Marketing Analysis in Banking Domain: HARISH GANESAN

1) **Load data and create a Spark data frame:**

// connecting to spark 2 shell using master local as other users are also connecting to the lab now.

```
[harishga_buffalo@ip-10-0-1-10 ~]$ spark2-shell --master local --packages com.databricks:spark-csv_2.10:1.4.0,com.databricks:spark-xml_2.10:0.4.1
Ivy Default Cache set to: /home/harishga_buffalo/.ivy2/cache
The jars for the packages stored in: /home/harishga_buffalo/.ivy2/jars
:: loading settings :: url = jar:file:/opt/cloudera/parcels/SPARK2-2.4.0.cloudera2-1.cdh5.13.3.p0.1041012/lib/spark2/jars/ivy-2.4.0.jar!/org/apache/i
vy/core/settings/ivysettings.xml
com.databricks#spark-csv_2.10 added as a dependency
com.databricks#spark-xml_2.10 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-dc136772-915c-4152-81dd-220d51c2a42e;1.0
        confs: [default]
        found com.databricks#spark-csv_2.10;1.4.0 in central
        found org.apache.commons#commons-csv;1.1 in central
        found com.univocity#univocity-parsers;1.5.1 in central
        found com.databricks#spark-xml_2.10;0.4.1 in central
:: resolution report :: resolve 312ms :: artifacts dl 5ms
        :: modules in use:
        com.databricks#spark-csv_2.10;1.4.0 from central in [default]
        com.databricks#spark-xml_2.10;0.4.1 from central in [default]
        com.univocity#univocity-parsers;1.5.1 from central in [default]
        org.apache.commons#commons-csv;1.1 from central in [default]
        ---------------------------------------------------------------------
        |                  |            modules            ||   artifacts   |
        |       conf       | number| search|dwnlded|evicted|| number|dwnlded|
        ---------------------------------------------------------------------
        |      default     |   4   |   0   |   0   |   0   ||   4   |   0   |
        ---------------------------------------------------------------------
:: retrieving :: org.apache.spark#spark-submit-parent-dc136772-915c-4152-81dd-220d51c2a42e
        confs: [default]
        0 artifacts copied, 4 already retrieved (0kB/7ms)
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
```

// using spark.read.format and a bunch of options, we are loading the data as a data frame ; If spark.read.format("csv") isn't mentioned then spark would expect a parquet format by default as input

```
Learning on Simplilearn    harishga_buffalo@ip-10-0-1-10:~
sl.cloudloka.com:4200

Type in expressions to have them evaluated.
Type :help for more information.

scala> val df = spark.read.format("csv").option("inferSchema","true").option("header","true").option("delimiter",",").load("/user/ajaykuma24_gmail/co
mmon
<console>:1: error: unclosed string literal
val df = spark.read.format("csv").option("inferSchema","true").option("header","true").option("delimiter",",").load("/user/ajaykuma24_gmail/common
                                                                                                                                          ^

scala> val df = spark.read.format("csv").option("inferSchema","true").option("header","true").option("delimiter",",").load("/user/ajaykuma24_gmail/co
mmon4all/resources/Bank_full.csv")
20/03/15 16:00:48 WARN lineage.LineageWriter: Lineage directory /var/log/spark2/lineage doesn't exist or is not writable. Lineage for this applicatio
n will be disabled.
20/03/15 16:00:48 WARN datasources.DataSource: Multiple sources found for csv (org.apache.spark.sql.execution.datasources.csv.CSVFileFormat, com.data
bricks.spark.csv.DefaultSource15), defaulting to the internal datasource (org.apache.spark.sql.execution.datasources.csv.CSVFileFormat).
20/03/15 16:00:48 WARN datasources.DataSource: Multiple sources found for csv (org.apache.spark.sql.execution.datasources.csv.CSVFileFormat, com.data
bricks.spark.csv.DefaultSource15), defaulting to the internal datasource (org.apache.spark.sql.execution.datasources.csv.CSVFileFormat).
df: org.apache.spark.sql.DataFrame = [age: int, job: string ... 15 more fields]

scala>
```

