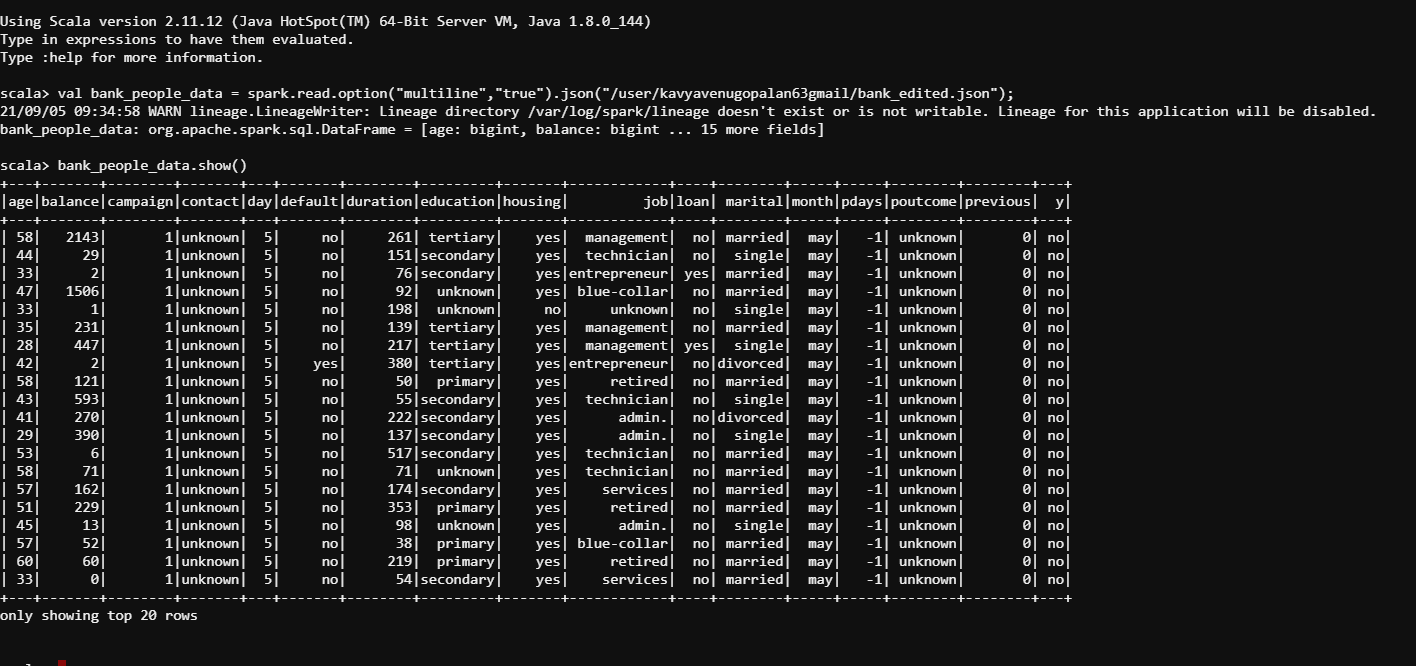
**Market Analysis in Banking Domain**

*//Load data and create a Spark data frame*

val bank\_people\_data= spark.read.option("multiline","true").json("/user/kavyavenugopalan63gmail/Bank\_edited.json");

bank\_people\_data.show()



*//Register the dataframe bank\_people\_data as a temporary table ‘datanewtable’ so that we are able to query the table.*

bank\_people\_data.registerTempTable("datanewtable”)

*//Give the maximum, mean, and minimum age of the average targeted customer*

bank\_people\_data.select(max($"age")).show()

bank\_people\_data.select(min($"age")).show()

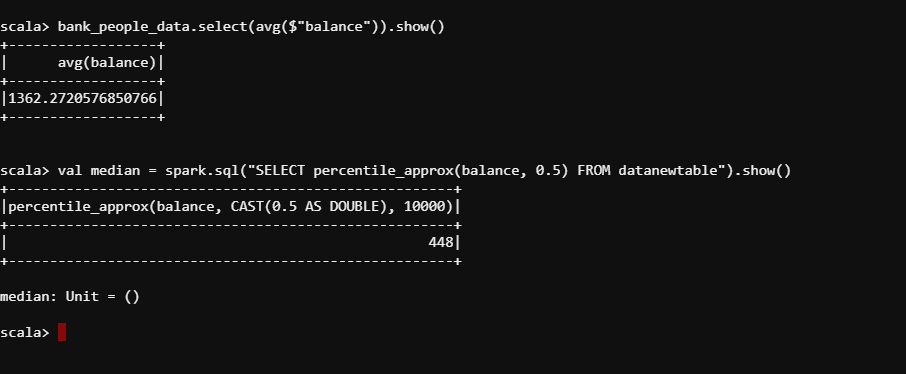
bank\_people\_data.select(avg($"age")).show()



