1)CREATE AN ACCOUNT IN TWEETER APP. Get app

api\_key = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

api\_secret = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

access\_token = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

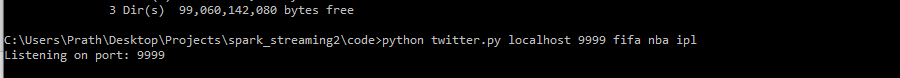
access\_token\_secret = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

2)WRITE PROGRAM TO CONNECT TO TWITTER STREAMING API USING TWEEPY.

3) RUN THE PYTHON PROGRAM.

**Twitter.py localhost 9999 fifa nba ipl**

**Twitter.py localhost 9999 fifa nba corona**



4) WRITE SPARK PROGRAM TO FILTER AND PROCESS DATA (Count hashtag in twitter to find the overall trend.)

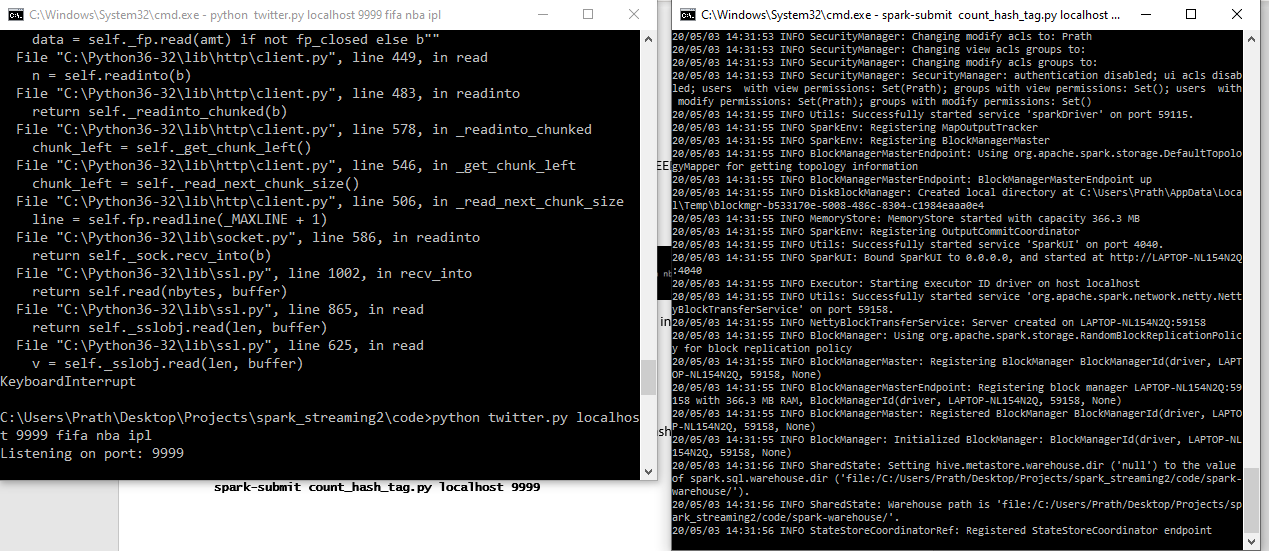
4.1) Need to connect to a host and port.

