TP-ML-NANA-ROMARIC-v2

September 16, 2021

1 TRAVAUX PRATIQUES

1.0.1 COURS DE MACHINE LEARNING - UNIVERSITE VIRTUELLE DU BURKINA FASO - MASTER FD & IA

- ETUDIANT : NANA SIDWENDLUIAN ROMARIC
- ENSEIGNANT : MADAME BIRBA ELIANE

1.0.2 1. Chargement des données

[1]:
```
#importing pyspark
import pyspark

#importing sparksession
from pyspark.sql import SparkSession

from pyspark.sql.functions import *

from pyspark.ml.classification import LogisticRegression
from pyspark.ml.evaluation import BinaryClassificationEvaluator

from pyspark.ml.classification import DecisionTreeClassifier

from pyspark.ml.classification import RandomForestClassifier
```

[2]:
```
#creating a sparksession object and providing appName
spark=SparkSession.builder.master("local").appName("tp").getOrCreate()
```

Constatant les resultats mitigés obtenus dans le fichier précédent, nous avons rétiré certaines lignes du dataset qui contenaient des anomalies. Il s'agit notamment des lignes où: * la variable EDUCATION a des valeurs plus grandes que 4 * la variable MARRIAGE a des valeurs 0 (ZÉRO) * les variables PAY_0,PAY_1,PAY_2,….,PAY_6 ont des valeurs -2

[3]:
```
datadft_bis = spark.read.format("csv").options(header=True,inferSchema=True).
↪load("data/ccdefault-bis.csv")
```

1.0.3 2. Analyse exploratoire

[4]:
```
datadft_bis.printSchema()
```

```
root
 |-- ID: integer (nullable = true)
 |-- LIMIT_BAL: integer (nullable = true)
 |-- SEX: integer (nullable = true)
 |-- EDUCATION: integer (nullable = true)
 |-- MARRIAGE: integer (nullable = true)
 |-- AGE: integer (nullable = true)
 |-- PAY_0: integer (nullable = true)
 |-- PAY_2: integer (nullable = true)
 |-- PAY_3: integer (nullable = true)
 |-- PAY_4: integer (nullable = true)
 |-- PAY_5: integer (nullable = true)
 |-- PAY_6: integer (nullable = true)
 |-- BILL_AMT1: integer (nullable = true)
 |-- BILL_AMT2: integer (nullable = true)
 |-- BILL_AMT3: integer (nullable = true)
 |-- BILL_AMT4: integer (nullable = true)
 |-- BILL_AMT5: integer (nullable = true)
 |-- BILL_AMT6: integer (nullable = true)
 |-- PAY_AMT1: integer (nullable = true)
 |-- PAY_AMT2: integer (nullable = true)
 |-- PAY_AMT3: integer (nullable = true)
 |-- PAY_AMT4: integer (nullable = true)
 |-- PAY_AMT5: integer (nullable = true)
 |-- PAY_AMT6: integer (nullable = true)
 |-- DEFAULT: integer (nullable = true)
```

