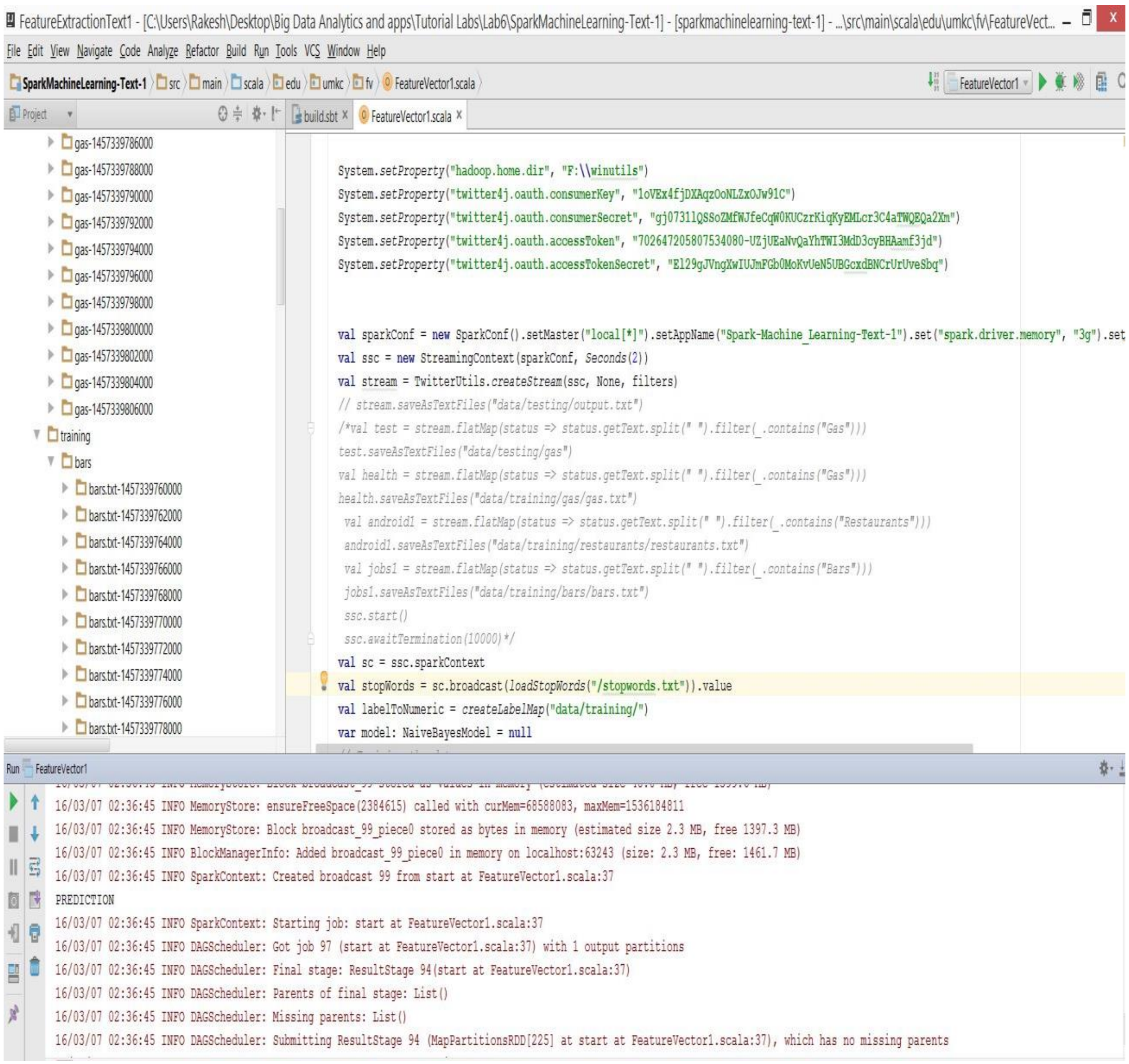
CS 5542 Big Data Applications and Analytics

LAB Assignment #5 & #6

-Sricharan Pamuru (20)

2. Spark ML Lib Application

The below screenshot depicts the training data provided in terms of three constraints, gas, bars and restaurants. Prediction is found in the console and it can be visualized as follows:



