Twitter Data analysis using Spark Streaming, Kafka & NodeJS

**Gautam Saini, Shilpi Sirohi**

**University of South Florida**

This is Project 4 for the course

CIS6930.002 ADVANCED DATABASES

Abstract

In today’s world, Twitter is one of the biggest sources of streaming data. So much analysis can be done on that the Twitter data like text analytics, sentiment analytics. We decided to do extract the live data and then performed two analyses: sentiment analysis and hashtag counts from streaming data.

The project aims to design an app to fetch Twitter live data using Kafka broker service and Spark streams and then perform in place sentiment analysis. The project starts a node server, in the end, to display the final analyzed data onto a locally hosted web app.

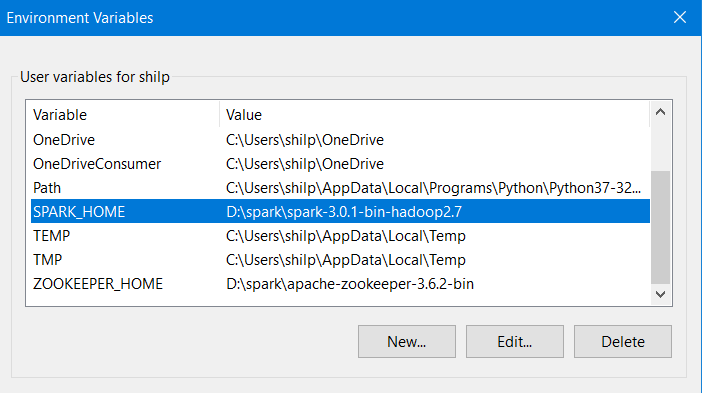
# **Project Setup**

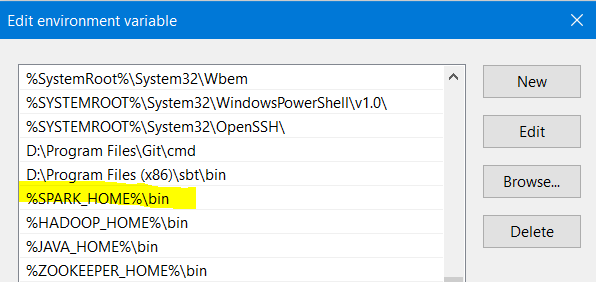
## Setting up the Machine

We need to set up the machine so that we can perform the task. The instructions are to setup Windows 10 machine. Below are the requirements-

* Java 8
* Python 3.7
* Node.js
* Spark
* Zookeeper
* Kafka
* Visual Studio Code Editor to install c++ development distribution

To setup Spark, download the latest Spark from the official website ( <https://spark.apache.org/downloads.html> ). While downloading make sure to select package type as “Pre-build”. Then set up the environment variables



Also, add it to the Path

To run Kafka on windows 10, the zookeeper needs to be set up first. Please follow the instructions given at the link for the setup of zookeeper and Kafka.

<https://dzone.com/articles/running-apache-kafka-on-windows-os>

## Creating Topics for Kafka

Topics are virtual groups of one or many partitions across Kafka brokers in a Kafka cluster. A single Kafka broker stores messages in a partition in an ordered fashion i.e. appends them one message after another and creates a log file.

Producers write messages to the tail of these logs that consumers read at their own pace. Kafka scales topic consumption by distributing partitions among a consumer group, which is a set of consumers sharing a common group identifier. Here we are two different types of analysis, so we created two topics to store the results. For hashtag count, the topic is “**processedtweets**” For sentiment analysis, the topic is “**processSentiments”.**

The start\_app.py script automatically takes care of creating topics but in case if a new topic is required then execute the following command to create it:

Kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic <your\_topic>

Open the command prompt and go to %KAFKA\_HOME%\bin\windows and then execute the above command.

## Setting up the project in VS code

Clone the code from the GitHub repo <https://github.com/KaleidoscopeIM/kafka-Spark-Node-Realtime-Twitter-Sentiment-Analysis.git>

There are two parts of the code: UI which is a node.js application and Spark Application which is written in python.

