Live Cryptocurrency Sentiment Analysis on Twitter

Dmytro Siniukov siniukov@kth.se 951228-2436
Miquel Larsson miquell@kth.se 920614-5998

Group Name: LarssonSiniukov

1 Summary

Real time sentiment analysis is a hot topic in the last years - as having live information of how users and consumers feel regarding almost any topic is extremely valuable for companies or any organization. Twitter is a perfect source of real time information to analyze, as we can filter by specific topics and keywords. Also, the limited nature of the tweets (140 characters maximum) makes application of natural language processing techniques accessible.

Our project had the main focus of applying natural language techniques (provided by a well known library) in order to extract general sentiments from real time tweets regarding cryptocurrencies, and then compare this aggregated sentiment with a cryptocurrency price dynamics. To reduce complexity, the chosen cryptocurrency was Bitcoin ($BTC), as it is the most well known one, with the highest volume of tweets.

2 Data

The tweets are taken live from the Twitter API. In order to achieve that, we have created a Twitter developer account and a test application to login from. After that a Spark stream is created (using `TwitterUtils.createStream(...)`) to read live tweets.

Real time market valuation of Bitcoin is obtained from the Cryptocompare public REST API. This turned out to be non-trivial, and will be commented further on in the present report.

3 Methodology

First, live Twitter data is pulled from the API through Spark Streaming (as described in the *Data* section). Then, the sentiment analysis is performed (using `edu.stanford.nlp.pipeline.StanfordCoreNLP` scala library) in order to determine the sentiment of each tweet. In parallel, real time market valuation of Bitcoin is read.

Then, results of the sentiment analysis together with the corresponding bitcoin price are written into an Elasticsearch instance. Finally, we visualize different aggregated metrics of the streaming data in a dashboard with several plots using Kibana.

4 Final Results

When executed successfully, an output as the one seen in Figure 1 is viewed. If everything is working correctly, the current Bitcoin price should be displayed, and the obtained tweets with their sentiments should be shown below, with the defined frequency of the stream (30 seconds in our case).

```
Bitcoin Price (USD): 6421.92
+--------------------+--------+---+
|                  _1|      _2| _3|
+--------------------+--------+---+
|RT @cryptorecruit...|POSITIVE|  3|
|The Surety really...|NEGATIVE|  1|
|RT @cliffor586702...|NEGATIVE|  1|
|RT @Remidax: Remi...|NEGATIVE|  1|
|Huge coin selecti...| NEUTRAL|  2|
|The average price...|NEGATIVE|  1|
|It then displays ...|NEGATIVE|  1|
|RT @Platioecosyst...|POSITIVE|  3|
|RT @PanteraCapita...|NEGATIVE|  1|
|RT @buying_com: N...|POSITIVE|  3|
|Order your secure...| NEUTRAL|  2|
|Earning #cryptocu...|NEGATIVE|  1|
|Bitcoin Cash BCH ...|NEGATIVE|  1|
|RT @blockchain: F...|POSITIVE|  3|
|Ethereum [ETH] & ...|NEGATIVE|  1|
|RT @MoonOverlord:...|NEGATIVE|  1|
|RT @CryptonityEx:...|NEGATIVE|  1|
|Bitcoin Price Est...|NEGATIVE|  1|
|Crypto Markets Se...|NEGATIVE|  1|
|Buy/Sell Bitcoin/...|NEGATIVE|  1|
+--------------------+--------+---+
```

Figure 1: Example of console output when executing the code. At the top, the current Bitcoin price is displayed in USD, and below, a sample of the obtained and analyzed tweets, with their corresponding sentiment.

Then, Kibana is able to read from the data pushed to Elasticsearch, and display different aggregated metrics of the streaming data in a dashboard with several plots, as seen in Figure 2.

