Twitter Sentimental Analysis using Apache Spark Streaming and Python

Goal:

In this project, we will learn about processing live data streams using Spark's streaming APIs **(Spark stream introduces concept of DStream(Discretized stream)a Dstream is nothing but streams of RDD where one RDD holds one time of data i.e. 1 batch=1 RDD)** and Python. We will be performing a basic sentiment analysis of realtime tweets. In addition,
you will also get a basic introduction to Apache Kafka, which is a queuing service for data streams.

Background:

In this project, we will be processing streaming data in real time. One of the first requirements is to get access to the streaming data, in this case, realtime tweets. In addition, you will also be using Kafka to buffer the tweets before processing. Kafka provides a distributed queuing service which can be used to store the data when the data creation rate is more than processing rate.

Project Setup (Pipelining)

A.Installing Required Python Libraries

kafka-python==0.9.4
oauthlib==1.0.3
requests==2.8.1
requests-oauthlib==0.5.0
six==1.10.0
tweepy==3.3.0
configparser==3.3.0.post2
matplotlib==1.5.0

B. JDK Setup

1. Start the JRE installation and hit the “Change destination folder” checkbox, then click 'Install.' (Imp!!)
2. Change the installation directory to any path without spaces in the folder name. E.g. C:\Java\jre1.8.0_xx\. (By default it will be C:\Program Files\Java\jre1.8.0_xx), then click 'Next.'

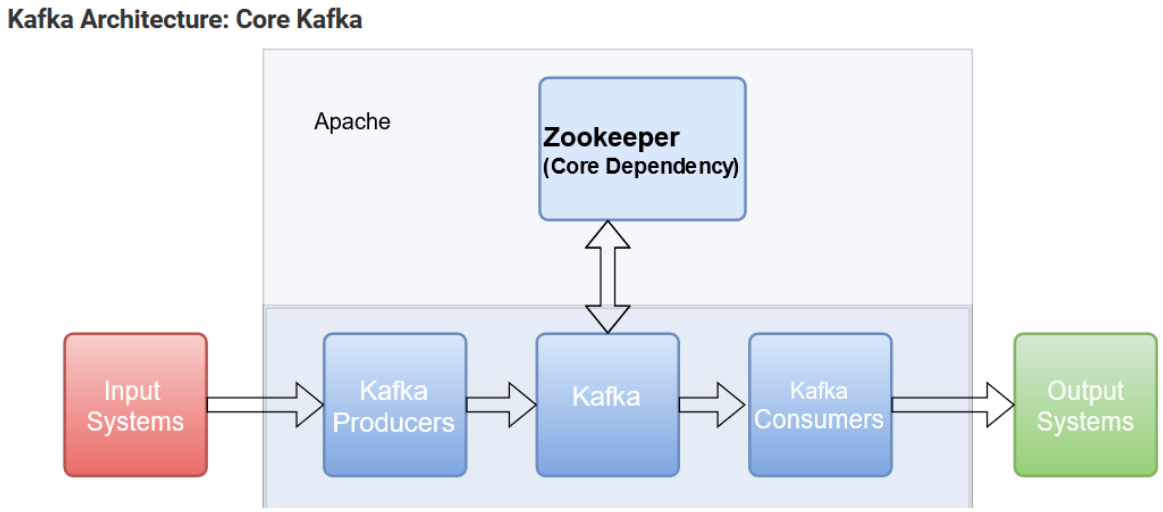
3. Now open the system environment variables dialogue by opening Control Panel -> System -> Advanced system settings -> Environment Variables.

4. Hit the New User Variable button in the User variables section, then type JAVA_HOME in Variable name and give your jre path in the Variable value. It should look like the below image:
Java path and version may change according to the version of Kafka you are using)

5. Now click OK.

6. Search for a Path variable in the “System Variable” section in the “Environment Variables” dialogue box you just opened.

7. Edit the path and type “;%JAVA_HOME%\bin” at the end of the text already written there,
To confirm the Java installation, just open cmd and type “java –version.”



