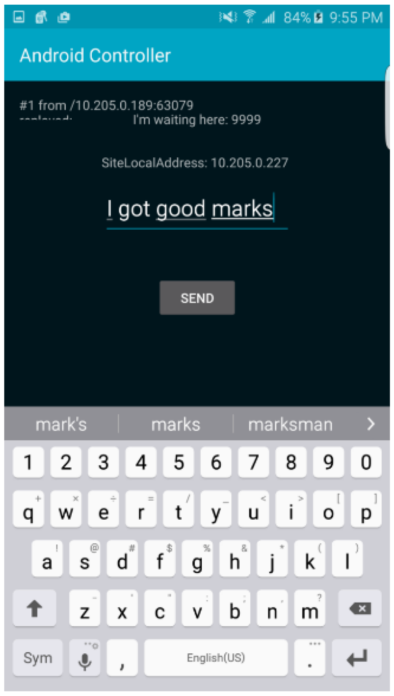
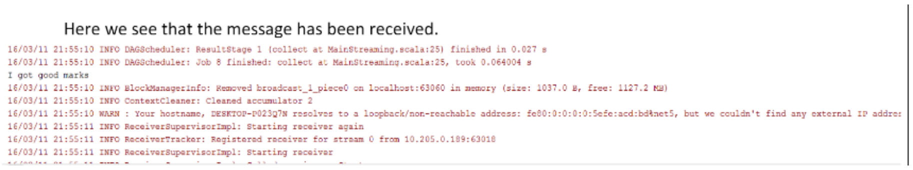
Report for Lab Assignment 5&6

Question 1: Spark and Smartphone/Watch Application

Implement a smart watch application with big data analytics related to your project showing the collaboration between spark and the smart apps. Implement Twitter Streaming and perform word count on it and publish the results and showcase it in your Smart Phone/Watch Application

Screenshots:





Here we see that the message has been received.

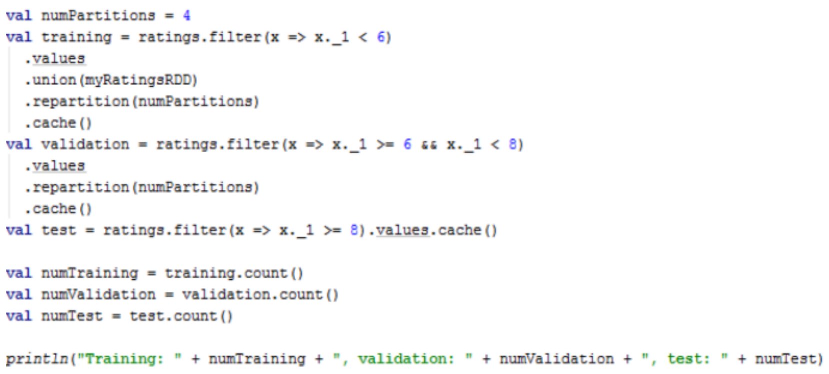
```
16/03/11 21:55:10 INFO DAGScheduler: ResultStage 1 (collect at MainStreaming.scala:25) finished in 0.027 s
16/03/11 21:55:10 INFO DAGScheduler: Job 8 finished: collect at MainStreaming.scala:25, took 0.064004 s
I got good marks
16/03/11 21:55:10 INFO BlockManagerInfo: Removed broadcast_1_piece0 on localhost:63060 in memory (size: 1037.0 B, free: 1127.2 MB)
16/03/11 21:55:10 INFO ContextCleaner: Cleaned accumulator 2
16/03/11 21:55:10 WARN : Your hostname, DESKTOP-P023Q7N resolves to a loopback/non-reachable address: fe80:0:0:0:0:5efe:acd:bd%net5, but we couldn't find any external IP addre
16/03/11 21:55:11 INFO ReceiverSupervisorImpl: Starting receiver again
16/03/11 21:55:11 INFO ReceiverTracker: Registered receiver for stream 0 from 10.205.0.189:63018
16/03/11 21:55:11 INFO ReceiverSupervisorImpl: Starting receiver
```