[5]:
```
datadft_bis.count()
```

[5]: 53104

[6]:
```
datadft_bis.where("ID is null").count()
```

[6]: 29946

[7]:
```
datadft_bis.where("ID is not null").count()
```

[7]: 23158

[8]:
```
datadft_bis_clean=datadft_bis.where("ID is not null")
```

[9]:
```
datadft_bis_clean.count()
```

[9]: 23158

[12]:
```
datadft_bis_clean.describe(datadft_bis_clean.columns).show()
```

```
+-------+------------------+------------------+------------------+--------------
----+------------------+-----------------+------------------+-------------------
+-------------------+------------------+------------------+-------------------+-
----------------+-----------------+-----------------+------------------+--------
----------+------------------+-----------------+------------------+-------------
-----+------------------+------------------+------------------+-----------------
-+
|summary|                ID|         LIMIT_BAL|               SEX|
EDUCATION|          MARRIAGE|              AGE|             PAY_0|
PAY_2|              PAY_3|             PAY_4|             PAY_5|
PAY_6|        BILL_AMT1|        BILL_AMT2|        BILL_AMT3|         BILL_AMT4|
BILL_AMT5|         BILL_AMT6|         PAY_AMT1|          PAY_AMT2|
PAY_AMT3|          PAY_AMT4|          PAY_AMT5|          PAY_AMT6|
DEFAULT|
+-------+------------------+------------------+------------------+--------------
----+------------------+-----------------+------------------+-------------------
+-------------------+------------------+------------------+-------------------+-
----------------+-----------------+-----------------+------------------+--------
----------+------------------+-----------------+------------------+-------------
-----+------------------+------------------+------------------+-----------------
-+
|  count|             23158|             23158|             23158|
23158|             23158|            23158|             23158|
23158|              23158|             23158|             23158|
23158|            23158|            23158|            23158|             23158|
23158|             23158|            23158|             23158|
23158|             23158|             23158|             23158|
23158|
|   mean|14916.514509024959|156289.99395457294|1.5915450384316434|1.851412039036
1861|1.5644269798773642|35.24509888591415| 0.180326453061577|0.18874686933241214
|0.17445375248294326|0.1332584851887037|0.0918041281630538|0.08826323516711287|6
1282.07107694965|59663.90612315399|57343.27321012177|53264.461222903534|49786.01
8870368775|47915.041195267295|6075.015243112532| 6121.004836341653|
5562.976941013904| 5089.545815700838| 5049.283875982382|
5334.334312116763|0.2315398566370153|
| stddev| 8614.422528409888|127579.85957421701|0.4915586840970848|0.700640636966
5048|0.5187412618034684|9.291428745301381|0.9848565503867994|
1.0345749644615636| 1.0223398941110264|0.9888780901255524| 0.941911434746279|
0.9498271923186254|77602.34303734612|75193.40105589147|72368.62997241326|
68063.73018666815| 64341.13974498193|63112.234644456585|16902.04758187008|20302.
039638491944|18130.689034411655|15541.648029188553|14947.882729376512|17297.4285
48142714|0.4218255978798094|
|    min|                 2|             10000|                 1|
```

```
0|                  1|                21|                 -1|                   -1|
-1|                 -1|                 -1|                 -1|            -165580|
-67526|           -157264|            -170000|             -81334|
-339603|                  0|                  0|                  0|
0|                  0|                  0|                  0|
|     max|              30000|            1000000|                        2|
4|                  3|                79|                  8|                    8|
8|                  8|                  8|                  8|             964511|
983931|             693131|             891586|             927171|
961664|             873552|            1227082|             896040|
621000|             426529|             528666|                  1|
+--------+------------------+------------------+------------------+--------------
----+------------------+------------------+------------------+-------------------
+------------------+------------------+------------------+------------------+-
-----------------+------------------+------------------+------------------+--------
----------+------------------+------------------+------------------+-------------
-----+------------------+------------------+------------------+-----------------
-+
```

[13]:
```
datadft_bis_clean.describe("LIMIT_BAL",␣
↪"BILL_AMT1","PAY_AMT1","BILL_AMT2","PAY_AMT2").show()
```

```
+--------+------------------+-----------------+------------------+----------------
-+------------------+
|summary|         LIMIT_BAL|        BILL_AMT1|          PAY_AMT1|
BILL_AMT2|          PAY_AMT2|
+--------+------------------+-----------------+------------------+----------------
-+------------------+
|   count|              23158|            23158|             23158|
23158|             23158|
|
mean|156289.99395457294|61282.07107694965|6075.015243112532|59663.90612315399|
6121.004836341653|
| stddev|127579.85957421701|77602.34303734612|16902.04758187008|75193.4010558914
7|20302.039638491944|
|     min|              10000|          -165580|                 0|
-67526|                  0|
|     max|            1000000|           964511|            873552|
983931|            1227082|
+--------+------------------+-----------------+------------------+----------------
-+------------------+
```