Kafka uses ZooKeeper to manage the cluster

C. Zookeeper Installation

Go to your Zookeeper config directory. Like:-

1. its C:\zookeeper-3.4.7\conf

2. Rename file “zoo_sample.cfg” to “zoo.cfg”

3. Open zoo.cfg in any text editor, like Notepad; I prefer Notepad++.

4. Find and edit dataDir=F:/TSA/zookeeper/data

5. Add an entry in the System Environment Variables as we did for Java.

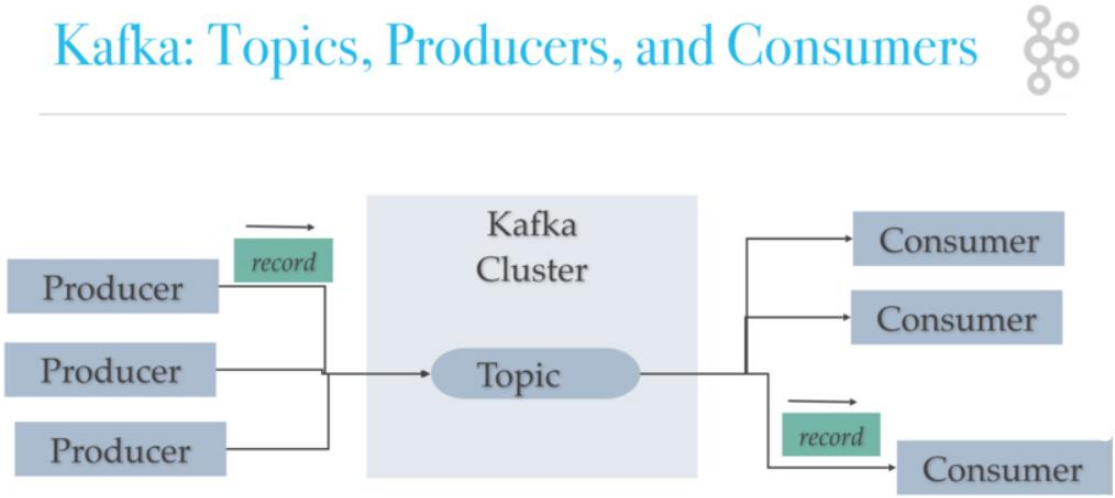
a. Add ZOOKEEPER_HOME = C:\zookeeper-3.4.7 to the System Variables.

b. Edit the System Variable named “Path” and add ;%ZOOKEEPER_HOME%\bin;

6. You can change the default Zookeeper port in zoo.cfg file (Default port 2181).

7. Run Zookeeper by opening a new cmd and type **zkserver**.

Kafka producers write to Topics. Kafka consumers read from Topics



Kafka consists of Records, Topics, Consumers, Producers, Brokers, Logs, Partitions, and Clusters

D. Setting Up Kafka

. Go to your Kafka config directory. For me its C:\kafka_2.11-0.9.0.0\config

2. Edit the file “server.properties.”

3. Find and edit the line log.dirs=/tmp/kafka-logs” to “log.dir= C:\kafka_2.11-0.9.0.0\kafka-logs.

4. If your Zookeeper is running on some other machine or cluster you can edit “zookeeper.connect:2181” to your custom IP and port. For this demo, we are using the same machine so there's no need to change. Also the Kafka port and broker.id are configurable in this file. Leave other settings as is.

5. Your Kafka will run on default port 9092 and connect to Zookeeper’s default port, 2181.

E. Running a Kafka Server

Important: Please ensure that your Zookeeper instance is up and running before starting a Kafka server.

1. Go to your Kafka installation directory: C:\kafka_2.11-0.9.0.0\

2. Open a command prompt here by pressing Shift + right click and choose the “Open command window here” option).

3. Now type

bin\windows\kafka-server-start.bat config\server.properties

and press Enter.

4. Now your Kafka Server is up and running, you can create topics to store messages. Also, we can produce or consume data from Java or Scala code or directly from the command prompt.

F. Creating Topics

1. Now create a topic with the name “test” and a replication factor of 1, as we have only one Kafka server running. If you have a cluster with more than one Kafka server running, you can increase the replication-factor accordingly, which will increase the data availability and act like a fault-tolerant system.