//Check the Schema of the dataframe

```
scala> val df = spark.read.format("csv").option("inferSchema","true").option("header","true").option("delimiter",","
mmon4all/resources/Bank_full.csv")
20/03/15 16:00:48 WARN lineage.LineageWriter: Lineage directory /var/log/spark2/lineage doesn't exist or is not writ
n will be disabled.
20/03/15 16:00:48 WARN datasources.DataSource: Multiple sources found for csv (org.apache.spark.sql.execution.dataso
bricks.spark.csv.DefaultSource15), defaulting to the internal datasource (org.apache.spark.sql.execution.datasources
20/03/15 16:00:48 WARN datasources.DataSource: Multiple sources found for csv (org.apache.spark.sql.execution.dataso
bricks.spark.csv.DefaultSource15), defaulting to the internal datasource (org.apache.spark.sql.execution.datasources
df: org.apache.spark.sql.DataFrame = [age: int, job: string ... 15 more fields]

scala> df.printSchema
root
 |-- age: integer (nullable = true)
 |-- job: string (nullable = true)
 |-- marital: string (nullable = true)
 |-- education: string (nullable = true)
 |-- default: string (nullable = true)
 |-- balance: integer (nullable = true)
 |-- housing: string (nullable = true)
 |-- loan: string (nullable = true)
 |-- contact: string (nullable = true)
 |-- day: integer (nullable = true)
 |-- month: string (nullable = true)
 |-- duration: integer (nullable = true)
 |-- campaign: integer (nullable = true)
 |-- pdays: integer (nullable = true)
 |-- previous: integer (nullable = true)
 |-- poutcome: string (nullable = true)
 |-- y: string (nullable = true)

scala>
```

Chat

val df = spark.read.format("csv").option ("inferSchema","true").option ("header","true").option("delimiter",",").load ("/user/ajaykuma24_gmail/common 4all/resources/Bank_full.csv")
from Zafar_patel to All Participants:
now it is working
from hubertmachado to All Participants:
yes
from Rajesh.mereddy to All Participants:
not working
from VengalaReddy to All Participants:
Done
from Sangeeta to All Attendees:
am tryng
from Soumo.baral to All Participants:
DOne
from Zafar_patel to All Participants:
i noticed while copying it is putting extra junk character
from Rajesh.mereddy to All Participants:
done
from Harishga to All Participants:
yes
from Ajay Singhal (Faculty) to All Participants:
df.printSchema
from Ajay Singhal (Faculty) to All Participants:
df.show(5)
from Harishga to All Participants:
is there a way to clear in scala console like clear or cls

Send to: All Participants

Send

//Check the top 5 rows of a dataframe to see if the contents are pulled from the source just fine.

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:~ | Webex | Viewing Ajay Singhal (Faculty)'s de...

sl.cloudloka.com:4200

Speak

```
scala> df.show(5)
+---+------------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
|age|         job|marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|  y|
+---+------------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
| 58|  management|married| tertiary|     no|   2143|    yes|  no|unknown|  5|  may|     261|       1|   -1|       0| unknown| no|
| 44|  technician| single|secondary|     no|     29|    yes|  no|unknown|  5|  may|     151|       1|   -1|       0| unknown| no|
| 33|entrepreneur|married|secondary|     no|      2|    yes| yes|unknown|  5|  may|      76|       1|   -1|       0| unknown| no|
| 47| blue-collar|married|  unknown|     no|   1506|    yes|  no|unknown|  5|  may|      92|       1|   -1|       0| unknown| no|
| 33|     unknown| single|  unknown|     no|      1|     no|  no|unknown|  5|  may|     198|       1|   -1|       0| unknown| no|
+---+------------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
only showing top 5 rows
```

// Check the total number of records – by default the result is in long format. We are changing to double

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10: | Webe

sl.cloudloka.com:4200

```
scala> val totCount = df.count()
totCount: Long = 45211

scala>
```

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:

sl.cloudloka.com:4200

```
scala> val totCount = df.count().toDouble
totCount: Double = 45211.0
```

2. **Give marketing success rate (No. of people subscribed / total no. of entries)**

- **Give marketing failure rate**

//Below we see the Success rate and Failure rate both as a percentage

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:~

sl.cloudloka.com:4200

```
scala> val totCount = df.count().toDouble
totCount: Double = 45211.0

scala> val successC = df.filter($"y" === "yes").count().toDouble
successC: Double = 5289.0

scala> val successP = successC / totCount
successP: Double = 0.11698480458295547

scala> val successP = successC / totCount *100
successP: Double = 11.698480458295547

scala> val failP = 1- successP
failP: Double = -10.698480458295547

scala> val failP = 100- successP
failP: Double = 88.30151954170445
```

