Week7_Lab01_Simonsen

June 22, 2024

[ ]:
```
from pyspark.sql.types import StructType, StructField, StringType , IntegerType,␣
 ↪FloatType, TimestampType, DoubleType
from pyspark.ml.feature import StringIndexer, OneHotEncoder, VectorAssembler,␣
 ↪MinMaxScaler
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.evaluation import BinaryClassificationEvaluator,␣
 ↪MulticlassClassificationEvaluator
from pyspark.sql import functions as F
from pyspark.ml import Pipeline


schema = StructType([
                     StructField('age',DoubleType()),
                     StructField('workclass',StringType()),
                     StructField('fnlwgt',FloatType()),
                     StructField('education',StringType()),
                     StructField('education-num',FloatType()),
                     StructField('marital-status',StringType()),
                     StructField('occupation',StringType()),
                     StructField('relationship',StringType()),
                     StructField('race',StringType()),
                     StructField('sex',StringType()),
                     StructField('capital-gain',FloatType()),
                     StructField('capital-loss',FloatType()),
                     StructField('hours-per-week',FloatType()),
                     StructField('native-country',StringType()),
                     StructField('income',StringType()),


])

census_data = spark.read.csv("dbfs:/FileStore/Merrimack/Week_7/adult.data",␣
 ↪schema=schema)
```

[ ]:
```
trainDF, testDF = census_data.randomSplit([0.8, 0.2], seed=42)

print(trainDF.cache().count())
```

```
print(testDF.count())
```

26076
6485

[ ]:
```
display(trainDF)
```

[ ]:
```
unlist = F.udf(lambda x: round(float(x[0]), 3), DoubleType())

for i in ["age", "fnlwgt", "education-num", "capital-gain", "capital-loss",␣
 ↪"hours-per-week"]:

    #VectorAssembler Transformation - convert column to vector
    assembler = VectorAssembler(inputCols=[i], outputCol=i+"_Vect")

    #MinMaxScaler Transformation
    scaler = MinMaxScaler(inputCol=i+"_Vect", outputCol=i+"_Scaled")

    pipeline = Pipeline(stages=[assembler, scaler])

    trainDF = pipeline.fit(trainDF).transform(trainDF).withColumn(i+"_Scaled",␣
 ↪unlist(i+"_Scaled")).drop(i+"_Vect")
```

```
Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]
```

[ ]:
```
trainDF.printSchema()
```

```
root
 |-- age: double (nullable = true)
 |-- workclass: string (nullable = true)
 |-- fnlwgt: float (nullable = true)
```

```
 |-- education: string (nullable = true)
 |-- education-num: float (nullable = true)
 |-- marital-status: string (nullable = true)
 |-- occupation: string (nullable = true)
 |-- relationship: string (nullable = true)
 |-- race: string (nullable = true)
 |-- sex: string (nullable = true)
 |-- capital-gain: float (nullable = true)
 |-- capital-loss: float (nullable = true)
 |-- hours-per-week: float (nullable = true)
 |-- native-country: string (nullable = true)
 |-- income: string (nullable = true)
 |-- age_Scaled: double (nullable = true)
 |-- fnlwgt_Scaled: double (nullable = true)
 |-- education-num_Scaled: double (nullable = true)
 |-- capital-gain_Scaled: double (nullable = true)
 |-- capital-loss_Scaled: double (nullable = true)
 |-- hours-per-week_Scaled: double (nullable = true)
```

[ ]:
```
trainDF = trainDF.drop(*("age",
                         "fnlwgt",
                         "education-num",
                         "capital-gain",
                         "capital-loss",
                         "hours-per-week"))
```

[ ]:
```
trainDF.display()
```

[ ]:
```
#Repeat steps on test df
unlist = F.udf(lambda x: round(float(x[0]), 3), DoubleType())

for i in ["age", "fnlwgt", "education-num", "capital-gain", "capital-loss",␣
 ↪"hours-per-week"]:

    #VectorAssembler Transformation - convert column to vector
    assembler = VectorAssembler(inputCols=[i], outputCol=i+"_Vect")

    #MinMaxScaler Transformation
    scaler = MinMaxScaler(inputCol=i+"_Vect", outputCol=i+"_Scaled")

    pipeline = Pipeline(stages=[assembler, scaler])

    testDF = pipeline.fit(testDF).transform(testDF).withColumn(i+"_Scaled",␣
 ↪unlist(i+"_Scaled")).drop(i+"_Vect")
```

```
Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]
Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]
```

```
Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]

Downloading artifacts:   0%|          | 0/11 [00:00<?, ?it/s]

Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]
```