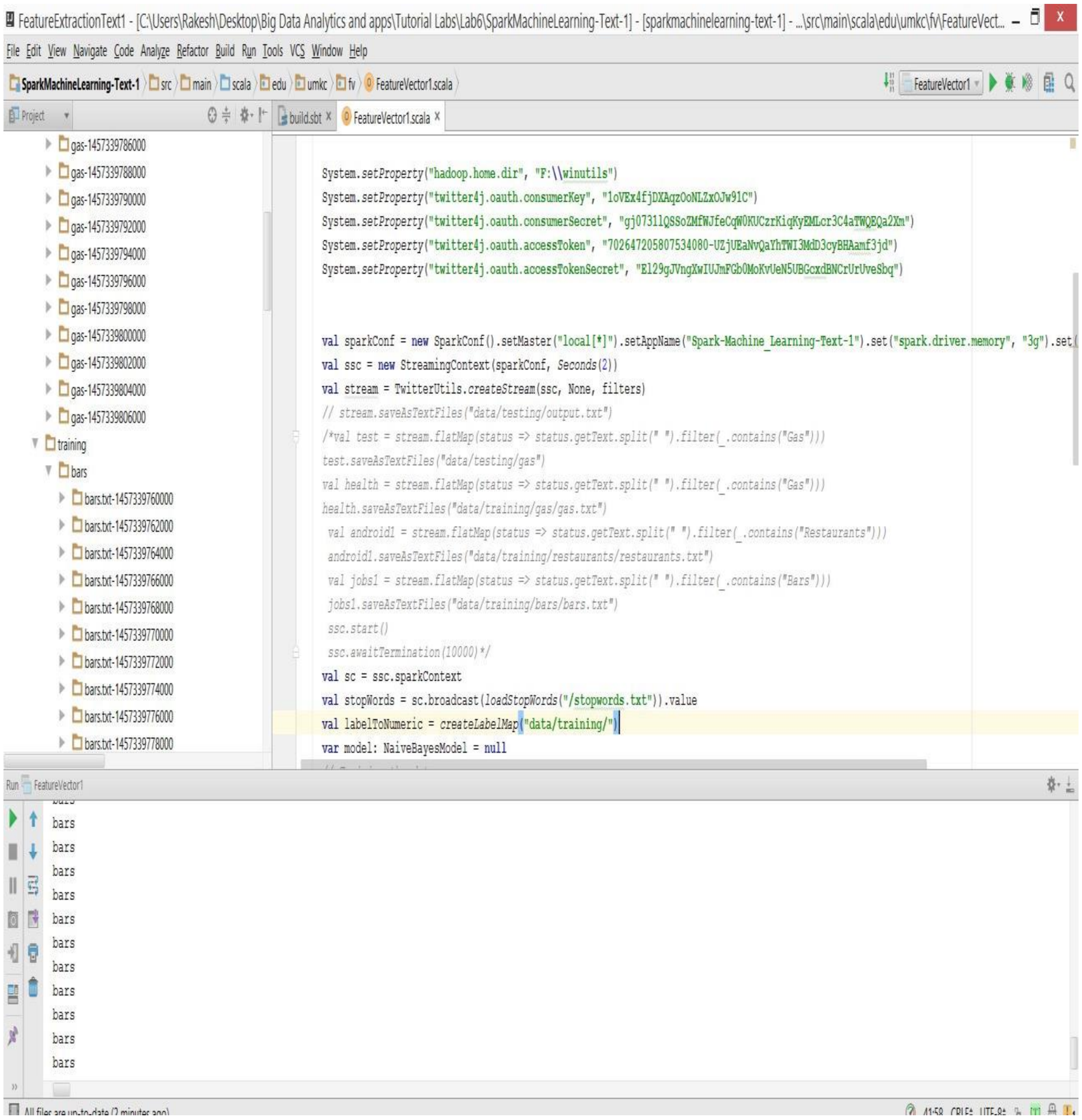
```
System.setProperty("hadoop.home.dir", "F:\\winutils")
System.setProperty("twitter4j.oauth.consumerKey", "1oVEx4fjDXAqzOoNLZxOJw91C")
System.setProperty("twitter4j.oauth.consumerSecret", "gj0731lQSSoZMfWJfeCqW0KUCzrKiqKyEMLcr3C4aTWQEQa2Xm")
System.setProperty("twitter4j.oauth.accessToken", "702647205807534080-UZjUEaNvQaYhTWI3MdD3cyBHAamf3jd")
System.setProperty("twitter4j.oauth.accessTokenSecret", "El29gJVngXwIUJmFGb0MoKvUeN5UBGcxdBNCrUrUveSbq")



val sparkConf = new SparkConf().setMaster("local[*]").setAppName("Spark-Machine_Learning-Text-1").set("spark.driver.memory", "3g").set
val ssc = new StreamingContext(sparkConf, Seconds(2))
val stream = TwitterUtils.createStream(ssc, None, filters)
// stream.saveAsTextFiles("data/testing/output.txt")
/*val test = stream.flatMap(status => status.getText.split(" ").filter(_.contains("Gas")))
test.saveAsTextFiles("data/testing/gas")
val health = stream.flatMap(status => status.getText.split(" ").filter(_.contains("Gas")))
health.saveAsTextFiles("data/training/gas/gas.txt")
 val android1 = stream.flatMap(status => status.getText.split(" ").filter(_.contains("Restaurants")))
 android1.saveAsTextFiles("data/training/restaurants/restaurants.txt")
 val jobs1 = stream.flatMap(status => status.getText.split(" ").filter(_.contains("Bars")))
 jobs1.saveAsTextFiles("data/training/bars/bars.txt")
 ssc.start()
 ssc.awaitTermination(10000)*/
val sc = ssc.sparkContext
val stopWords = sc.broadcast(loadStopWords("/stopwords.txt")).value
val labelToNumeric = createLabelMap("data/training/")
var model: NaiveBayesModel = null
```

```
16/03/07 02:36:45 INFO MemoryStore: ensureFreeSpace(2384615) called with curMem=68588083, maxMem=1536184811
16/03/07 02:36:45 INFO MemoryStore: Block broadcast_99_piece0 stored as bytes in memory (estimated size 2.3 MB, free 1397.3 MB)
16/03/07 02:36:45 INFO BlockManagerInfo: Added broadcast_99_piece0 in memory on localhost:63243 (size: 2.3 MB, free: 1461.7 MB)
16/03/07 02:36:45 INFO SparkContext: Created broadcast 99 from start at FeatureVector1.scala:37
PREDICTION
16/03/07 02:36:45 INFO SparkContext: Starting job: start at FeatureVector1.scala:37
16/03/07 02:36:45 INFO DAGScheduler: Got job 97 (start at FeatureVector1.scala:37) with 1 output partitions
16/03/07 02:36:45 INFO DAGScheduler: Final stage: ResultStage 94(start at FeatureVector1.scala:37)
16/03/07 02:36:45 INFO DAGScheduler: Parents of final stage: List()
16/03/07 02:36:45 INFO DAGScheduler: Missing parents: List()
16/03/07 02:36:45 INFO DAGScheduler: Submitting ResultStage 94 (MapPartitionsRDD[225] at start at FeatureVector1.scala:37), which has no missing parents
```

