Project:Marketing Analysis

1. Load data and create Spark data frame

Code:

I.First FTP the file from your local system to simpliLearn webconsole then

Run following command on webconsole to copy “Project1_DataSet.csv “(renamed from Project 1_dataset_bank-full.csv) to Hadoop file system.

hdfs dfs -put Project1_DataSet.csv /user/adrao45_gmail/

II.Run the following code in scala command line to load data and create Spark data frame

```
import org.apache.spark.sql.Row;

import org.apache.spark.sql.types.{StructType,StructField,StringType,IntegerType};

import sqlContext.implicits._;


val data = sc.textFile("/user/adrao45_gmail/Project1_DataSet.csv")

val header = data.first()

val rddSet = data.filter(x=> x!=header)

val rddFinalSet = rddSet.map(_.split(";").map(_.trim.replace("\"", "")))



val rdds =
rddFinalSet.map(x=>Row(x(0).toInt,x(1).toString,x(2).toString,x(3).toString,x(4).toString,x(5).toInt,x(6).to
String,x(7).toString,x(8).toString,x(9).toInt,x

(10).toString,x(11).toInt,x(12).toInt,x(13).toInt,x(14).toInt,x(15).toString,x(16).toString))



val schema = StructType(List(StructField("age", IntegerType, true),StructField("job",StringType,
true),StructField("martial",StringType, true),StructField
```

("education",StringType, true),StructField("default",StringType, true),StructField("balance",IntegerType, true),StructField("housing",StringType, true),StructField

("loan",StringType, true),StructField("contact",StringType, true),StructField("weekday",IntegerType, true),StructField("month",StringType, true),StructField

("duration",IntegerType, true),StructField("campaign",IntegerType, true),StructField("pdays",IntegerType, true),StructField("previous",IntegerType, true),StructField

("poutcome",StringType, true),StructField("y",StringType, true)))

val df = sqlContext.createDataFrame(rdds,schema)

df.show()

df.registerTempTable("bank")

2. Marketing success rate.

Code:

val success = sqlContext.sql("select (a.subscribed/b.total)*100 as success_percent from (select count(*) as subscribed from bank where y='yes') a,(select count(*) as total from bank) b").show()

2a Give marketing failure rate
Code:
val failure = sqlContext.sql("select (a.not_subscribed/b.total)*100 as failure_percent from (select count(*) as not_subscribed from bank where y='no') a,(select count(*) as total from bank) b").show()

3. Maximum, Mean, and Minimum age of average targeted customer
Code:
df.select(max($"age")).show()
df.select(min($"age")).show()
df.select(avg($"age")).show()

4. Check quality of customers by checking average balance, median balance of customers
Code:
df.select(avg($"balance")).show()
val median = sqlContext.sql("SELECT percentile_approx(balance, 0.5) FROM bank").show()

5. Check if age matters in marketing subscription for deposit

Code:

```
val age = sqlContext.sql("select age, count(*) as number from bank where y='yes' group by age order by
number desc ").show()
```

6. Check if marital status mattered for subscription to deposit.

Code:

```
val marital = sqlContext.sql("select martial, count(*) as number from bank where y='yes' group by martial
order by number desc ").show()
```

7. Check if age and marital status together mattered for subscription to deposit scheme

Code:

```
val Marital_Age = sqlContext.sql("select age, martial, count(*) as number from bank where y='yes' group
by age,martial order by number desc ").show()
```

8. Do feature engineering for column—age and find right age effect on campaign

Code:

```
import org.apache.spark.sql.functions.udf

//Defining a new UDF to divide the age groups into 4 categories.
val ageRDD = sqlContext.udf.register("ageRDD",(age:Int) => {
if (age < 20)
"Teen"
else if (age > 20 && age <= 32)
"Young"
else if (age > 33 && age <= 55)
"Middle Aged"
else
"Old"
})

// Replacing old “age” column with new “age” column
val bankDF = df.withColumn("age",ageRDD(df("age")))
bankDF.registerTempTable("bank1")

// To check the age group which subscribed the most. We will see it is Middle Aged from output.
val Target_AgeGroup = sqlContext.sql("select age, count(*) as number from bank1 where y='yes' group
by age order by number desc ").show()
```