[14]:
```
datadft_bis_clean.describe("LIMIT_BAL", "AGE").show()
```

```
+--------+------------------+-----------------+
|summary|         LIMIT_BAL|              AGE|
```

```
+-------+------------------+-----------------+
|  count|             23158|            23158|
|   mean|156289.99395457294|35.24509888591415|
| stddev|127579.85957421701|9.291428745301381|
|    min|             10000|               21|
|    max|           1000000|               79|
+-------+------------------+-----------------+
```

[15]:
```
datadft_bis_clean.groupBy("SEX").count().orderBy(asc("count")).show() # SEXE ?␣
↪HOMME=1 FEMME=2
```

```
+---+-----+
|SEX|count|
+---+-----+
|  1| 9459|
|  2|13699|
+---+-----+
```

[16]:
```
datadft_bis_clean.groupBy("DEFAULT").count().orderBy(asc("count")).show() #␣
↪DÉFAUT DE PAIEMENT ? OUI=1 NON=0
```

```
+-------+-----+
|DEFAULT|count|
+-------+-----+
|      1| 5362|
|      0|17796|
+-------+-----+
```

[17]:
```
datadft_bis_clean.groupBy(['DEFAULT','SEX']).count().orderBy(asc("DEFAULT")).
↪show()
# repartition de la variable cible en fonction du sexe
```

```
+-------+---+-----+
|DEFAULT|SEX|count|
+-------+---+-----+
|      0|  1| 7081|
|      0|  2|10715|
|      1|  2| 2984|
|      1|  1| 2378|
+-------+---+-----+
```

[18]:
```
datadft_bis_clean.groupBy(['DEFAULT','EDUCATION']).count().
↪orderBy(asc("DEFAULT")).show()
# repartition de la variable cible en fonction du niveau d'instruction
```

```
+-------+---------+-----+
|DEFAULT|EDUCATION|count|
+-------+---------+-----+
|      0|        0|    8|
|      0|        1| 6085|
|      0|        2| 8672|
|      0|        3| 2962|
|      0|        4|   69|
|      1|        2| 2847|
|      1|        1| 1479|
|      1|        3| 1033|
|      1|        4|    3|
+-------+---------+-----+
```

[19]:
```
datadft_bis_clean.groupBy(['DEFAULT','MARRIAGE']).count().
 ↪orderBy(asc("DEFAULT")).show()
# repartition de la variable cible en fonction du statut matrimonial
```

```
+-------+--------+-----+
|DEFAULT|MARRIAGE|count|
+-------+--------+-----+
|      0|       1| 7792|
|      0|       2| 9809|
|      0|       3|  195|
|      1|       2| 2724|
|      1|       1| 2564|
|      1|       3|   74|
+-------+--------+-----+
```

[20]:
```
datadft_bis_clean.groupBy(['DEFAULT','AGE']).count().orderBy(asc("DEFAULT")).
 ↪show()
```

```
+-------+---+-----+
|DEFAULT|AGE|count|
+-------+---+-----+
|      0| 42|  452|
|      0| 27|  918|
|      0| 39|  566|
|      0| 31|  708|
|      0| 71|    3|
|      0| 28|  868|
|      0| 56|  104|
|      0| 50|  224|
|      0| 22|  344|
|      0| 58|   62|
|      0| 67|   11|
```

```
|      0| 40|  491|
|      0| 57|   76|
|      0| 32|  699|
|      0| 60|   31|
|      0| 73|    1|
|      0| 65|   17|
|      0| 70|    6|
|      0| 48|  275|
|      0| 25|  722|
+-------+---+-----+
only showing top 20 rows
```