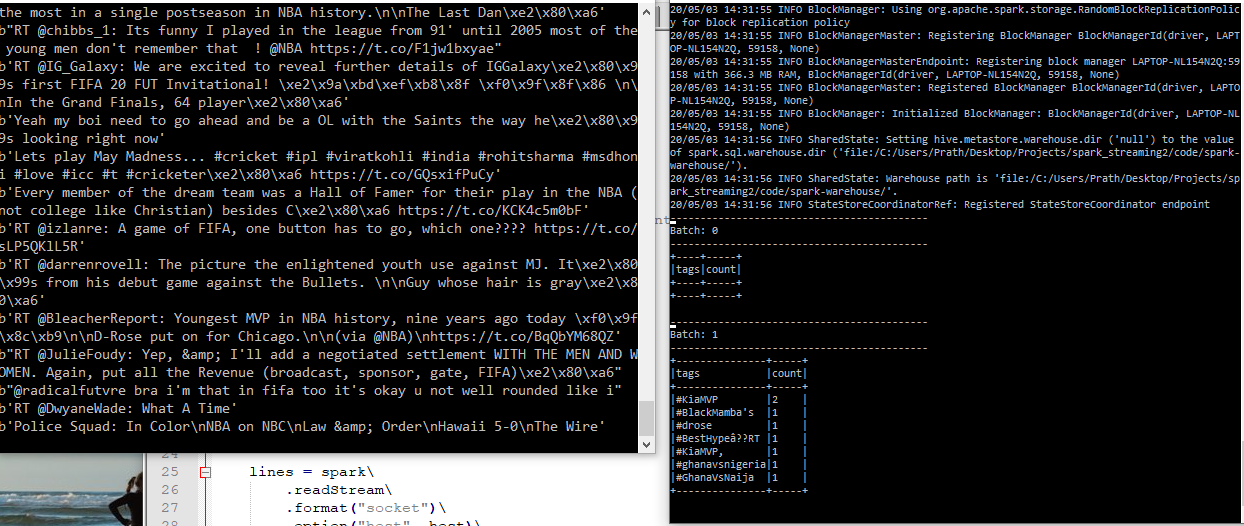
4.2) Use spark session to read streaming data

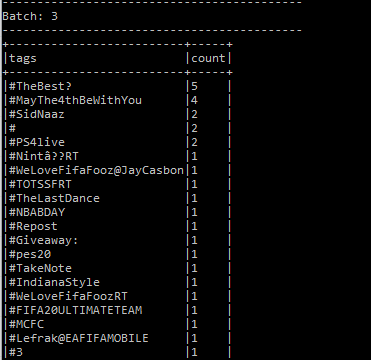
4.3) Extract the message and split the message to get individual hash tag

5) Submit the spark job

**spark-submit count\_hash\_tag.py localhost 9999**



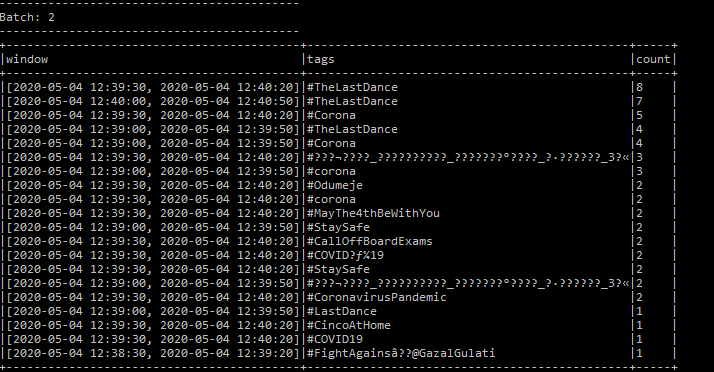


Git 

6) Include window function to see the trending tweets for every 50 seconds, with sliding window of 30 seconds.

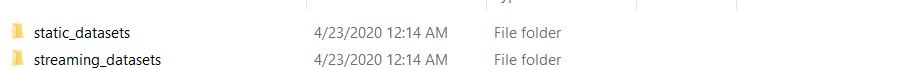
**spark-submit count\_hash\_tag\_window.py localhost 9999**

**Output**

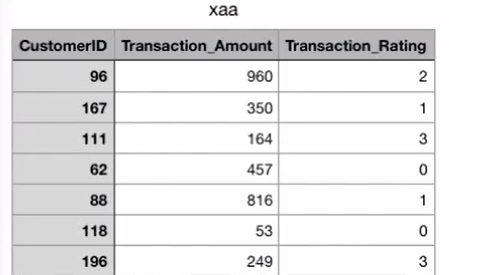


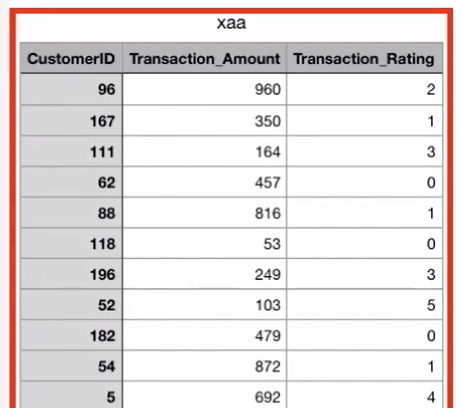
7) Join with batch and streaming data

Within the dataset folder we have



This contains customer specific data transactions done by the customers.