### Setup Spark Backend application

Go to the folder and then download all the dependencies. The requirement file is already available with the code. Go to terminal and execute the below command

pip install requirement.txt

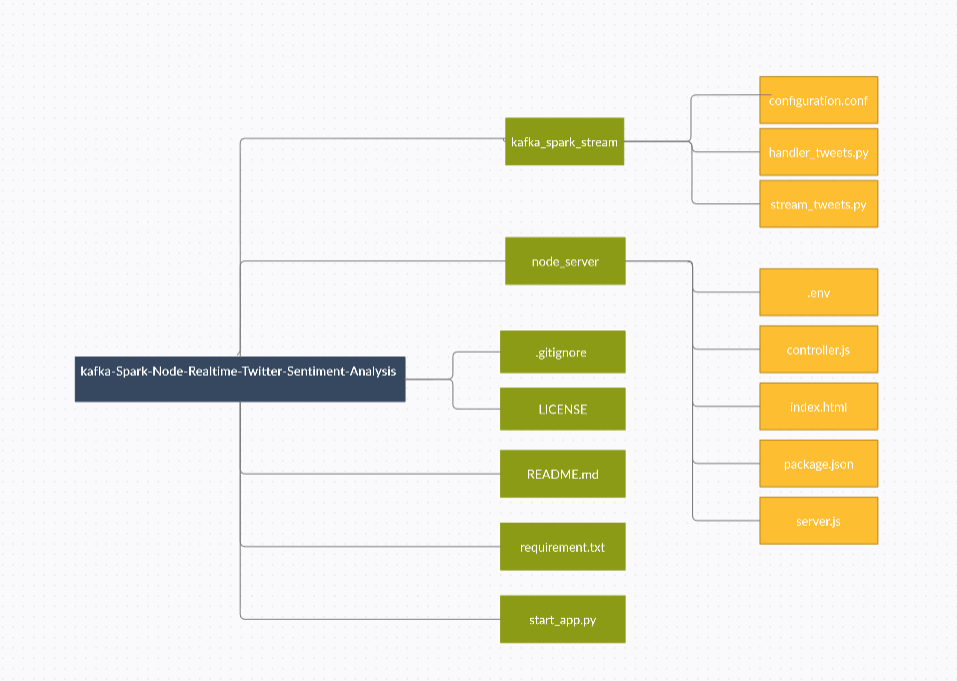
### Setup UI application

Go into the folder **node\_server**/ and then install all node related dependencies. Go to the terminal and execute the command npm install

Now the machine and application setup is completed.

## Application Code

Below is the structure of the code



### Kafka\_spark\_stream

##### **Configuration.conf file:**

This file contains the configuration of the application. To run the code, the user has to add/modify the below values.

|  |
| --- |
| [API\_details] |
|  |

|  |
| --- |
| consumer\_key = ${{ secrets.consumer\_key }} |
|  |

|  |
| --- |
| consumer\_secret = ${{ secrets.consumer\_secret }} |
|  |

|  |
| --- |
| access\_token = ${{ secrets.access\_token }} |
|  |

access\_secret = ${{ secrets.access\_secret }}

|  |
| --- |
| [Kafka\_installation] |
|  |

kafka\_window\_dir = c:/kafka\_2.12-2.6.0

##### [**stream\_tweets.py**](https://github.com/KaleidoscopeIM/kafka-Spark-Node-Realtime-Twitter-Sentiment-Analysis/blob/main/kafka_spark_stream/stream_tweets.py)**:**

This file contains the code to fetch the data from Twitter using the library tweepy. Tweets are fetched based on the filters provided (languages, track, and locations). These values are fetched from the config file. Currently, the language is set to English, the location is set to the USA and we are tracking the tweets based on hashtags. Users can update those as per the requirement. The fetched tweets are sent to Kafka producer which will be later fetched by Spark code for analysis.

##### [**handler\_tweets.py**](https://github.com/KaleidoscopeIM/kafka-Spark-Node-Realtime-Twitter-Sentiment-Analysis/blob/main/kafka_spark_stream/stream_tweets.py)**:**

Here we will fetch the data from Kafka producer and then do the computation. This file contains the logic to do get the Top 10 hashtags and get the sentiment analysis of the tweets. We use TEXTBLOB library to find the sentiment of the tweets. Then, we will send the computed output to the topics. We are sending the data every 30 seconds.

### Node\_server

##### **Server.js:**

This file contains the code to create a client-side server.

##### **Index.html:**

Here we are listening to the socket for data. As soon as the application receives the data, it will update the dashboard charts. Charts are created using the Chart.js library.

### Start\_app.py

This is the file that will start the whole application. Once the setup and configuration are completed; execute this file in the terminal python start\_app.py

This will start zookeeper, then Kafka. After that, it will execute the stream\_tweets.py and then handler\_tweets.py. In the end, it will trigger the node server and open the default browser window to display the dashboard.

## Output

Below are the screenshots of the final output.