[21]:
```
datadft_bis_clean.groupBy(['DEFAULT','LIMIT_BAL']).count().
 ↪orderBy(desc("DEFAULT")).orderBy(desc("count")).show()
```

```
+-------+---------+-----+
|DEFAULT|LIMIT_BAL|count|
+-------+---------+-----+
|      0|    50000| 2137|
|      0|    20000| 1070|
|      0|    80000|  946|
|      0|    30000|  867|
|      0|   200000|  811|
|      1|    50000|  790|
|      0|   150000|  626|
|      0|   100000|  620|
|      1|    20000|  593|
|      0|   180000|  567|
|      0|    60000|  537|
|      1|    30000|  522|
|      0|   140000|  488|
|      0|    70000|  480|
|      0|   500000|  458|
|      0|   130000|  448|
|      0|   210000|  431|
|      0|   120000|  430|
|      0|   230000|  429|
|      0|   360000|  410|
+-------+---------+-----+
only showing top 20 rows
```

[22]:
```
datadft_bis_clean.createOrReplaceTempView("dataView")
spark.sql("SELECT DEFAULT, avg(LIMIT_BAL) AS BALANCE FROM dataView GROUP BY␣
 ↪DEFAULT ORDER BY BALANCE DESC").show()
```

```
+-------+-----------------+
```

```
|DEFAULT|           BALANCE|
+-------+------------------+
|      0|168451.11260957518|
|      1|115928.32525177173|
+-------+------------------+
```

[23]:
```python
datadft_bis_clean
```

[23]:
```
DataFrame[ID: int, LIMIT_BAL: int, SEX: int, EDUCATION: int, MARRIAGE: int, AGE:
int, PAY_0: int, PAY_2: int, PAY_3: int, PAY_4: int, PAY_5: int, PAY_6: int,
BILL_AMT1: int, BILL_AMT2: int, BILL_AMT3: int, BILL_AMT4: int, BILL_AMT5: int,
BILL_AMT6: int, PAY_AMT1: int, PAY_AMT2: int, PAY_AMT3: int, PAY_AMT4: int,
PAY_AMT5: int, PAY_AMT6: int, DEFAULT: int]
```

1.0.4 3. Preparation des données

Renommons la colonne DEFAULT en label

[24]:
```python
# datadft_bis = datadft.
↪withColumn("ID","LIMIT_BAL","SEX","EDUCATION","MARRIAGE","AGE","PAY_0","PAY_2","PAY_3","PAY
renamedDatadft_bis_clean = datadft_bis_clean.
↪withColumnRenamed("DEFAULT","label")
```

[25]:
```python
renamedDatadft_bis_clean.show(5)
```

```
+---+---------+---+---------+--------+---+-----+-----+-----+-----+-----+-----+--
-------+---------+---------+---------+---------+---------+--------+--------+----
----+--------+--------+--------+-----+
| ID|LIMIT_BAL|SEX|EDUCATION|MARRIAGE|AGE|PAY_0|PAY_2|PAY_3|PAY_4|PAY_5|PAY_6|BI
LL_AMT1|BILL_AMT2|BILL_AMT3|BILL_AMT4|BILL_AMT5|BILL_AMT6|PAY_AMT1|PAY_AMT2|PAY_
AMT3|PAY_AMT4|PAY_AMT5|PAY_AMT6|label|
+---+---------+---+---------+--------+---+-----+-----+-----+-----+-----+-----+--
-------+---------+---------+---------+---------+---------+--------+--------+----
----+--------+--------+--------+-----+
|  2|   120000|  2|        2|       2| 26|   -1|    2|    0|    0|    0|    2|
2682|     1725|     2682|     3272|     3455|     3261|       0|    1000|
1000|    1000|       0|    2000|    1|
|  3|    90000|  2|        2|       2| 34|    0|    0|    0|    0|    0|    0|
29239|    14027|    13559|    14331|    14948|    15549|    1518|    1500|
1000|    1000|    1000|    5000|    0|
|  4|    50000|  2|        2|       1| 37|    0|    0|    0|    0|    0|    0|
46990|    48233|    49291|    28314|    28959|    29547|    2000|    2019|
1200|    1100|    1069|    1000|    0|
|  5|    50000|  1|        2|       1| 57|   -1|    0|   -1|    0|    0|    0|
8617|     5670|    35835|    20940|    19146|    19131|    2000|   36681|
10000|    9000|     689|     679|    0|
|  6|    50000|  1|        1|       2| 37|    0|    0|    0|    0|    0|    0|
64400|    57069|    57608|    19394|    19619|    20024|    2500|    1815|
```

```
657|     1000|     1000|      800|     0|
+---+---------+---+---------+--------+---+-----+-----+-----+-----+-----+--
-------+---------+---------+---------+---------+---------+--------+--------+----
----+--------+--------+--------+-----+
only showing top 5 rows
```