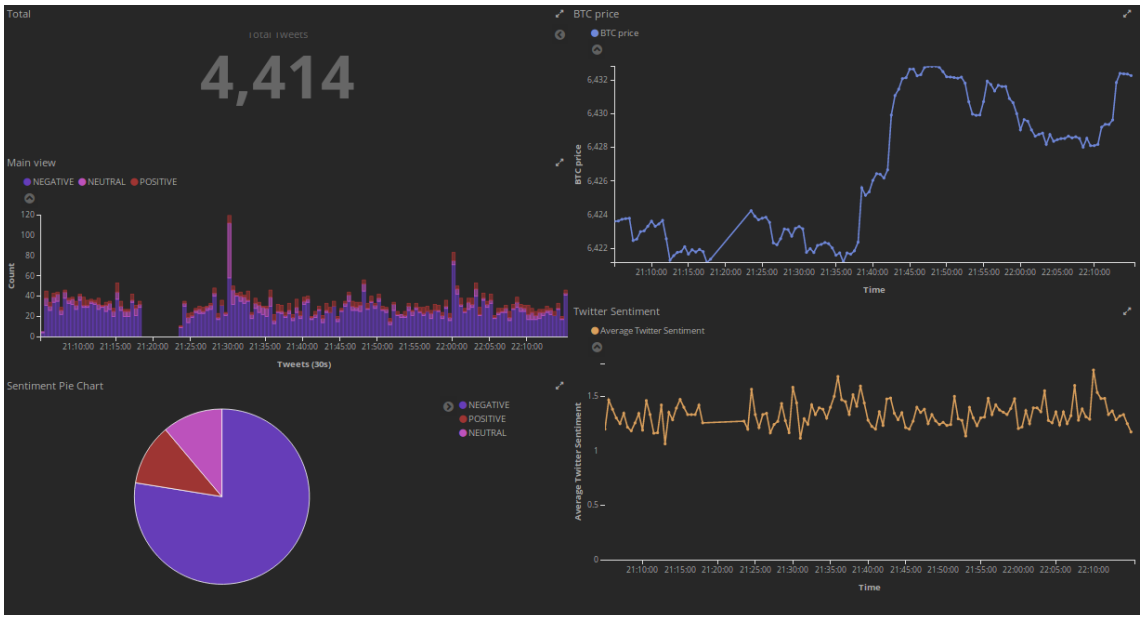

Figure 2: Final dashboard with real time updated plots.

Concretely, the following plots are displayed:

- A counter with the total amount of obtained and analyzed tweets.

- A stacked bar chart with respect to time and sentiments.
- A pie chart with a break down with the total count of each detected sentiment.
- A line chart displaying the evolution of the Bitcoin market valuation in USD with respect to time.
- A line chart displaying the average measure of sentiment of the tweets for each processed batch (30 seconds in this case). A value of 2 Corresponds to "neutral", above that is considered "positive", and below, "negative".

5 How to run the code

The steps to run the full code and reproduce the displayed results are:

- Check that all the required dependencies and libraries are installed and running: Java, Scala, sbt, elasticsearch, kibana...
- In order to pull the live tweets from the twitter API, API credentials are required. These have to be input through the standard input after running (`sbt run`).
- The Elasticsearch index has to be set prior to executing the code, as some datatypes are not set correctly if left to automatic index setting (Sentiment for example, has to be set to type keyword in order for it to be aggregatable). The exact index can be found in `/visualization/elasticsearch.sh`. If the elasticsearch indices have to be deleted, this can be done with the command `curl -X DELETE 'http://localhost:9200/_all'`.
- Kibana has to be set to plot correctly the incomming data: the json to import is defined in `kibana_dashboard_and_visualizations.json`. Kibana can be opened going to the direction `http://localhost:5601` in any web browser.
- Now, the code can be compiled to verify that there is no problem with dependencies: `sbt compile`.
- Finally, the code can be run with `sbt run`. If everything works correctly, the results of the last section should be obtained.

6 What we have learned and takeaways

- How to use Spark streaming and obtain real time tweets about a certain topic.
- How to create and use a twitter developer account and twitter apps.
- How to read and parse a json from an API in scala. This turned out to be non-trivial, as the required libraries to perform this task are not very well documented and the required imports were prone to be incompatible with the needed prior libraries.
- It seems that the NLP library tends to classify tweets as "NEGATIVE". We did not go in depth in this aspect, as the NLP part was not the objective of the project. This should be checked in further revisions of the project.
- Inserting streaming data in the elasticsearch DB. This also resulted to have quite a lot of difficulties, as any mistake with the indices would result in a long and obscure list of errors.
- Visualization of data in Kibana. Kibana was a surprise, as we had not considered using it at the beginning, but resulted to be simple and work well right from the beginning.

7 Conclusions

To conclude, we have implemented a real-time twitter sentiment analyzer, that pushes the processed data to elasticsearch, which then is used by Kibana for visualization processes. While the current implementation is simple, it sets a good ground work for future extensions that could be interesting. For example, in the current implementation, real time twitter sentiment is measured, but it should be expected that price and sentiment have a delay between each other. Furthermore, an analysis on correlation and causality could be done to see which parameter precedes which. We believe that while Bitcoin is a good starting point, analyzing smaller cryptocurrencies (with higher market valuation volatility) would give more relevant results. Nevertheless, the current project has allowed us to interact and work with several tools, which has allowed us to learn and improve our skills.