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:~

sl.cloudloka.com:4200

```
scala> df.groupBy("y").count().show()
+---+-----+
|  y|count|
+---+-----+
| no|39922|
|yes| 5289|
+---+-----+


scala>
```

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

```
scala> df.select(max("age"),min("age")).show()
+--------+--------+
|max(age)|min(age)|
+--------+--------+
|      95|      18|
+--------+--------+


scala> df.select(max("age"),min("age"),mean("age")).show()
+--------+--------+-----------------+
|max(age)|min(age)|         avg(age)|
+--------+--------+-----------------+
|      95|      18|40.93621021432837|
+--------+--------+-----------------+


scala> df.select(max("age"),min("age"),mean("age") as "mean").show()
+--------+--------+-----------------+
|max(age)|min(age)|             mean|
+--------+--------+-----------------+
|      95|      18|40.93621021432837|
+--------+--------+-----------------+

scala> df.select(max("age") as "max",min("age") as "min",mean("age") as "mean").show()
+---+---+-----------------+
|max|min|             mean|
+---+---+-----------------+
| 95| 18|40.93621021432837|
+---+---+-----------------+

scala>
```

Chat

from Harishga to All Participants:
got it
from Prashant Kumar Ojha to All Participants:
pls allow some time
from hubertmachado to All Participants:
got it
from VengalaReddy to All Participants:
Done
from Sunilhmohd to All Attendees:
yes
from Soumo.baral to All Participants:
basically an EDA?
from Harishga to All Participants:
yes
from Ajay Singhal (Faculty) to All Participants:
df.select(max("age"),min("age"),avg("age")).show()
from Harishga to All Participants:
we can also use mean("age") for avg
from Zafar_patel to All Participants:
how to rename the column
from Harishga to All Participants:
correct?
from Harishga to All Participants:
df.select(max("age"),min("age"),mean("age") as "mean").show()
from Harishga to All Participants:
in quotes alias
from Harishga to All Participants:
alias should be in quotes
from Harishga to All Participants:
yes

Send to: All Participants

Send

Harish Ganesan: Market Analysis in Banking Domain

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:~

sl.cloudloka.com:4200

```
scala> df.select("balance").show()
+-------+
|balance|
+-------+
|   2143|
|     29|
|      2|
|   1506|
|      1|
|    231|
|    447|
|      2|
|    121|
|    593|
|    270|
|    390|
|      6|
|     71|
|    162|
|    229|
|     13|
|     52|
|     60|
|      0|
+-------+
only showing top 20 rows


scala>
```

Check the quality of customers by checking average balance, median balance of customers:

Average: Using avg function. We can calculate average either using mean or avg functions. Since we used mean function above, using avg function below

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:~

sl.cloudloka.com:4200

```
scala> df.select(avg("balance")).show()
+------------------+
|      avg(balance)|
+------------------+
|1362.2720576850766|
+------------------+


scala>
```

Median:// by using spark.sql and a percentile function we calculate the median

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10: | Webex | Viewing Ajay Singhal (Faculty)'s de...

sl.cloudloka.com:4200

```
scala> df.select(median("balance")).show()
<console>:26: error: not found: value median
       df.select(median("balance")).show()
                 ^

scala> df.createorReplaceTempView("bankdata")
<console>:26: error: value createorReplaceTempView is not a member of org.apache.spark.sql.DataFrame
       df.createorReplaceTempView("bankdata")
          ^

scala> df.createOrReplaceTempView("bankdata")

scala> spark.sql("select * from bankdata limit 1")
res13: org.apache.spark.sql.DataFrame = [age: int, job: string ... 15 more fields]

scala> spark.sql("select * from bankdata limit 1").show()
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
|age|       job|marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|  y|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
| 58|management|married| tertiary|     no|   2143|    yes|  no|unknown|  5|  may|     261|       1|   -1|       0| unknown| no|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
```

```
scala> spark.sql("select percentile(balance,0.5) as median from bankdata").show()
+------+
|median|
+------+
 448.0|
+------+
```