[26]:
```python
#  colonne des etiquettes
colLabel = "label"

#  colonne numerique
colNum = [col for col in renamedDatadft_bis_clean.columns if col!= colLabel]
```

[27]:
```python
colNum
```

[27]:
```
['ID',
 'LIMIT_BAL',
 'SEX',
 'EDUCATION',
 'MARRIAGE',
 'AGE',
 'PAY_0',
 'PAY_2',
 'PAY_3',
 'PAY_4',
 'PAY_5',
 'PAY_6',
 'BILL_AMT1',
 'BILL_AMT2',
 'BILL_AMT3',
 'BILL_AMT4',
 'BILL_AMT5',
 'BILL_AMT6',
 'PAY_AMT1',
 'PAY_AMT2',
 'PAY_AMT3',
 'PAY_AMT4',
 'PAY_AMT5',
 'PAY_AMT6']
```

[28]:
```python
from pyspark.ml.feature import VectorAssembler, StandardScaler

va =  VectorAssembler().setInputCols(colNum).
 ↪setOutputCol("to_be_scaled_features")

featuredDatadft_bis_clean = va.transform(renamedDatadft_bis_clean)
```

```
scaler =  StandardScaler().setInputCol("to_be_scaled_features").
 ↪setOutputCol("features")

dataset_bis = scaler.fit(featuredDatadft_bis_clean).
 ↪transform(featuredDatadft_bis_clean).select("features", "label")

dataset_bis.show(5)
```

```
+--------------------+-----+
|            features|label|
+--------------------+-----+
|[2.32168783618879…|    1|
|[3.48253175428319…|    0|
|[4.64337567237759…|    0|
|[5.80421959047199…|    0|
|[6.96506350856639…|    0|
+--------------------+-----+
only showing top 5 rows
```

1.0.5 4. Application des modèles

[29]:
```
trainSetb, testSetb = dataset_bis.randomSplit([0.8,0.2])
```

[30]:
```
trainSetb.count()
```

[30]: 18494

[31]:
```
testSetb.count()
```

[31]: 4664

[32]:
```
trainSetb.show(5)
```

```
+--------------------+-----+
|            features|label|
+--------------------+-----+
|(24,[0,1,2,3,4,5,…|    0|
|(24,[0,1,2,3,4,5,…|    0|
|(24,[0,1,2,3,4,5,…|    0|
|(24,[0,1,2,3,4,5,…|    0|
|(24,[0,1,2,3,4,5,…|    0|
+--------------------+-----+
only showing top 5 rows
```

1.0.6 4.1 Logistic Regression

[33]:
```
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.evaluation import BinaryClassificationEvaluator

lr_b = LogisticRegression(maxIter=100, regParam=0.0001, elasticNetParam=0.1)
lrModel_b = lr_b.fit(trainSetb)
```

[34]:
```
# trainingSummary = lrModel.summary
print("Coefficients: " + str(lrModel_b.coefficients))
print("Intercept: " + str(lrModel_b.intercept))
```

```
Coefficients: [-0.02094330588103058,-0.17998491808473413,-0.04570017072716318,-0
.03772318212736771,-0.08207283798231185,0.05980937514460258,0.6760420221760369,0
.11032980661501554,0.08180686709890893,0.04774788768784148,0.07102923985889595,0
.07978566469835958,-0.3711038032003112,0.2841025868205868,0.002833425120911844,0
.049224513407792644,0.015627669327713645,-0.015931441855878616,-0.15447222157692
014,-0.1449474191299082 2,-0.007457465858492866,-0.05608250525945112,-0.047916706
055974104,0.02873570956534439]
Intercept: -0.928546893254043
```

