CS5542 Big Data Apps and Analytics LAB ASSIGNMENT #5 & #6

- 1. Spark and Smartphone/Watch Application
 - a) Implement a smart application with big data analytics related to your project showing the collaboration between Spark and Smart Apps.
 - b) Implement Twitter Streaming and perform word count on it and publish the results and showcase it in your Smart Phone/Watch Application.

Solution:

Gathered tweets from twitter related to disease and performed a word count program on them and displayed via an android application on a smart phone.

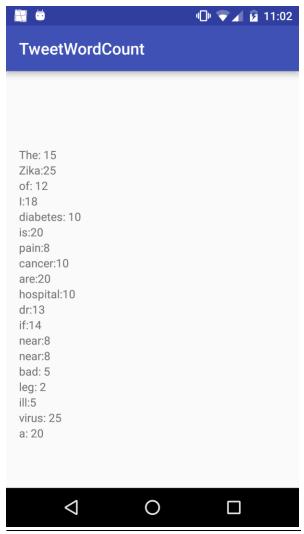


Figure 1.Screen shot

- 2. Spark ML Lib Application Perform a machine learning algorithm with the Twitter Streaming data to categorize each Tweet
 - a) Training datasets:
 Collect different categories of Tweets related to your project.
 (Categories can be based on HashTags /Subjects etc.)
 - b) Test data: the upcoming twitter stream

Solution:

Collected tweets from twitter using following line of code:

```
package com.databricks.apps.twitter_classifier
import java.io.File
import com.google.gson.Gson
import org.apache.spark.streaming.twitter.TwitterUtils
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark.{SparkConf, SparkContext}
* Collect at least the specified number of tweets into json text files.
object Collect {
 private var numTweetsCollected = 0L
 private var partNum = 0
 private var gson = new Gson()
 def main(args: Array[String]) {
    // Process program arguments and set properties
   if (args.length < 3)
     System.err.println("Usage: " + this.getClass.getSimpleName +
        "<outputDirectory> <numTweetsToCollect> <intervalInSeconds> <partitionsEachInterval>")
     System.exit(1)
   val Array(outputDirectory, Utils.IntParam(numTweetsToCollect), Utils.IntParam(intervalSecs), Utils.IntParam(partitionsEachInterval)) =
     Utils.parseCommandLineWithTwitterCredentials(args)
    val outputDir = new File(outputDirectory.toString)
   if (outputDir.exists())
     System.err.println("ERROR - %s already exists: delete or specify another directory".format(
       outputDirectory))
     System.exit(1)
   outputDir.mkdirs()
   println("Initializing Streaming Spark Context...")
   val conf = new SparkConf().setAppName(this.getClass.getSimpleName)
   val sc = new SparkContext(conf)
   val ssc = new StreamingContext(sc, Seconds(intervalSecs))
   val tweetStream = TwitterUtils.createStream(ssc, Utils.getAuth)
      .map(gson.toJson(_))
   tweetStream.foreachRDD((rdd, time) => {
     val count = rdd.count()
     if (count > 0) {
       val outputRDD = rdd.repartition(partitionsEachInterval)
       outputRDD.saveAsTextFile(outputDirectory + "/tweets_" + time.milliseconds.toString)
       numTweetsCollected += count
       if (numTweetsCollected > numTweetsToCollect) {
         System.exit(0)
   })
   ssc.start()
   ssc.awaitTermination()
```

Figure 2. Tweet collection code

```
1
2
3
4 from collections import Counter
5
\stackrel{\cdot}{6} def category_histogram(texts, short_texts):
       # Classify the bios and tweets with MonkeyLearn's topic classifier.
7
       topics = classify batch(texts, MONKEYLEARN TOPIC CLASSIFIER ID)
       # The histogram will keep the counters of how many texts fall in
8
       # a given category.
9
      histogram = Counter()
10
      samples = \{\}
11
       for classification, text, short text in zip (topics, texts, short texts):
12
           # Join the parent and child category names in one string.
category = classification[0]['label'] + '/' + classification[1]['label']
13
           probability = (classification[0]['probability'] *
14
                            classification[1]['probability'])
15
           MIN PROB = 0.3
16
           # Discard texts with a predicted topic with probability lower than a treshold
           if probability < MIN PROB:
17
               continue
18
           # Increment the category counter.
19
           histogram[category] += 1
20
           # Store the texts by category
           samples.setdefault(category, []).append((short_text, text))
21
       returnhistogram, samples
22
23# Classify the expanded tweets using MonkeyLearn, return the histogram
24tweets_histogram, tweets_categorized = category_histogram(expanded_tweets, tweets english)
25 \, \text{\# Classify} the expanded bios of the followed users using MonkeyLearn, return the histogram
26 descriptions histogram, descriptions categorized = category histogram(expanded descriptions,
27<sup>descriptions_english)</sup>
28
29
30
```

Figure 3. Tweet Segregation code