Project 1- Apache Spark—Real Time Project—Marketing Analysis

Pre-requisites:

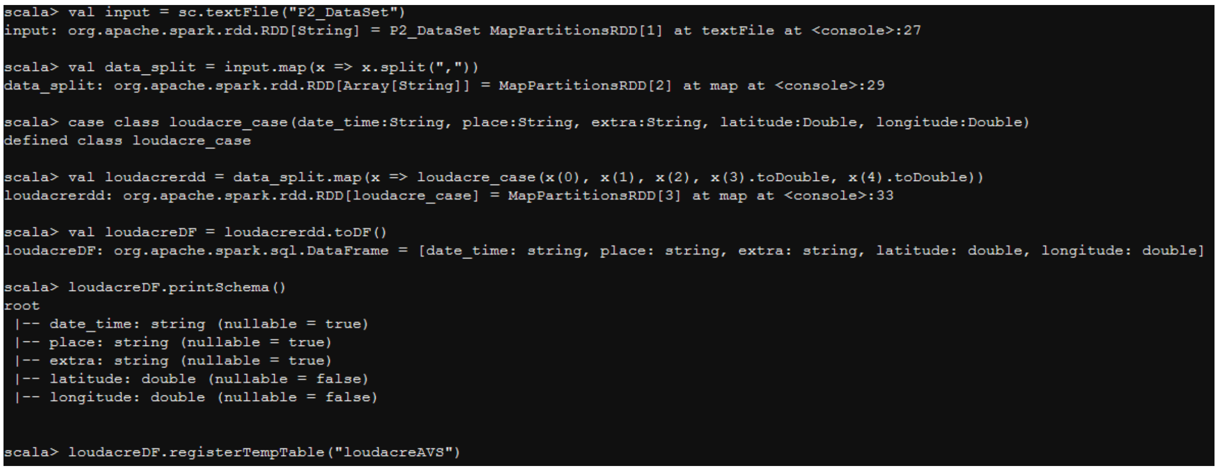
The data set file was uploaded to cloudlab using FTP service.

Then they were uploaded to Hadoop FS using the command:

```
hadoop fs –put P2_DataSet .
```

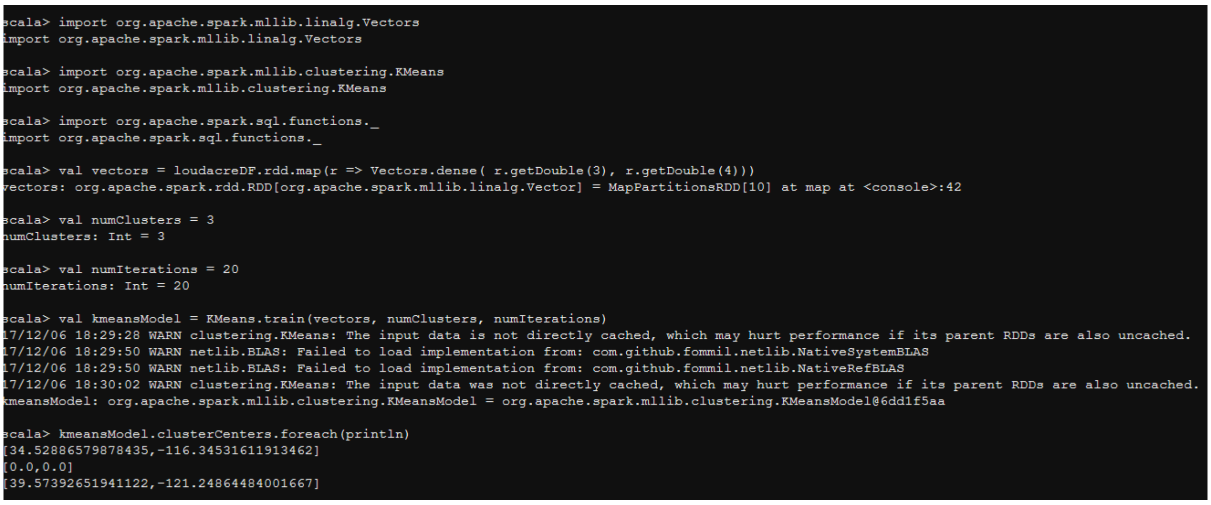
The spark shell is then launched and the data processing starts.

Load data and create Spark data frame



```
scala> val input = sc.textFile("P2_DataSet")
input: org.apache.spark.rdd.RDD[String] = P2_DataSet MapPartitionsRDD[1] at textFile at <console>:27

scala> val data_split = input.map(x => x.split(","))
data_split: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[2] at map at <console>:29

scala> case class loudacre_case(date_time:String, place:String, extra:String, latitude:Double, longitude:Double)
defined class loudacre_case

scala> val loudacrerdd = data_split.map(x => loudacre_case(x(0), x(1), x(2), x(3).toDouble, x(4).toDouble))
loudacrerdd: org.apache.spark.rdd.RDD[loudacre_case] = MapPartitionsRDD[3] at map at <console>:33

scala> val loudacreDF = loudacrerdd.toDF()
loudacreDF: org.apache.spark.sql.DataFrame = [date_time: string, place: string, extra: string, latitude: double, longitude: double]

scala> loudacreDF.printSchema()
root
 |-- date_time: string (nullable = true)
 |-- place: string (nullable = true)
 |-- extra: string (nullable = true)
 |-- latitude: double (nullable = false)
 |-- longitude: double (nullable = false)


scala> loudacreDF.registerTempTable("loudacreAVS")
```

K-Means Working



```
scala> import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.linalg.Vectors

scala> import org.apache.spark.mllib.clustering.KMeans
import org.apache.spark.mllib.clustering.KMeans

scala> import org.apache.spark.sql.functions._
import org.apache.spark.sql.functions._

scala> val vectors = loudacreDF.rdd.map(r => Vectors.dense( r.getDouble(3), r.getDouble(4)))
vectors: org.apache.spark.rdd.RDD[org.apache.spark.mllib.linalg.Vector] = MapPartitionsRDD[10] at map at <console>:42

scala> val numClusters = 3
numClusters: Int = 3

scala> val numIterations = 20
numIterations: Int = 20

scala> val kmeansModel = KMeans.train(vectors, numClusters, numIterations)
17/12/06 18:29:28 WARN clustering.KMeans: The input data is not directly cached, which may hurt performance if its parent RDDs are also uncached.
17/12/06 18:29:50 WARN netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
17/12/06 18:29:50 WARN netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
17/12/06 18:30:02 WARN clustering.KMeans: The input data was not directly cached, which may hurt performance if its parent RDDs are also uncached.
kmeansModel: org.apache.spark.mllib.clustering.KMeansModel = org.apache.spark.mllib.clustering.KMeansModel@6dd1f5aa

scala> kmeansModel.clusterCenters.foreach(println)
[34.52886579878435,-116.34531611913462]
[0.0,0.0]
[39.57392651941122,-121.24864484001667]
```

Solution:

This is the required information of latitude and longitude and the three clusters of users found with k-means algorithm is as below:

[34.52886579878435,-116.34531611913462]

[0.0,0.0]

[39.57392651941122,-121.24864484001667]