[35]:
```
summary = lrModel_b.summary
print("Training set areaUnderROC:",summary.areaUnderROC)
summary.roc.show()
summary.pr.show()
```

```
Training set areaUnderROC: 0.7524071908449781
+--------------------+--------------------+
|                 FPR|                 TPR|
+--------------------+--------------------+
|                 0.0|                 0.0|
|4.907459338194055E-4|0.002600472813238…|
|0.001051598429613…|0.004964539007092199|
|0.001542344363432…|0.007565011820330969|
|0.001752664049355…|0.011111111111111112|
|0.002243409983174425|0.013711583924349883|
|0.002383623107122…|0.017494089834515367|
|0.002734155916993…| 0.020567375886524 82|
|0.003084688726864…| 0.02364066193853428|
|0.003295008412787437|0.027186761229314422|
|0.003715647784632…|0.030023640661938536|
|0.003996074032529445| 0.03333333333333333|
|0.004136287156477846|0.037115839243498816|
|0.004416713404374649| 0.04042553191489362|
|0.004837352776219…| 0.04326241134751773|
|0.005047672462142457| 0.04680851063829787|
| 0.00532809871003926|0.050118203309692674|
|0.005748738081884464|0.052955082742316785|
```

```
|0.005959057767807066|0.056501182033096925|
|0.006449803701626472|  0.0591016548463357|
+--------------------+--------------------+
only showing top 20 rows

+--------------------+------------------+
|              recall|         precision|
+--------------------+------------------+
|                 0.0|0.6111111111111112|
|0.002600472813238…|0.6111111111111112|
|0.004964539007092199|0.5833333333333334|
|0.007565011820330969|0.5925925925925926|
|0.011111111111111112|0.6527777777777778|
|0.013711583924349883|0.6444444444444445|
|0.017494089834515367|0.6851851851851852|
| 0.02056737588652482|0.6904761904761905|
| 0.02364066193853428|0.6944444444444444|
|0.027186761229314422|0.7098765432098766|
|0.030023640661938536|0.7055555555555556|
| 0.03333333333333333|0.7121212121212122|
|0.037115839243498816|0.7268518518518519|
| 0.04042553191489362|0.7307692307692307|
| 0.04326241134751773|0.7261904761904762|
| 0.04680851063829787|0.7333333333333333|
|0.050118203309692674|0.7361111111111112|
|0.052955082742316785|0.7320261437908496|
|0.056501182033096925|0.7376543209876543|
|  0.0591016548463357|0.7309941520467836|
+--------------------+------------------+
only showing top 20 rows
```

[36]:

```python
# make predictions on the test data
lr_predictions_b = lrModel_b.transform(testSetb)
lr_predictions_b.select("prediction", "label", "features").show(25)


lr_evaluator_b = BinaryClassificationEvaluator()
print('Test Area Under ROC', lr_evaluator_b.evaluate(lr_predictions_b))
```

```
+----------+-----+--------------------+
|prediction|label|            features|
+----------+-----+--------------------+
|       0.0|    1|(24,[0,1,2,3,4,5,…|
|       0.0|    0|(24,[0,1,2,3,4,5,…|
|       0.0|    1|(24,[0,1,2,3,4,5,…|
|       0.0|    0|(24,[0,1,2,3,4,5,…|
|       0.0|    0|(24,[0,1,2,3,4,5,…|
```

```
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    1|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    1|[2.32168783618879...|
|       0.0|    0|[0.00139301270171...|
|       0.0|    0|[0.00185735026895...|
|       0.0|    0|[0.00336644736247...|
|       0.0|    0|[0.00359861614609...|
|       0.0|    0|[0.00429512249694...|
|       0.0|    0|[0.00464337567237...|
|       0.0|    0|[0.00499162884780...|
|       0.0|    0|[0.00684897911675...|
|       1.0|    1|[0.00835807621027...|
+----------+-----+--------------------+
only showing top 25 rows

Test Area Under ROC 0.7364789966745253
```