2. Open a new command prompt in the location C:\kafka_2.11-0.9.0.0\bin\windows.

3. Create a topic named **twitterstream** in kafka:

kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic twitterstream

G. Creating a Producer and Consumer to Test Server

<u>To start a producer</u>:

In order to download the tweets from twitter streaming API and push them to kafka queue, we have provided a python script Python **app.py**
you can start downloading tweets from the
twitter stream API and push them to the **twitterstream** topic in Kafka. Do this by running our program as follows:
.
$ python app.py

<u>Now start a consumer by typing the following command:</u>

To check if the data is landing in Kafka:

kafka-console-consumer.bat --bootstrap-server localhost:2181 --topic test --from-beginning

Run the Stream Analysis Program: twitterStream.py

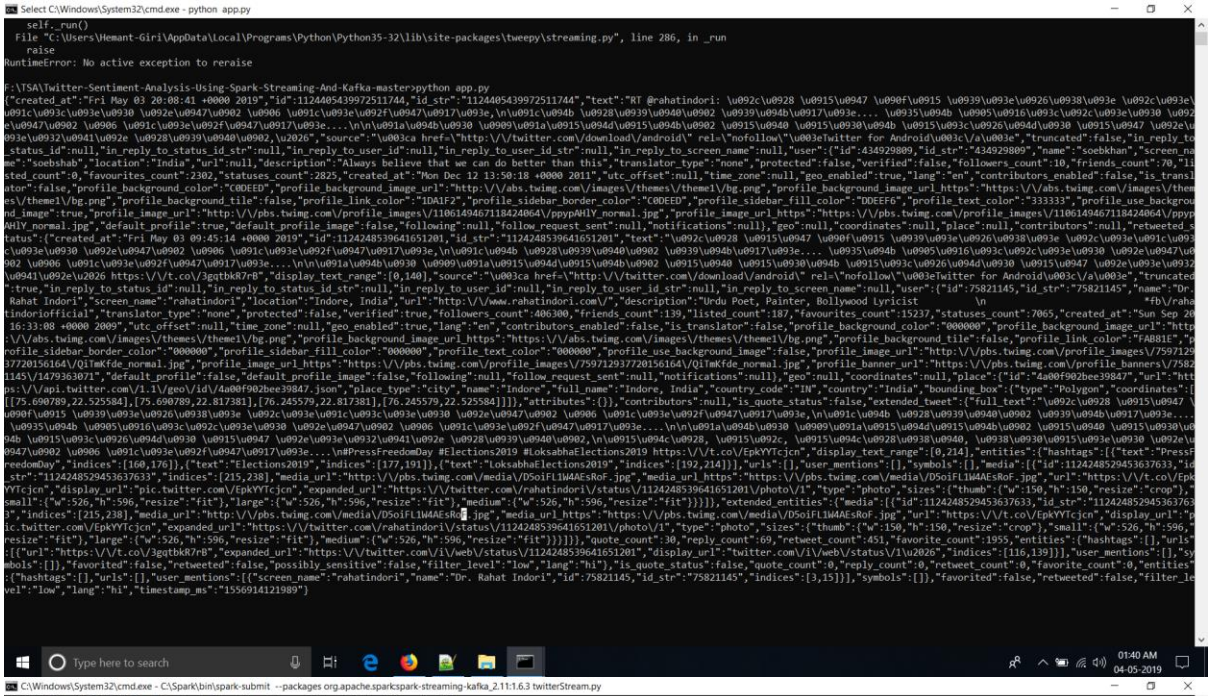
C:\Spark\bin\spark-submit --packages org.apache.spark:spark-streaming-kafka_2.11:1.6.3 twitterStream.py

