

Market Analysis in Banking Domain

1. Starting SPARK SHELL

```

Practice Labs x ajaykumarhorapetigmail@ip-10 x Hue - File Browser x +
< -> C silbdt-webconsole.coresstackio ☆ ⚙️ 🌐
ip-10-0-41-79 login: ajaykumarhorapetigmail
Password:
Last login: Wed Feb 17 13:06:04 on pts/1986
[ajaykumarhorapetigmail@ip-10-0-41-79 ~]$ spark-shell
Setting default log level to "WARN"
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
21/02/18 15:02:50 WARN lineage.LineageWriter: Lineage directory /var/log/spark/lineage doesn't exist or is not writable. Lineage for this application will be disabled.
Spark context available as 'sc' (master = yarn, app id = application_1608530820093_12151).
Spark session available as 'spark'.
Welcome to

  ____  __
 / ___/ /_  __
/ /   / __/ /_
/ /___/ __/ /_
/____/_/ /_

version 2.4.0-cdh6.3.2

Using Scala version 2.11.12 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_144)
Type in expressions to have them evaluated.
Type :help for more information.

scala>

```

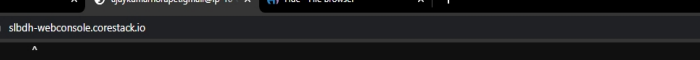
2. Load data and create a Spark data frame.

```
scala> df.printSchema()
root
 |-- serNo: integer (nullable = true)
 |-- age: integer (nullable = true)
 |-- job: string (nullable = true)
 |-- marital: string (nullable = true)
 |-- education: string (nullable = true)
 |-- defaulter: 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> df.show(10)
```

	serNo	age	job	marital	education	defaulter	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
1	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no	
2	technician	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no				
3	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no	
4	47	blue-collar	married	unknown	no	1586	yes	no	unknown	5	may	92	1	-1	0	unknown	no	
5	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no	
6	35	management	married	tertiary	no	231	yes	no	unknown	5	may	139	1	-1	0	unknown	no	
7	28	management	single	tertiary	no	447	yes	yes	unknown	5	may	217	1	-1	0	unknown	no	
8	42	entrepreneur	divorced	tertiary	yes	2	yes	no	unknown	5	may	380	1	-1	0	unknown	no	
9	58	retired	married	primary	no	121	yes	no	unknown	5	may	50	1	-1	0	unknown	no	
10	43	technician	single	secondary	no	593	yes	no	unknown	5	may	55	1	-1	0	unknown	no	

3. Give marketing success rate (No. of people subscribed / total no. of entries)
- Give marketing failure rate



The screenshot shows a web browser with three tabs: 'Practice Labs', 'ajaykumarhorapetigmail@ip-10...', and 'Hue - File Browser'. The active tab is 'Practice Labs', which displays a Scala REPL interface. The browser's address bar shows 'slbhd-webconsole.corestack.io'. The REPL session contains the following code and output:

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

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

scala> val succrate = suscount/totCount * 100
succrate: Double = 11.698488458295547

scala> val failrate = 100 - succrate
failrate: Double = 88.30151954170445

scala>
```

4. Give the maximum, mean, and minimum age of the average targeted customer

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

scala> df.select(min($"age")).show()
+-----+
|min(age)|
+-----+
|    18|
+-----+

scala> df.select(avg($"age")).show()
+-----+
| avg(age)|
+-----+
|40.93621021432837|
+-----+
```

5. Check the quality of customers by checking average balance, median balance of customers

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

scala> df.createOrReplaceTempView("bankdata")

scala> spark.sql("select * from bankdata limit 2").show()
+-----+
|serNo|age|job|marital|education|defaulter|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|y|
+-----+
|1|58|management|married|tertiary|no|2143|yes|no|unknown|5|may|261|1|-1|0|unknown|no|
|2|44|technician|single|secondary|no|29|yes|no|unknown|5|may|151|1|-1|0|unknown|no|
+-----+

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

6. Check if age matters in marketing subscription for deposit

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

7. Check if marital status mattered for a subscription to deposit

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

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

```
scala> df.groupBy("y").agg(count($"marital"), avg($"age")).show()
+-----+
| y | count(marital) |      avg(age) |
+-----+
| no | 39922 | 40.83898602274435 |
| yes | 5289 | 41.670069956513515 |
+-----+
```

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

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

scala> val newdf = df.withColumn("agegroup", agecategory(df("age")))
newdf: org.apache.spark.sql.DataFrame = [serNo: int, age: int ... 17 more fields]

scala> newdf.show(5)
+-----+
| serNo | age | job | marital | education | defaulter | balance | housing | loan | contact | day | month | duration | campaign | pdays | previous | poutcome | y | agegroup |
+-----+
| 1 | 58 | management | married | tertiary | no | 2143 | yes | no | unknown | 5 | may | 261 | 1 | -1 | 0 | unknown | no | mid |
| 2 | 44 | technician | single | secondary | no | 29 | yes | no | unknown | 5 | may | 151 | 1 | -1 | 0 | unknown | no | mid |
| 3 | 33 | entrepreneur | married | secondary | no | 2 | yes | yes | unknown | 5 | may | 76 | 1 | -1 | 0 | unknown | no | mid |
| 4 | 47 | blue-collar | married | unknown | no | 1506 | yes | no | unknown | 5 | may | 92 | 1 | -1 | 0 | unknown | no | mid |
| 5 | 33 | unknown | single | unknown | no | 1 | no | no | unknown | 5 | may | 198 | 1 | -1 | 0 | unknown | no | mid |
+-----+
only showing top 5 rows

scala> newdf.groupBy("agegroup", "y").count().sort($"count".desc).show()
+-----+
| agegroup | y | count |
+-----+
| mid | no | 35146 |
| young | no | 4345 |
| mid | yes | 4041 |
| young | yes | 928 |
| old | no | 431 |
| old | yes | 320 |
+-----+
```

scala> :quit
[ajaykumarhorapet@gmail@ip-10-0-42-218 ~]\$ *****THANK YOU!*****