1.0.7 4.2 Decision Tree

[40]:

```python
from pyspark.ml.classification import DecisionTreeClassifier
from pyspark.ml.evaluation import BinaryClassificationEvaluator

dt_b2 = DecisionTreeClassifier().setLabelCol("label").
 ↪setFeaturesCol("features").setMaxDepth(8)

dtModel_b2 = dt_b2.fit(trainSetb)

# make predictions on the test data
dt_predictions_b2 = dtModel_b2.transform(testSetb)
dt_predictions_b2.select("prediction", "label", "features").show(25)

# evaluate the model
dt_evaluator_b2 = BinaryClassificationEvaluator()
print("Test Area Under ROC: " + str(dt_evaluator_b2.evaluate(dt_predictions_b2,␣
 ↪{dt_evaluator_b2.metricName: "areaUnderROC"})))
```

```
+----------+-----+--------------------+
|prediction|label|            features|
+----------+-----+--------------------+
|       0.0|    1|(24,[0,1,2,3,4,5,...|
```

```
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    1|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    1|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    1|[2.32168783618879...|
|       0.0|    0|[0.00139301270171...|
|       0.0|    0|[0.00185735026895...|
|       0.0|    0|[0.00336644736247...|
|       0.0|    0|[0.00359861614609...|
|       0.0|    0|[0.00429512249694...|
|       0.0|    0|[0.00464337567237...|
|       0.0|    0|[0.00499162884780...|
|       1.0|    0|[0.00684897911675...|
|       1.0|    1|[0.00835807621027...|
+----------+-----+--------------------+
only showing top 25 rows

Test Area Under ROC: 0.43740745891175686
```

1.0.8 4.3 Random Forest

[41]:
```python
from pyspark.ml.classification import RandomForestClassifier

rf = RandomForestClassifier(featuresCol = "features", labelCol = "label")
rfModel_b = rf.fit(trainSetb)
rf_predictions_b = rfModel_b.transform(testSetb)
rf_predictions_b.select("prediction", "label", "features").show(25)
```

```
+----------+-----+--------------------+
|prediction|label|            features|
+----------+-----+--------------------+
|       0.0|    1|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    1|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
```

```
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    1|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    0|(24,[0,1,2,3,4,5,...|
|       0.0|    1|[2.32168783618879...|
|       0.0|    0|[0.00139301270171...|
|       0.0|    0|[0.00185735026895...|
|       0.0|    0|[0.00336644736247...|
|       0.0|    0|[0.00359861614609...|
|       0.0|    0|[0.00429512249694...|
|       0.0|    0|[0.00464337567237...|
|       0.0|    0|[0.00499162884780...|
|       0.0|    0|[0.00684897911675...|
|       1.0|    1|[0.00835807621027...|
+----------+-----+--------------------+
only showing top 25 rows
```

[42]:

```
rf_evaluator2 = BinaryClassificationEvaluator(labelCol="label")
accuracy = rf_evaluator2.evaluate(rf_predictions_b)
print("Accuracy = %s" % (accuracy))
print("Test Error = %s" % (1.0 - accuracy))
```

```
Accuracy = 0.7793560340791312
Test Error = 0.22064396592086877
```

[43]:

```
rf_evaluator_b = BinaryClassificationEvaluator()
print("Test Area Under ROC: " + str(rf_evaluator_b.evaluate(rf_predictions_b,
 ↪{rf_evaluator_b.metricName: "areaUnderROC"})))
```

```
Test Area Under ROC: 0.7793560340791312
```