Prediction output is displayed and the bars are obtained as output based on data collected. Comparison with training data brings a combined result.



```scala
System.setProperty("hadoop.home.dir", "F:\\winutils")
System.setProperty("twitter4j.oauth.consumerKey", "1oVEx4fjDXAqzOoNLZxOJw91C")
System.setProperty("twitter4j.oauth.consumerSecret", "gj07311QSSoZMfWJfeCqW0KUCzrKiqKyEMLcr3C4aTWQEQa2Xm")
System.setProperty("twitter4j.oauth.accessToken", "702647205807534080-UZjUEaNvQaYhTWI3MdD3cyBHAamf3jd")
System.setProperty("twitter4j.oauth.accessTokenSecret", "El29gJVngXwIUJmFGb0MoKvUeN5UBGcxdBNCrUrUveSbq")


val sparkConf = new SparkConf().setMaster("local[*]").setAppName("Spark-Machine_Learning-Text-1").set("spark.driver.memory", "3g").set(
val ssc = new StreamingContext(sparkConf, Seconds(2))
val stream = TwitterUtils.createStream(ssc, None, filters)
// stream.saveAsTextFiles("data/testing/output.txt")
/*val test = stream.flatMap(status => status.getText.split(" ").filter(_.contains("Gas")))
test.saveAsTextFiles("data/testing/gas")
val health = stream.flatMap(status => status.getText.split(" ").filter(_.contains("Gas")))
health.saveAsTextFiles("data/training/gas/gas.txt")
 val android1 = stream.flatMap(status => status.getText.split(" ").filter(_.contains("Restaurants")))
 android1.saveAsTextFiles("data/training/restaurants/restaurants.txt")
 val jobs1 = stream.flatMap(status => status.getText.split(" ").filter(_.contains("Bars")))
 jobs1.saveAsTextFiles("data/training/bars/bars.txt")
 ssc.start()
 ssc.awaitTermination(10000)*/
val sc = ssc.sparkContext
val stopWords = sc.broadcast(loadStopWords("/stopwords.txt")).value
val labelToNumeric = createLabelMap("data/training/")
var model: NaiveBayesModel = null
```

```
bars
bars
bars
bars
bars
bars
bars
bars
bars
bars
bars
```

All files are up-to-date (2 minutes ago)