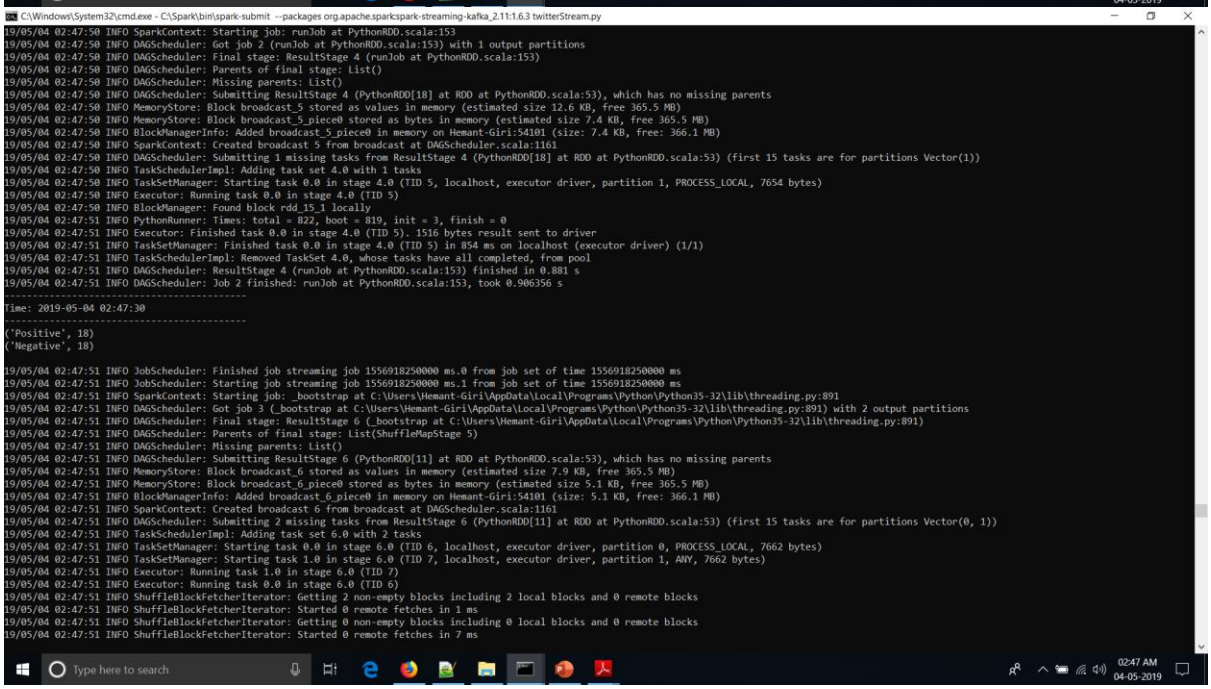
load_wordlist(filename):- This function is used to load the positive words from p ositive.txt and the negative words
from negative.txt. This function needs to return the words as a list or set.

stream(ssc, pwords, nwords, duration):- The *tweets* variable is a DStream object on which you can perform similar transformations that you could create an RDD. Currently, tweets consists of rows of strings, with each row representing one tweet.

We can aggregate the counts for all words. However, we want to combine the positive word counts together and to combine the negative word counts together. For that we can map each word to ("positive", 1) or ("negative", 1) depending on which class that word belongs. Then, the counts can be aggregated using the *reduceByKey* function to get the total counts for "positive" and the total counts for "negative".

updateFunction(newValues, runningCount):- the DStream would store the counts at each time step, not the running total and we want both. To get the running total counts, we use the *updateStateByKeyfunction*.


Select C:\Windows\System32\cmd.exe - python app.py
self._run()
File "C:\Users\Hemant-Giri\AppData\Local\Programs\Python\Python35-32\lib\site-packages\tweepy\streaming.py", line 286, in _run
raise
RuntimeError: No active exception to reraise
F:\TSA\Twitter-Sentiment-Analysis-Using-Spark-Streaming-And-Kafka-master>python app.py
Type here to search
01:40 AM
04-05-2019


