using spark streaming of tweets.

Data set:
1. Two months worth of tweets for three chosen stocks (AAPL, FB, TSLA) from www.StockTwits.com
2. Two months of historical stock prices from http://finance.yahoo.com for the same set of stocks.

Technology
Python and various API - StockTwits RESTful, NLTK, Spark Streaming, Tensorflow API to implement various phases of the project.

Benefits:
- Ease of use - python and its seamless integration with web downloads, text analysis, spark streaming and machine learning packages.
- Ease of web downloads using RESTful API.
- Ease of natural language processing using NLTK lemmatizer and stop-words packages.
- Parallel processing using spark streaming API by simultaneously running sentiment analysis on multiple stocks.

Drawbacks:
- ??

Challenges:
- StockTwits RESTful API has a limit of 200 requests per hour. Each request returns 30 tweets for a chosen stock in JSON format.
- Signal-to-noise ratio in the user tweets. Tweets are often highly unstructured in an informal language. Some noise can be filtered by using appropriate dictionary of words and better processing of text (stop-words, n-grams, lemmatization). Pre-processing has implication on accuracy of the built models.

Demo Workflow: We implemented various phases of sentiment analysis as a work-flow.
- Download AAPL tweets using StockTwits RESTful api and json processing. Output is a csv file with each record representing one tweet having following fields -
  TweetID, Stock, Date, CreateTime, Text, Sentiment
- Preprocess tweets csv file. Remove stop-words first. Then perform lemmatization to reduce the dimensionality for different usages of the same word (organize, organizes, organizing).
- Perform Bag-of-Words analysis to calculate sentiment score using two dictionaries. Loughran McDonald dictionary and the Harvard dictionary. Present confusion matrix for model accuracy.

- Perform Naive-Bayes analysis to predict sentiment score by training the model first and using test data to verify. Present confusion matrix for model accuracy.
- Perform Regression analysis using Tensorflow api to see if changes in sentiment score can predict changes in underlying stock prices.

Documentation:
Github repository for code, data and logs - https://github.com/mpatnam/CSCIE63-Project
YouTube URL of the full presentation video: http://youtu.be/….
YouTube URL of the 2min preview presentation video: http://youtu.be/….