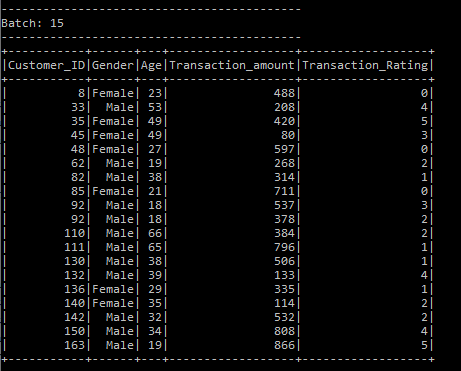




Run:

**spark-submit join\_batch\_streaming.py localhost 9999**

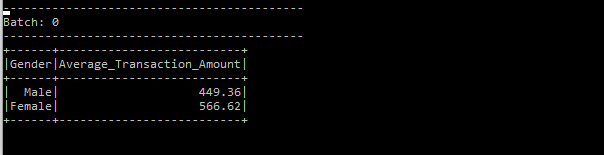
Output



8) GET THE AVERAGE TRANSACTION AMOUNT PER GENDER

**spark-submit join\_batch\_aggregate.py localhost 9999**

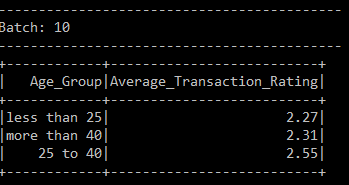
output



9) Find aggregate ratings based on age groups

**spark-submit aggregate\_ratings.py localhost 9999**

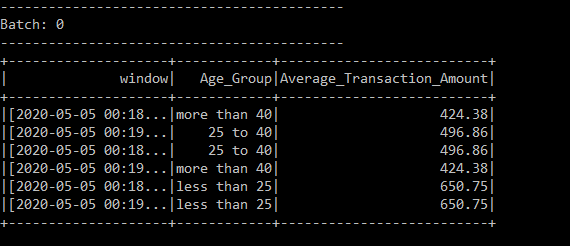
output



10) Average transactions of the customers spend split by age group over certain interval of time

Note:window is of 2 minutes in length and sliding interval of 1 minutes

**spark-submit aggregate\_ratings\_window.py localhost 9999**



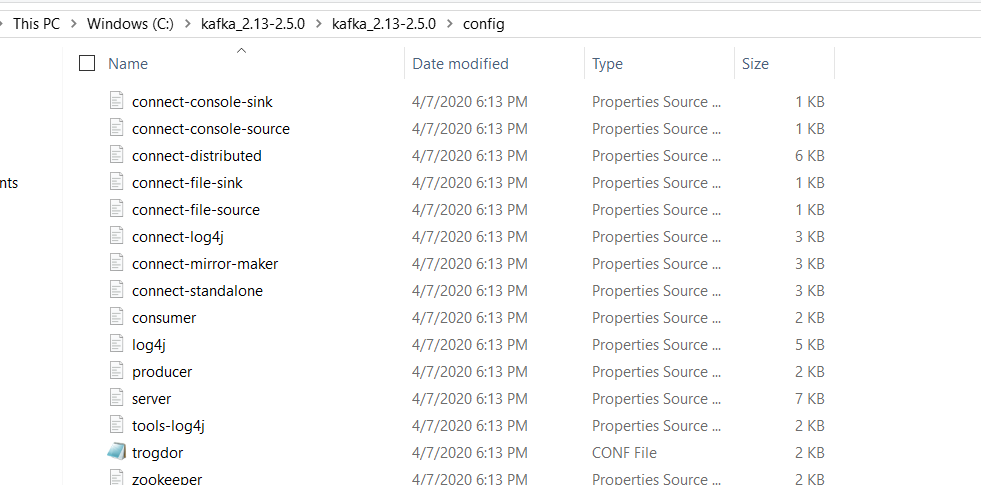
**KAFKA**

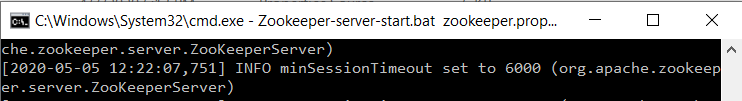
1.Install kafka and set the path in the environment variable