Check if age matters in marketing subscription for deposit

//exploratory analysis – checking age and y

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10: | Webex

sl.cloudloka.com:4200

```
scala> df.groupBy("age","y").count().show()
+---+---+-----+
|age|  y|count|
+---+---+-----+
| 20| no|   35|
| 78| no|   16|
| 56|yes|   68|
| 28|yes|  162|
| 29|yes|  171|
| 71| no|   29|
| 86|yes|    4|
| 57| no|  750|
| 79|yes|   10|
| 22|yes|   40|
| 42| no| 1131|
| 31|yes|  206|
| 59|yes|   88|
| 87|yes|    3|
| 25| no|  414|
| 34|yes|  198|
| 23|yes|   44|
| 63| no|   47|
| 24| no|  234|
| 64| no|   39|
+---+---+-----+
only showing top 20 rows


scala>
```

Sorting by highest count in the top

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:~ | Webex | Viewing A

sl.cloudloka.com:4200

```
| 64| no|   39|
+---+---+-----+
only showing top 20 rows


scala> df.groupBy("age","y").count().sort($"count".desc).show()
+---+---+-----+
|age|  y|count|
+---+---+-----+
| 32| no| 1864|
| 31| no| 1790|
| 33| no| 1762|
| 34| no| 1732|
| 35| no| 1685|
| 36| no| 1611|
| 30| no| 1540|
| 37| no| 1526|
| 39| no| 1344|
| 38| no| 1322|
| 40| no| 1239|
| 41| no| 1171|
| 42| no| 1131|
| 45| no| 1110|
| 43| no| 1058|
| 46| no| 1057|
| 44| no| 1043|
| 29| no| 1014|
| 47| no|  975|
| 48| no|  915|
+---+---+-----+
only showing top 20 rows


scala>
```

The following indicates that the age matters and is different between the groups as average of no and yes is around ~ approx. 41 and 42 yrs. For now we have simply compared the mean between the groups. However please note that it will be affected by outliers as arithmetic mean is not the best estimate for comparison. We can further analyze these using either geometric mean or median but for now, we are limiting the analysis as the scope of this project is to show our spark skills.

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10: | Web

sl.cloudloka.com:4200

```
scala> df.groupBy("y").count().show()
+---+-----+
|  y|count|
+---+-----+
| no|39922|
|yes| 5289|
+---+-----+


scala> df.groupBy("y").agg(avg("age")).show()
+---+------------------+
|  y|          avg(age)|
+---+------------------+
| no| 40.83898602274435|
|yes|41.670069956513515|
+---+------------------+


scala>
```

Check if marital status mattered for a subscription to deposit

//grouping the column, we get individual group counts	Learning on Simplilearn × harishga_buffalo@ip-10-0-1-10:~ ×
	sl.cloudloka.com:4200
	`cala> df.groupBy("marital").count().show()`
	`--------+-----+`
	`marital|count|`
	`--------+-----+`
	`divorced| 5207|`
	`married|27214|`
	`single|12790|`
	`--------+-----+`

Marital status seems to matter, as we can see the married ppl have the max subscription as well as max non subscription. Divorced has the least and that is valid too as we don't expect many people to be in that group.

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:~ | Webex | Viewing Ajay Singh

sl.cloudloka.com:4200

```
+--------+-----+
|divorced| 5207|
| married|27214|
|  single|12790|
+--------+-----+


scala> df.groupBy("marital","y").count().show()
+--------+---+-----+
| marital|  y|count|
+--------+---+-----+
|divorced|yes|  622|
|  single| no|10878|
|  single|yes| 1912|
|divorced| no| 4585|
| married|yes| 2755|
| married| no|24459|
+--------+---+-----+


scala> df.groupBy("marital","y").count().sort($"count".desc).show()
+--------+---+-----+
| marital|  y|count|
+--------+---+-----+
| married| no|24459|
|  single| no|10878|
|divorced| no| 4585|
| married|yes| 2755|
|  single|yes| 1912|
|divorced|yes|  622|
+--------+---+-----+


scala>
```

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

Since individually as indicated above, they mattered, together too they are significant