*//Check the quality of customers by checking average balance, median balance of customers*

bank\_people\_data.select(avg($"balance")).show()

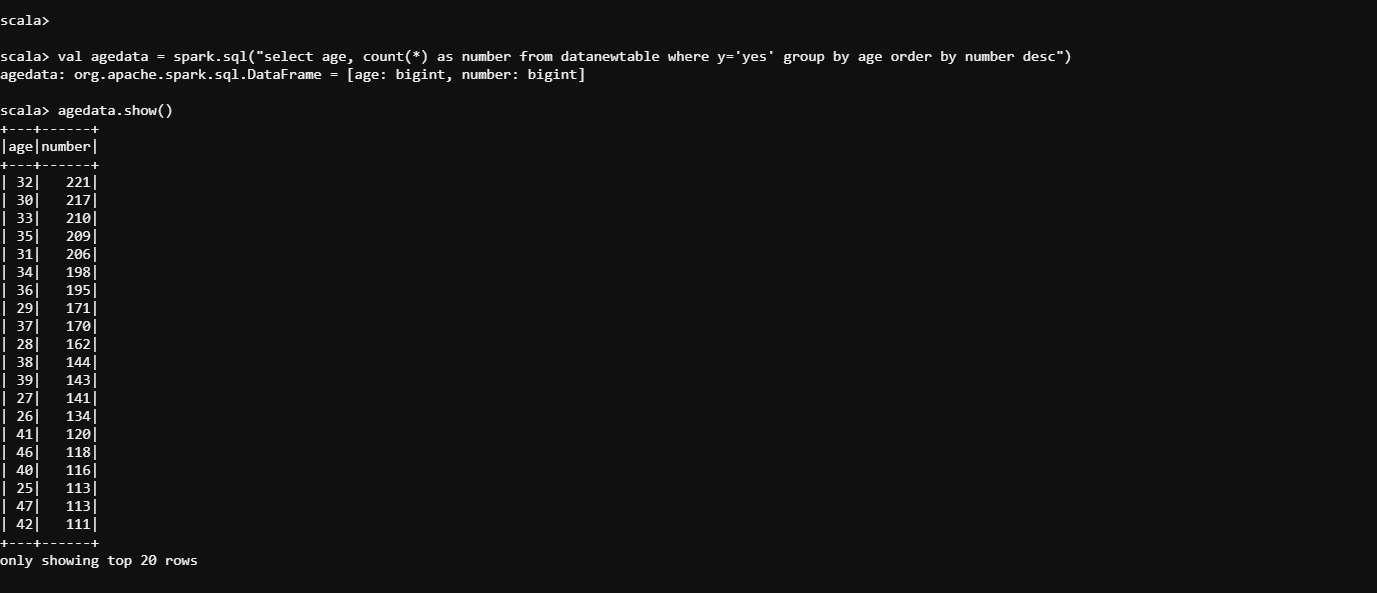
val median = spark.sql("SELECT percentile\_approx(balance, 0.5) FROM datanewtable").show()



*//Check if age matters in marketing subscription for deposit*

val agedata = spark.sql("select age, count(\*) as number from datanewtable where y='yes' group by age order by number desc")

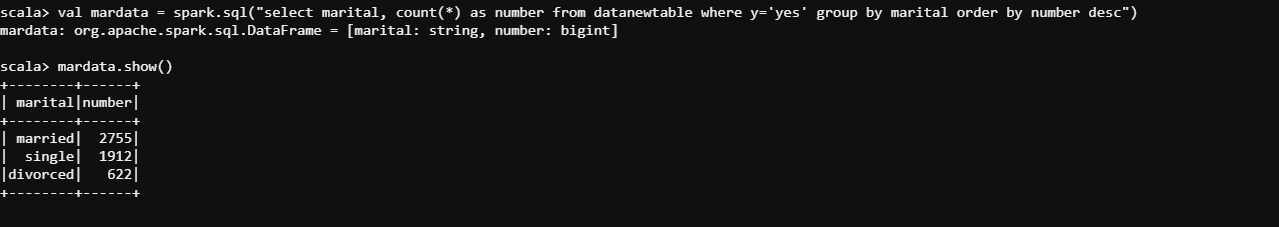
agedata.show()



//Check if marital status mattered for a subscription to deposit

val mardata = spark.sql("select marital, count(\*) as number from datanewtable where y='yes' group by marital order by number desc")

mardata.show()



//Check if age and marital status together mattered for a subscription to deposit scheme

val ageandmaritaldata = spark.sql("select age, marital, count(\*) as number from datanewtable where y='yes' group by age,marital order by number desc")

ageandmaritaldata.show()