C:\Windows\System32\cmd.exe - C:\Spark\bin\spark-submit --packages org.apache.sparkspark-streaming-kafka_2.11:1.6.3 twitterStream.py
19/05/04 02:47:50 INFO SparkContext: Starting job: runJob at PythonRDD.scala:153
19/05/04 02:47:50 INFO DAGScheduler: Got job 2 (runJob at PythonRDD.scala:153) with 1 output partitions
19/05/04 02:47:50 INFO DAGScheduler: Final stage: ResultStage 4 (runJob at PythonRDD.scala:153)
19/05/04 02:47:50 INFO DAGScheduler: Parents of final stage: List()
19/05/04 02:47:50 INFO DAGScheduler: Missing parents: List()
19/05/04 02:47:50 INFO DAGScheduler: Submitting ResultStage 4 (PythonRDD[18] at RDD at PythonRDD.scala:53), which has no missing parents
19/05/04 02:47:50 INFO MemoryStore: Block broadcast_5 stored as values in memory (estimated size 12.6 KB, free 365.5 MB)
19/05/04 02:47:50 INFO MemoryStore: Block broadcast_5_piece0 stored as bytes in memory (estimated size 7.4 KB, free 365.5 MB)
19/05/04 02:47:50 INFO BlockManagerInfo: Added broadcast_5_piece0 in memory on Hemant-Giri:54101 (size: 7.4 KB, free: 366.1 MB)
19/05/04 02:47:50 INFO SparkContext: Created broadcast 5 from broadcast at DAGScheduler.scala:1161
19/05/04 02:47:50 INFO DAGScheduler: Submitting 1 missing tasks from ResultStage 4 (PythonRDD[18] at RDD at PythonRDD.scala:53) (first 15 tasks are for partitions Vector(1))
19/05/04 02:47:50 INFO TaskSchedulerImpl: Adding task set 4.0 with 1 tasks
19/05/04 02:47:50 INFO TaskSetManager: Starting task 0.0 in stage 4.0 (TID 5, localhost, executor driver, partition 1, PROCESS_LOCAL, 7654 bytes)
19/05/04 02:47:50 INFO Executor: Running task 0.0 in stage 4.0 (TID 5)
19/05/04 02:47:50 INFO BlockManager: Found block rdd_15_1 locally
19/05/04 02:47:51 INFO PythonRunner: Times: total = 822, boot = 819, init = 3, finish = 0
19/05/04 02:47:51 INFO Executor: Finished task 0.0 in stage 4.0 (TID 5). 1516 bytes result sent to driver
19/05/04 02:47:51 INFO TaskSetManager: Finished task 0.0 in stage 4.0 (TID 5) in 854 ms on localhost (executor driver) (1/1)
19/05/04 02:47:51 INFO TaskSchedulerImpl: Removed TaskSet 4.0, whose tasks have all completed, from pool
19/05/04 02:47:51 INFO DAGScheduler: ResultStage 4 (runJob at PythonRDD.scala:153) finished in 0.881 s
19/05/04 02:47:51 INFO DAGScheduler: Job 2 finished: runJob at PythonRDD.scala:153, took 0.906356 s
Time: 2019-05-04 02:47:30
('Positive', 18)
('Negative', 18)
19/05/04 02:47:51 INFO JobScheduler: Finished job streaming job 1556918250000 ms.0 from job set of time 1556918250000 ms
19/05/04 02:47:51 INFO JobScheduler: Starting job streaming job 1556918250000 ms.1 from job set of time 1556918250000 ms
19/05/04 02:47:51 INFO SparkContext: Starting job: _bootstrap at C:\Users\Hemant-Giri\AppData\Local\Programs\Python\Python35-32\lib\threading.py:891
19/05/04 02:47:51 INFO DAGScheduler: Got job 3 (_bootstrap at C:\Users\Hemant-Giri\AppData\Local\Programs\Python\Python35-32\lib\threading.py:891) with 2 output partitions
19/05/04 02:47:51 INFO DAGScheduler: Final stage: ResultStage 6 (_bootstrap at C:\Users\Hemant-Giri\AppData\Local\Programs\Python\Python35-32\lib\threading.py:891)
19/05/04 02:47:51 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 5)
19/05/04 02:47:51 INFO DAGScheduler: Missing parents: List()
19/05/04 02:47:51 INFO DAGScheduler: Submitting ResultStage 6 (PythonRDD[11] at RDD at PythonRDD.scala:53), which has no missing parents
19/05/04 02:47:51 INFO MemoryStore: Block broadcast_6 stored as values in memory (estimated size 7.9 KB, free 365.5 MB)
19/05/04 02:47:51 INFO MemoryStore: Block broadcast_6_piece0 stored as bytes in memory (estimated size 5.1 KB, free 365.5 MB)
19/05/04 02:47:51 INFO BlockManagerInfo: Added broadcast_6_piece0 in memory on Hemant-Giri:54101 (size: 5.1 KB, free: 366.1 MB)
19/05/04 02:47:51 INFO SparkContext: Created broadcast 6 from broadcast at DAGScheduler.scala:1161
19/05/04 02:47:51 INFO DAGScheduler: Submitting 2 missing tasks from ResultStage 6 (PythonRDD[11] at RDD at PythonRDD.scala:53) (first 15 tasks are for partitions Vector(0, 1))
19/05/04 02:47:51 INFO TaskSchedulerImpl: Adding task set 6.0 with 2 tasks
19/05/04 02:47:51 INFO TaskSetManager: Starting task 0.0 in stage 6.0 (TID 6, localhost, executor driver, partition 0, PROCESS_LOCAL, 7662 bytes)
19/05/04 02:47:51 INFO TaskSetManager: Starting task 1.0 in stage 6.0 (TID 7, localhost, executor driver, partition 1, ANY, 7662 bytes)
19/05/04 02:47:51 INFO Executor: Running task 1.0 in stage 6.0 (TID 7)
19/05/04 02:47:51 INFO Executor: Running task 0.0 in stage 6.0 (TID 6)
19/05/04 02:47:51 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks including 2 local blocks and 0 remote blocks
19/05/04 02:47:51 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 1 ms
19/05/04 02:47:51 INFO ShuffleBlockFetcherIterator: Getting 0 non-empty blocks including 0 local blocks and 0 remote blocks
19/05/04 02:47:51 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 7 ms
Type here to search
02:47 AM
04-05-2019

