# Project-3: Market Analysis in Banking Domain

**Sachin Dhumale**

## --1)Create Dataframe

val lines = sc.textFile("test/simplilearn/bankmarketingdata.csv") val bank = lines.map(x => x.split(";"))

## --Drop header

val bankf = bank.mapPartitionsWithIndex { (idx, iter) => if (idx == 0) iter.drop(1) else iter }

## --Define Class for the schema

case class Bank(age:Int, job:String, marital:String, education:String, default:String, balance:Int, housing:String, loan:String, contact:String, day:Int, month: String, duration:Int, campaign:Int, pdays:Int, previous:Int, poutcome:String, y:String)

val bankrdd = bankf.map( x => Bank(x(0).toInt,

x(1).replaceAll("\"","")

,x(2).replaceAll("\"","")

,x(3).replaceAll("\"","")

,x(4).replaceAll("\"","")

,x(5).toInt

,x(6).replaceAll("\"","")

,x(7).replaceAll("\"","")

,x(8).replaceAll("\"","")

,x(9).toInt

,x(10).replaceAll("\"","")

,x(11).toInt

,x(12).toInt

,x(13).toInt

,x(14).toInt

,x(15).replaceAll("\"","")

,x(16).replaceAll("\"","")

)

)

val bankDF = bankrdd.toDF()

bankDF.registerTempTable("bank")

## --2) Marketing Success Rate

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()

## --2.a) Marketing Failure Rate

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)Max,Min, Mean age of targeted customer

bankDF.select(max($"age")).show()

bankDF.select(min($"age")).show()

bankDF.select(avg($"age")).show()

## --4) Avg and Median balance of customers

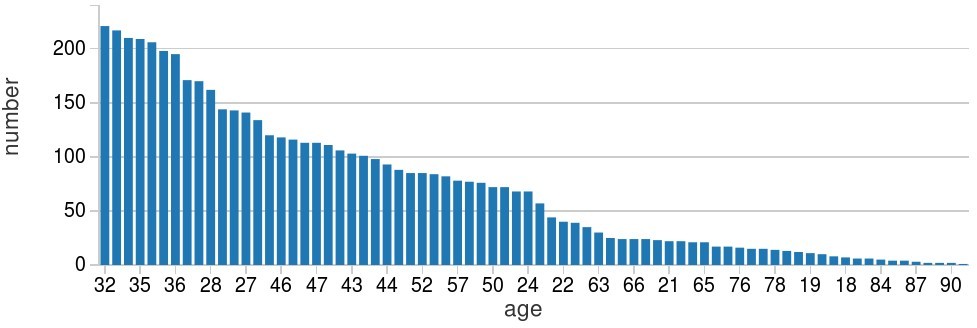
bankDF.select(avg($"balance")).show()

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

## --5)Check if age matters in the marketing subscription for deposit

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

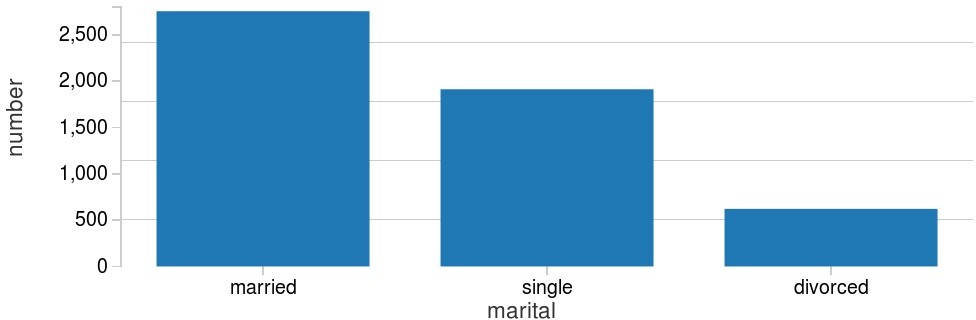
## --We see age indeed matters. The age range between (30-36) shows most promise.



**--6)Check if marital status matters**

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

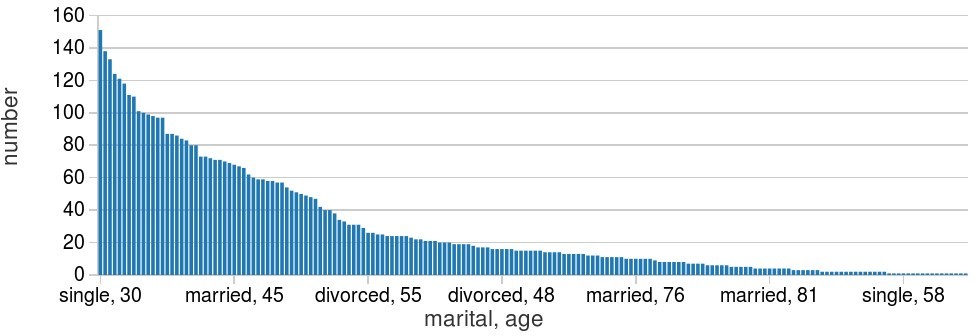
## --We see it’s the married couples who go for the subscriptions the most



**--7) Check if both matters**

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

## --Single people around the age 30-35 shows most subscriptions



**--8) Feature Engineering for cloumn "age"**

**--Import necessary libraries**

import scala.reflect.runtime.universe import org.apache.spark.SparkConf import org.apache.spark.SparkContext

import org.apache.spark.sql.DataFrame import org.apache.spark.sql.SQLContext import org.apache.spark.sql.functions.mean

## --Defining a new UDF with which we will generate new features.We 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"

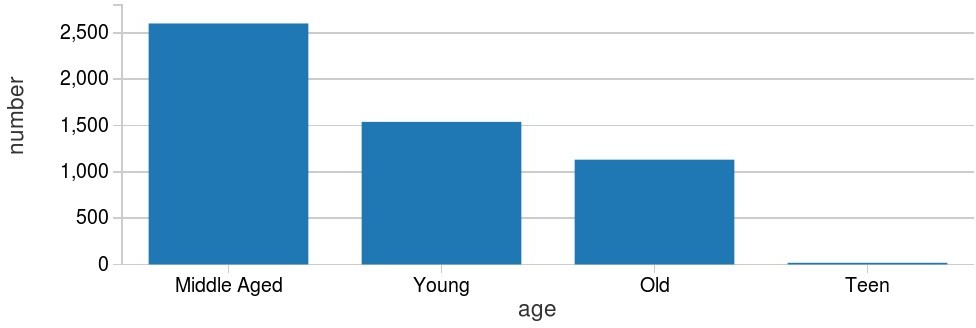
})

## --Replacing old “age” column with new “age” column

val banknewDF = bankDF.withColumn("age",ageRDD(bankDF("age"))) banknewDF.registerTempTable("bank\_new")

## --Running a query to see the age group which subscribed the most. We see it’s ‘Middle-Aged’

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



## --Pipeline

val ageInd = new StringIndexer().setInputCol("age").setOutputCol("ageIndex")

## --Fitting the model

var strIndModel = ageInd.fit(banknewDF)

## --StringIndexerModel.transform() assigns the generated index to each value of the column in the given DataFrame.

**--Middle aged is the most frequent word in this data, so it is given index 0**

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

## --So we can conclude from the Feature Engineering that It is the ‘Middle Aged’ people between age 33 and 55 who should be the targeted customers as they subscribe the most