Question 2: Spark ML Lib Application

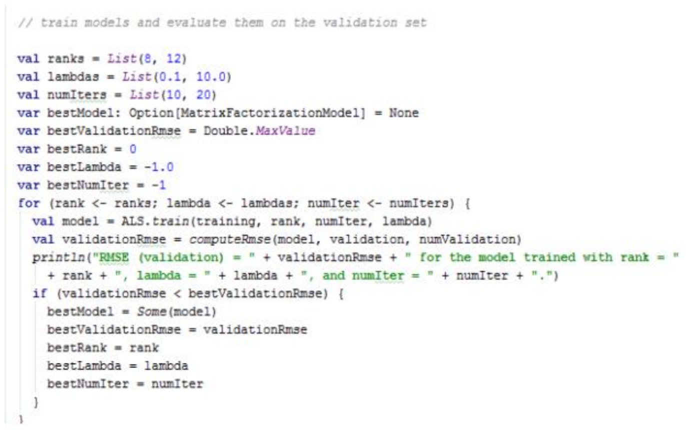
Perform a machine learning algorithm with the Twitter Streaming data to categorize each Tweet

Description: In our project, we use collaborative filtering, which is used here to recommend restaurants among other things.

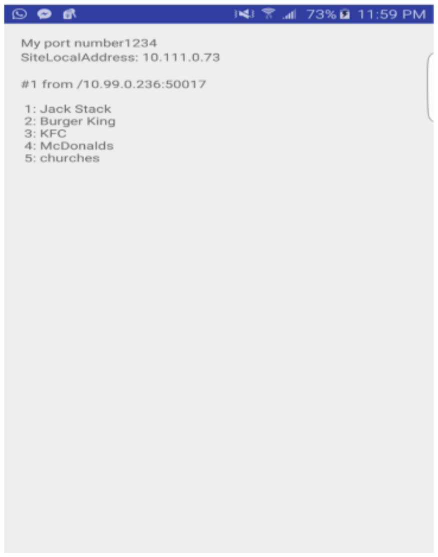
Screenshots:



```
val numPartitions = 4
val training = ratings.filter(x => x._1 < 6)
  .values
  .union(myRatingsRDD)
  .repartition(numPartitions)
  .cache()
val validation = ratings.filter(x => x._1 >= 6 && x._1 < 8)
  .values
  .repartition(numPartitions)
  .cache()
val test = ratings.filter(x => x._1 >= 8).values.cache()

val numTraining = training.count()
val numValidation = validation.count()
val numTest = test.count()

println("Training: " + numTraining + ", validation: " + numValidation + ", test: " + numTest)
```



```
// train models and evaluate them on the validation set

val ranks = List(8, 12)
val lambdas = List(0.1, 10.0)
val numIters = List(10, 20)
var bestModel: Option[MatrixFactorizationModel] = None
var bestValidationRmse = Double.MaxValue
var bestRank = 0
var bestLambda = -1.0
var bestNumIter = -1
for (rank <- ranks; lambda <- lambdas; numIter <- numIters) {
  val model = ALS.train(training, rank, numIter, lambda)
  val validationRmse = computeRmse(model, validation, numValidation)
  println("RMSE (validation) = " + validationRmse + " for the model trained with rank = "
    + rank + ", lambda = " + lambda + ", and numIter = " + numIter + ".")
  if (validationRmse < bestValidationRmse) {
    bestModel = Some(model)
    bestValidationRmse = validationRmse
    bestRank = rank
    bestLambda = lambda
    bestNumIter = numIter
  }
}
```



```
My port number1234
SiteLocalAddress: 10.111.0.73

#1 from /10.99.0.236:50017

 1: Jack Stack
 2: Burger King
 3: KFC
 4: McDonalds
 5: churches
```