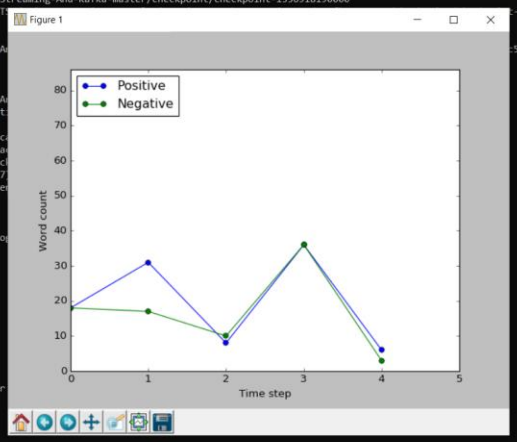
```
C:\Windows\System32\cmd.exe - C:\Spark\bin\spark-submit  --packages org.apache.sparkspark-streaming-kafka_2.11:1.6.3 twitterStream.py
19/05/04 02:48:17 INFO SparkContext: Starting job: runJob at PythonRDD.scala:153
19/05/04 02:48:17 INFO DAGScheduler: Got job 6 (runJob at PythonRDD.scala:153) with 1 output partitions
19/05/04 02:48:17 INFO DAGScheduler: Final stage: ResultStage 11 (runJob at PythonRDD.scala:153)
19/05/04 02:48:17 INFO DAGScheduler: Parents of final stage: List()
19/05/04 02:48:17 INFO DAGScheduler: Missing parents: List()
19/05/04 02:48:17 INFO DAGScheduler: Submitting ResultStage 11 (PythonRDD[43] at RDD at PythonRDD.scala:53), which has no missing parents
19/05/04 02:48:17 INFO MemoryStore: Block broadcast_12 stored as values in memory (estimated size 13.8 KB, free 365.3 MB)
19/05/04 02:48:17 INFO MemoryStore: Block broadcast_12_piece0 stored as bytes in memory (estimated size 8.2 KB, free 365.3 MB)
19/05/04 02:48:17 INFO BlockManagerInfo: Added broadcast_12_piece0 in memory on Hemant-Giri:54101 (size: 8.2 KB, free: 366.1 MB)
19/05/04 02:48:17 INFO SparkContext: Created broadcast 12 from broadcast at DAGScheduler.scala:1161
19/05/04 02:48:17 INFO DAGScheduler: Submitting 1 missing tasks from ResultStage 11 (PythonRDD[43] at RDD at PythonRDD.scala:53) (first 15 tasks are for partitions Vector(1))
19/05/04 02:48:17 INFO TaskSchedulerImpl: Adding task set 11.0 with 1 tasks
19/05/04 02:48:17 INFO TaskSetManager: Starting task 0.0 in stage 11.0 (TID 13, localhost, executor driver, partition 1, PROCESS_LOCAL, 7654 bytes)
19/05/04 02:48:17 INFO Executor: Running task 0.0 in stage 11.0 (TID 13)
19/05/04 02:48:17 INFO BlockManager: Found block rdd_40_1 locally
19/05/04 02:48:18 INFO PythonRunner: Times: total = 877, boot = 874, init = 3, finish = 0
19/05/04 02:48:18 INFO Executor: Finished task 0.0 in stage 11.0 (TID 13). 1516 bytes result sent to driver
19/05/04 02:48:18 INFO TaskSetManager: Finished task 0.0 in stage 11.0 (TID 13) in 911 ms on localhost (executor driver) (1/1)
19/05/04 02:48:18 INFO TaskSchedulerImpl: Removed TaskSet 11.0, whose tasks have all completed, from pool
19/05/04 02:48:18 INFO DAGScheduler: ResultStage 11 (runJob at PythonRDD.scala:153) finished in 0.936 s
19/05/04 02:48:18 INFO DAGScheduler: Job 6 finished: runJob at PythonRDD.scala:153, took 0.956787 s
-------------------------------------------
Time: 2019-05-04 02:48:00
-------------------------------------------
('Positive', 49)
('Negative', 35)