2. Try to run zookeeper. If we get error, then we must delete class path from environment variable

3. start zookeeper

To start zookeeper, we need to go the config folder and run the bat command

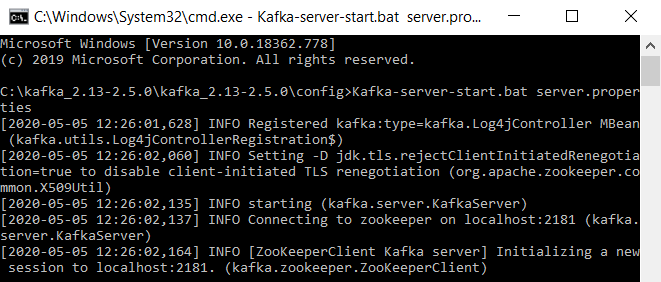




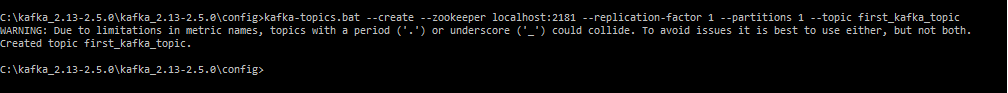
**Zookeeper-server-start.bat zookeeper.properties**

5. Start kafka server

**Kafka-server-start.bat server.properties**



6.Create a topic



List kafka topics

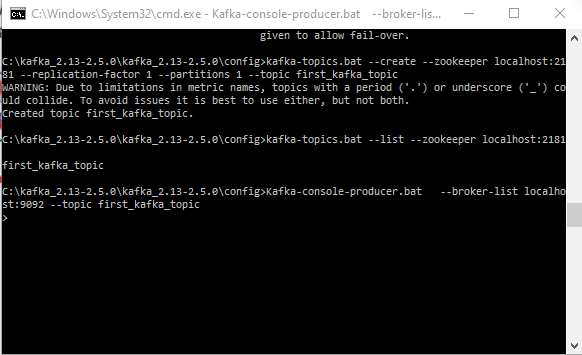
**Kafka-topics –list –zookeeper localhost:2021**



7.Create kafka producer and publish to the topic which we created

9092 is our kafka broker

**Kafka-console-producer.bat --broker-list localhost:9092 --topic first\_kafka\_topic**



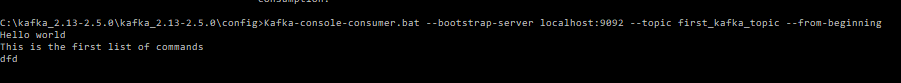
8. Create kafka consumer.

Note 1) We need to specify the bootstrap server.

we need message from the beginning.

**Kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic first\_kafka\_topic --from-beginning**

**Output**



**KAFKA TWEET PRODUCER**

In this example we would be using a kafka producer to stream tweets.

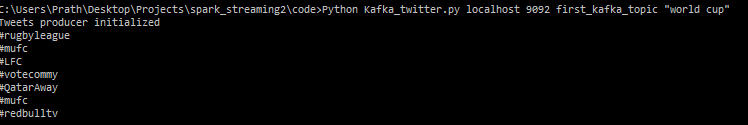
Note:

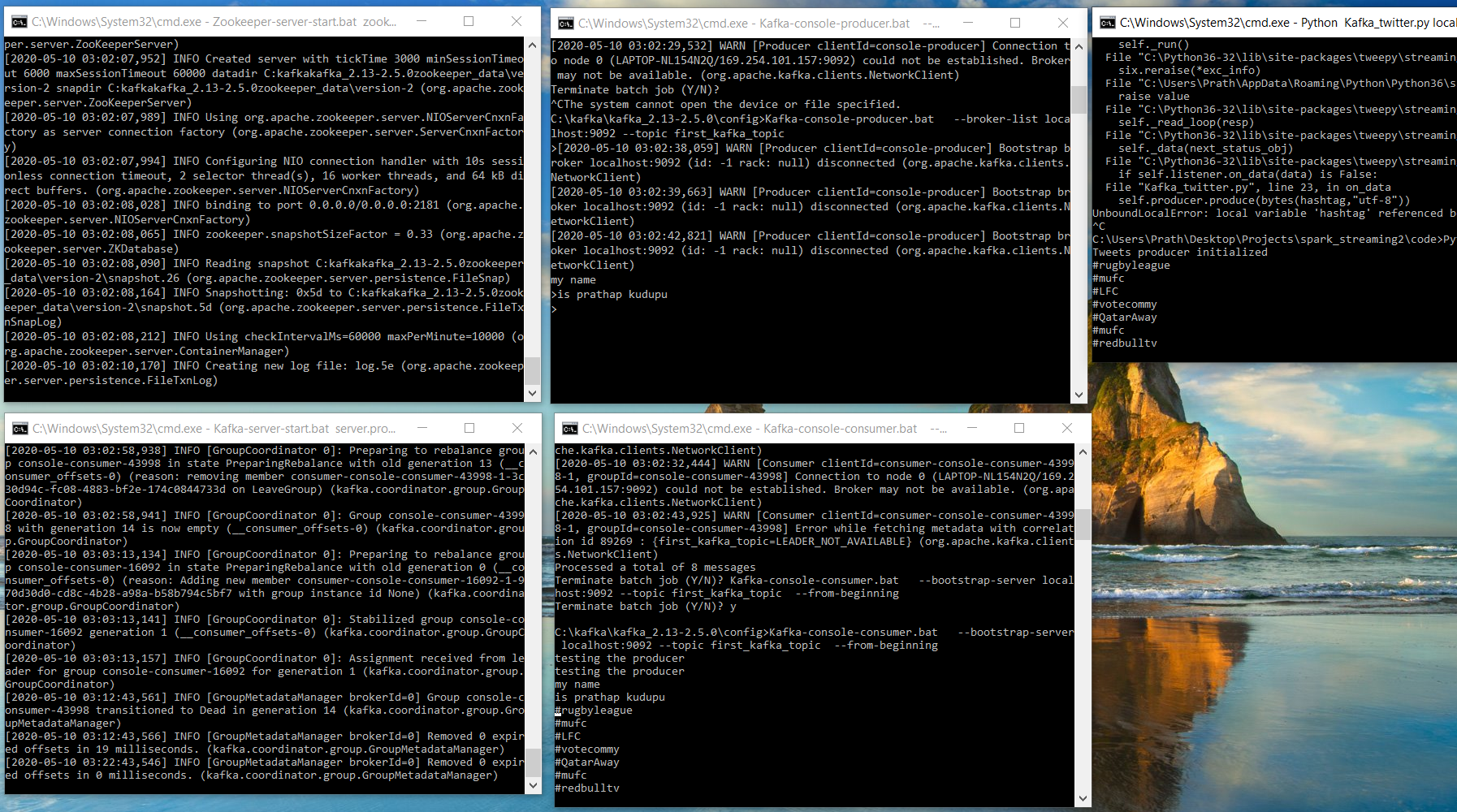
* Here we are streaming the tweets from twitter to kafka producer
* We need to set up the tweets listener
* Tweets are received in the form of JSON
* We load the data to a python dictionary
* Split tweets into individual words and find the tweets which has hashtag

1. We need to run all the services including zookeeper, brokers, producers, and consumers.
2. We then need start the tweet listener

**Python Kafka\_twitter.py localhost 9092 first\_kafka\_topic “world cup”**

Output





We can see that that the consumer which has subscribed to the topic is receiving the data from the tweet listener

**KAFKA SENTIMENT ANALYSIS**

* Use kafka producer with tweet
* Analyze tweet sentiment using afinn (We can do sentiment analysis on text without using any machine learning)
* Entire message would be sent

1. We need to run the zookeeper, kafka brokers, producers
2. Start the tweet listener

**Python kafka\_twitter\_sentiment\_analysis.py localhost 9092 first\_kafka\_topic “world cup”**



1. Specify the dependencies that the spark would use

**spark-submit --packages org.apache.spark:spark-sql-kafka-0-10\_2.12:2.4.5\ sentiment\_analysis\_streaming.py localhost 9092 first\_kafka\_topic**

Output