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10: | Webex | Viewing Ajay Singhal (Faculty)'s de

sl.cloudloka.com:4200

```
scala> df.groupBy("marital","age","y").count().sort($"count".desc).show()
+-------+---+---+-----+
|marital|age|  y|count|
+-------+---+---+-----+
|married| 34| no| 1013|
|married| 33| no|  978|
|married| 36| no|  976|
|married| 35| no|  976|
|married| 37| no|  975|
|married| 32| no|  920|
| single| 31| no|  906|
|married| 39| no|  873|
| single| 30| no|  861|
|married| 40| no|  856|
|married| 45| no|  825|
|married| 38| no|  819|
| single| 32| no|  817|
|married| 41| no|  803|
|married| 31| no|  801|
|married| 46| no|  772|
|married| 42| no|  770|
|married| 47| no|  743|
|married| 43| no|  743|
|married| 44| no|  734|
+-------+---+---+-----+
only showing top 20 rows


scala>
```

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

//Creating user defined function to categorize age bins

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10:~ | Webex | Viewing Ajay Singhal (Faculty)'s de...

sl.cloudloka.com:4200

```
scala> import.org.apache.spark.sql.functions.udf
<console>:1: error: identifier expected but '.' found.
import.org.apache.spark.sql.functions.udf
      ^
```

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

```
scala>      |      age match {

     |      case n if n <= 30 => "young"

     |      case n if n >= 65 => "old"

     |      case n if n >30 && n <65 => "mid"

     |      }

     |      }
     | ;
<console>:8: error: ')' expected but ';' found.
;
^
```

```
scala> def ageToCategory = udf((age:Int) => { age match { case n  if n <=30 => "young"
     | case n if n >=65 => "old"
     | case n if n >30 && n <65 =>"mid" }})
ageToCategory: org.apache.spark.sql.expressions.UserDefinedFunction
```

```
scala>
```

//We are comparing the newly created dataframe with the past. The age bin is added

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10: | Webex | Viewing Ajay Singhal (Faculty)'s de...

sl.cloudloka.com:4200

```
scala> val newdf = df.withColumn("ageGroup",ageToCategory(df("age")))
newdf: org.apache.spark.sql.DataFrame = [age: int, job: string ... 16 more fields]

scala> df.show(2)
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
|age|       job|marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|  y|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
| 58|management|married| tertiary|     no|   2143|    yes|  no|unknown|  5|  may|     261|       1|   -1|       0| unknown| no|
| 44|technician| single|secondary|     no|     29|    yes|  no|unknown|  5|  may|     151|       1|   -1|       0| unknown| no|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
only showing top 2 rows


scala> newdf.show(2)
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+--------+
|age|       job|marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|  y|ageGroup|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+--------+
| 58|management|married| tertiary|     no|   2143|    yes|  no|unknown|  5|  may|     261|       1|   -1|       0| unknown| no|     mid|
| 44|technician| single|secondary|     no|     29|    yes|  no|unknown|  5|  may|     151|       1|   -1|       0| unknown| no|     mid|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+--------+
only showing top 2 rows


scala> 
```

//From the following summary stats we can see that the mid age group has the largest subscription and it makes more sense because the mid group maybe planning for the future whereas the young might not even be into that working bracket and the old may have some health issues etc or would probably be settled that they don't need to subscribe. So mid group should be targeted by the bank.

Learning on Simplilearn | harishga_buffalo@ip-10-0-1-10: | Webex | Viewing Ajay Singhal (Faculty)'s de...

sl.cloudloka.com:4200

```
scala> df.show(2)
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
|age|       job|marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|  y|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
| 58|management|married| tertiary|     no|   2143|    yes|  no|unknown|  5|  may|     261|       1|   -1|       0| unknown| no|
| 44|technician| single|secondary|     no|     29|    yes|  no|unknown|  5|  may|     151|       1|   -1|       0| unknown| no|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
only showing top 2 rows


scala> newdf.show(2)
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+--------+
|age|       job|marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|  y|ageGroup|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+--------+
| 58|management|married| tertiary|     no|   2143|    yes|  no|unknown|  5|  may|     261|       1|   -1|       0| unknown| no|     mid|
| 44|technician| single|secondary|     no|     29|    yes|  no|unknown|  5|  may|     151|       1|   -1|       0| unknown| no|     mid|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+--------+
only showing top 2 rows
```

```
scala> newdf.groupBy("ageGroup","y").count().sort($"count".desc).show(25)
--------+---+-----+
 ageGroup|  y|count|
--------+---+-----+
      mid| no|33568|
    young| no| 5885|
      mid|yes| 3803|
    young|yes| 1145|
      old| no|  469|
      old|yes|  341|
--------+---+-----+
```

```
scala>
```