19/05/04 02:48:18 INFO JobScheduler: Finished job streaming job 1556918280000 ms.0 from job set of time 1556918280000 ms
19/05/04 02:48:18 INFO JobScheduler: Starting job streaming job 1556918280000 ms.1 from job set of time 1556918280000 ms
19/05/04 02:48:18 INFO SparkContext: Starting job: _bootstrap at C:\Users\Hemant-Giri\AppData\Local\Programs\Python\Python35-32\lib\threading.py:891
19/05/04 02:48:18 INFO DAGScheduler: Got job 7 (_bootstrap at C:\Users\Hemant-Giri\AppData\Local\Programs\Python\Python35-32\lib\threading.py:891) with 2 output partitions
19/05/04 02:48:18 INFO DAGScheduler: Final stage: ResultStage 13 (_bootstrap at C:\Users\Hemant-Giri\AppData\Local\Programs\Python\Python35-32\lib\threading.py:891)
19/05/04 02:48:18 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 12)
19/05/04 02:48:18 INFO DAGScheduler: Missing parents: List()
19/05/04 02:48:18 INFO DAGScheduler: Submitting ResultStage 13 (PythonRDD[30] at RDD at PythonRDD.scala:53), which has no missing parents
19/05/04 02:48:18 INFO MemoryStore: Block broadcast_13 stored as values in memory (estimated size 7.9 KB, free 365.3 MB)
19/05/04 02:48:18 INFO MemoryStore: Block broadcast_13_piece0 stored as bytes in memory (estimated size 5.1 KB, free 365.3 MB)
19/05/04 02:48:18 INFO BlockManagerInfo: Added broadcast_13_piece0 in memory on Hemant-Giri:54101 (size: 5.1 KB, free: 366.1 MB)
19/05/04 02:48:18 INFO SparkContext: Created broadcast 13 from broadcast at DAGScheduler.scala:1161
19/05/04 02:48:18 INFO DAGScheduler: Submitting 2 missing tasks from ResultStage 13 (PythonRDD[30] at RDD at PythonRDD.scala:53) (first 15 tasks are for partitions Vector(0, 1))
19/05/04 02:48:18 INFO TaskSchedulerImpl: Adding task set 13.0 with 2 tasks
19/05/04 02:48:18 INFO TaskSetManager: Starting task 0.0 in stage 13.0 (TID 14, localhost, executor driver, partition 0, PROCESS_LOCAL, 7662 bytes)
19/05/04 02:48:18 INFO TaskSetManager: Starting task 1.0 in stage 13.0 (TID 15, localhost, executor driver, partition 1, ANY, 7662 bytes)
19/05/04 02:48:18 INFO Executor: Running task 1.0 in stage 13.0 (TID 15)
19/05/04 02:48:18 INFO Executor: Running task 0.0 in stage 13.0 (TID 14)
19/05/04 02:48:18 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks including 2 local blocks and 0 remote blocks
19/05/04 02:48:18 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 1 ms
19/05/04 02:48:18 INFO ShuffleBlockFetcherIterator: Getting 0 non-empty blocks including 0 local blocks and 0 remote blocks
19/05/04 02:48:18 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 3 ms
```