//Do feature engineering for the bank and find the right age effect on the campaign

val agedata = spark.udf.register("agedata",(age:Int) => {

if (age < 20)

"Teen"

else if (age > 20 && age <= 32)

"Young"

else if (age > 33 && age <= 55)

"Middle Aged"

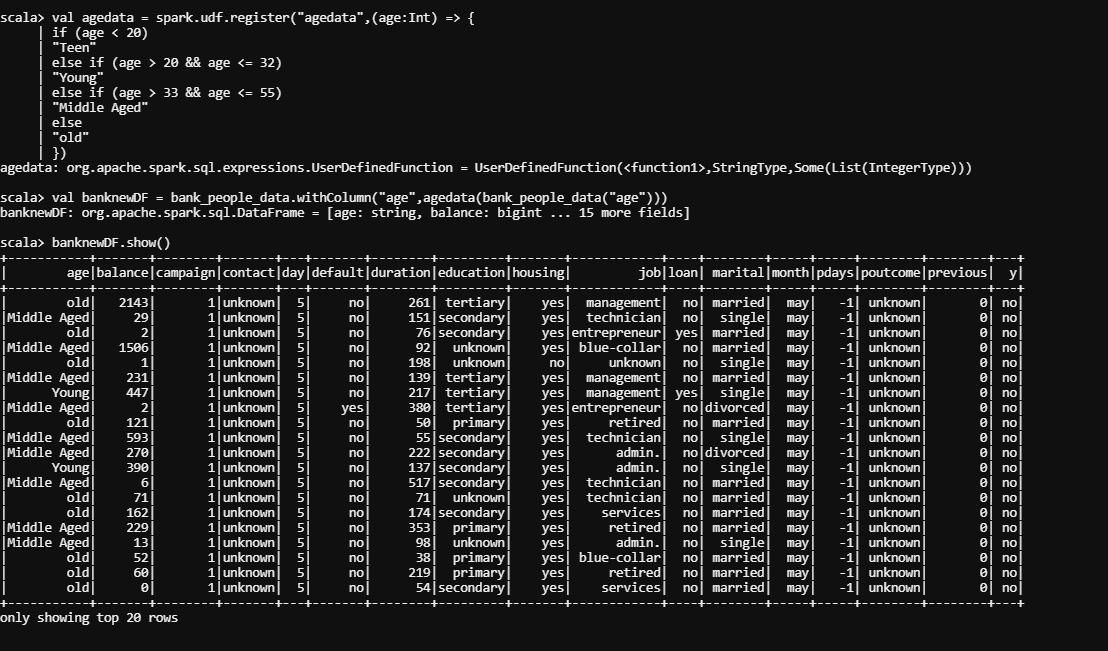
else

"old"

})

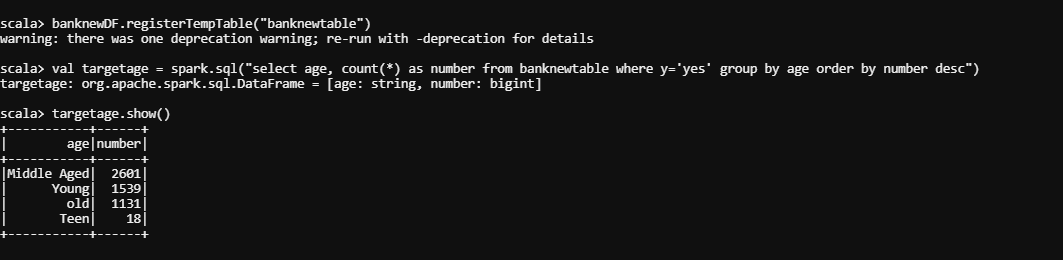
val banknewDF = bank\_people\_data.withColumn("age",agedata(bank\_people\_data("age")))

banknewDF.show()



banknewDF.registerTempTable("banknewtable")

val targetage = spark.sql("select age, count(\*) as number from banknewtable where y='yes' group by age order by number desc")

targetage.show()

import org.apache.spark.ml.Pipeline

import org.apache.spark.ml.classification.LogisticRegression

import org.apache.spark.ml.feature.Bucketizer

import org.apache.spark.ml.feature.Normalizer

import org.apache.spark.ml.feature.StringIndexer

import org.apache.spark.ml.feature.VectorAssembler

import org.apache.spark.mllib.evaluation.BinaryClassificationMetrics

val agedata2 = new StringIndexer().setInputCol("age").setOutputCol("ageindex")

var strindModel = agedata2.fit(banknewDF)

strindModel.transform(banknewDF).select("age","ageIndex").show(5)