Type here to search

02:48 AM
04-05-2019

```
C:\Windows\System32\cmd.exe - C:\Spark\bin\spark-submit  --packages org.apache.sparkspark-streaming-kafka_2.11:1.6.3 twitterStream.py
19/05/04 02:49:52 INFO BlockManager: Removing RDD 47
19/05/04 02:49:52 INFO JobGenerator: Checkpointing graph for time 1556918370000 ms
19/05/04 02:49:52 INFO DStreamGraph: Updating checkpoint data for time 1556918370000 ms
19/05/04 02:49:52 INFO DStreamGraph: Updated checkpoint data for time 1556918370000 ms
19/05/04 02:49:52 INFO BlockManager: Removing RDD 45
19/05/04 02:49:52 INFO BlockManager: Removing RDD 49
19/05/04 02:49:52 INFO BlockManager: Removing RDD 46
19/05/04 02:49:52 INFO CheckpointWriter: Submitted checkpoint of time 1556918370000 ms to writer queue
19/05/04 02:49:52 INFO CheckpointWriter: Saving checkpoint for time 1556918370000 ms to file 'file:/F:/TSA/Twitter-Sentiment-Analysis-Using-Spark-Streaming-And-Kafka-master/checkpoint/checkpoint-1556918370000'
19/05/04 02:49:52 INFO CheckpointWriter: Deleting file:/F:/TSA/Twitter-Sentiment-Analysis-Using-Spark-Streaming-And-Kafka-master/checkpoint/checkpoint-1556918190000
19/05/04 02:49:52 INFO CheckpointWriter: Checkpoint for time 1556918370000 ms saved to file 'file:/F:/T                                                                                -1556918370000', t
ook 91247 bytes and 121 ms
19/05/04 02:49:52 INFO DStreamGraph: Clearing checkpoint data for time 1556918370000 ms
19/05/04 02:49:52 INFO DStreamCheckpointData: Deleted checkpoint file 'file:/F:/TSA/Twitter-Sentiment-An                                                                                :568c/rdd-65' for t
ime 1556918310000 ms
19/05/04 02:49:52 INFO DStreamGraph: Cleared checkpoint data for time 1556918370000 ms
19/05/04 02:49:52 INFO ReceivedBlockTracker: Deleting batches:
19/05/04 02:49:52 WARN ReceivedBlockTracker: Exception thrown while writing record: BatchCleanupEvent(Ar
java.lang.IllegalStateException: close() was called on BatchedWriteAheadLog before write request with t
        at org.apache.spark.streaming.util.BatchedWriteAheadLog.write(BatchedWriteAheadLog.scala:87)
        at org.apache.spark.streaming.scheduler.ReceivedBlockTracker.writeToLog(ReceivedBlockTracker.sc
        at org.apache.spark.streaming.scheduler.ReceivedBlockTracker.cleanupOldBatches(ReceivedBlockTrac
        at org.apache.spark.streaming.scheduler.ReceiverTracker.cleanupOldBlocksAndBatches(ReceiverTrack
        at org.apache.spark.streaming.scheduler.JobGenerator.clearCheckpointData(JobGenerator.scala:287
        at org.apache.spark.streaming.scheduler.JobGenerator.org$apache$spark$streaming$scheduler$JobGe
        at org.apache.spark.streaming.scheduler.JobGenerator$$anon$1.onReceive(JobGenerator.scala:89)
        at org.apache.spark.streaming.scheduler.JobGenerator$$anon$1.onReceive(JobGenerator.scala:88)
        at org.apache.spark.util.EventLoop$$anon$1.run(EventLoop.scala:49)
19/05/04 02:49:52 WARN ReceivedBlockTracker: Failed to acknowledge batch clean up in the Write Ahead Log
19/05/04 02:49:52 INFO InputInfoTracker: remove old batch metadata: 1556918280000 ms
19/05/04 02:49:52 INFO JobGenerator: Waited for jobs to be processed and checkpoints to be written
19/05/04 02:49:52 INFO CheckpointWriter: CheckpointWriter executor terminated? true, waited for 0 ms.
19/05/04 02:49:52 INFO JobGenerator: Stopped JobGenerator
19/05/04 02:49:52 INFO JobScheduler: Stopped JobScheduler
19/05/04 02:49:52 INFO StreamingContext: StreamingContext stopped successfully
19/05/04 02:49:52 INFO SparkUI: Stopped Spark web UI at http://Hemant-Giri:4040
19/05/04 02:49:52 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
19/05/04 02:49:52 INFO MemoryStore: MemoryStore cleared
19/05/04 02:49:52 INFO BlockManager: BlockManager stopped
19/05/04 02:49:52 INFO BlockManagerMaster: BlockManagerMaster stopped
19/05/04 02:49:52 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator
19/05/04 02:49:52 INFO SparkContext: Successfully stopped SparkContext
19/05/04 02:49:52 INFO SparkContext: SparkContext already stopped.
```



Type here to search

02:52 AM
04-05-2019