[ ]:
```
testDF.printSchema()
```

```
root
 |-- age: double (nullable = true)
 |-- workclass: string (nullable = true)
 |-- fnlwgt: float (nullable = true)
 |-- education: string (nullable = true)
 |-- education-num: float (nullable = true)
 |-- marital-status: string (nullable = true)
 |-- occupation: string (nullable = true)
 |-- relationship: string (nullable = true)
 |-- race: string (nullable = true)
 |-- sex: string (nullable = true)
 |-- capital-gain: float (nullable = true)
 |-- capital-loss: float (nullable = true)
 |-- hours-per-week: float (nullable = true)
 |-- native-country: string (nullable = true)
 |-- income: string (nullable = true)
 |-- age_Scaled: double (nullable = true)
 |-- fnlwgt_Scaled: double (nullable = true)
 |-- education-num_Scaled: double (nullable = true)
 |-- capital-gain_Scaled: double (nullable = true)
 |-- capital-loss_Scaled: double (nullable = true)
 |-- hours-per-week_Scaled: double (nullable = true)
```

[ ]:
```
testDF = testDF.drop(*("age",
                       "fnlwgt",
                       "education-num",
                       "capital-gain",
                       "capital-loss",
                       "hours-per-week"))
```

[ ]:
```
testDF.show()
```

```
+----------+---------+--------------+---------------+------------+------+-------
+----------------+------+----------+-------------+--------------------+---------
----------+------------------+---------------------+
| workclass|education|marital-status|     occupation|relationship|  race|
sex|  native-country|income|age_Scaled|fnlwgt_Scaled|education-
num_Scaled|capital-gain_Scaled|capital-loss_Scaled|hours-per-week_Scaled|
+----------+---------+--------------+---------------+------------+------+-------
+----------------+------+----------+-------------+--------------------+---------
----------+------------------+---------------------+
|         ?|     11th| Never-married|              ?|   Own-child| White|
Female|   United-States| <=50K|       0.0|        0.019|                 0.4|
0.0|               0.0|                0.143|
|         ?|     10th| Never-married|              ?|   Own-child| White|
Male|   United-States| <=50K|       0.0|        0.035|               0.333|
0.0|               0.0|                0.296|
|         ?|     11th| Never-married|              ?|   Own-child| White|
Female|   United-States| <=50K|       0.0|        0.046|                 0.4|
0.0|               0.0|                0.194|
|         ?|     11th| Never-married|              ?|   Own-child| White|
Male|   United-States| <=50K|       0.0|        0.063|                 0.4|
0.0|               0.0|                0.173|
|         ?|     10th| Never-married|              ?|   Own-child| White|
Female|   United-States| <=50K|       0.0|        0.087|               0.333|
0.0|               0.0|                0.143|
|         ?|  HS-grad| Never-married|              ?|   Own-child| Black|
Male|   United-States| <=50K|       0.0|        0.094|               0.533|
0.0|               0.0|                0.398|
|         ?|     11th| Never-married|              ?|   Own-child| White|
Female|   United-States| <=50K|       0.0|        0.109|                 0.4|
0.0|               0.0|                0.071|
|         ?|     11th| Never-married|              ?|   Own-child| White|
Male|            Peru| <=50K|       0.0|        0.128|                 0.4|
0.0|               0.0|                0.194|
|         ?|     12th| Never-married|              ?|   Own-child| Other|
Female|   United-States| <=50K|       0.0|        0.158|               0.467|
0.0|               0.0|                0.398|
|         ?|     11th| Never-married|              ?|   Own-child| Black|
Male|   United-States| <=50K|       0.0|        0.164|                 0.4|
0.0|               0.0|                0.071|
|         ?|     10th| Never-married|              ?|   Own-child| White|
Female|   United-States| <=50K|       0.0|        0.168|               0.333|
0.0|               0.0|                0.143|
|         ?|     10th| Never-married|              ?|   Own-child| White|
Male|   United-States| <=50K|       0.0|        0.172|               0.333|
0.0|               0.0|                0.071|
|         ?|     10th| Never-married|              ?|   Own-child| White|
```

```
Female|   United-States| <=50K|       0.0|        0.185|               0.333|
0.0|               0.0|               0.398|
|         ?|     10th| Never-married|             ?|   Own-child| White|
Female|   United-States| <=50K|       0.0|        0.221|               0.333|
0.0|               0.0|               0.031|
|         ?|     11th| Never-married|             ?|   Own-child| Black|
Female| Trinadad&Tobago| <=50K|       0.0|        0.448|                 0.4|
0.0|               0.0|               0.398|
| Local-gov|     11th| Never-married| Prof-specialty|   Own-child| Black|
Male|   United-States| <=50K|       0.0|        0.111|                 0.4|
0.0|               0.0|               0.143|
| Local-gov|     11th| Never-married|   Adm-clerical|   Own-child| White|
Female|   United-States| <=50K|       0.0|        0.205|                 0.4|
0.0|               0.0|               0.143|
|   Private|     11th| Never-married|  Other-service|   Own-child| White|
Female|   United-States| <=50K|       0.0|        0.023|                 0.4|
0.0|               0.0|               0.143|
|   Private|     11th| Never-married|          Sales|   Own-child| White|
Male|   United-States| <=50K|       0.0|         0.03|                 0.4|
0.0|               0.0|               0.296|
|   Private|     11th| Never-married|   Craft-repair|   Own-child| White|
Male|   United-States| <=50K|       0.0|         0.04|                 0.4|
0.0|               0.0|               0.153|
+----------+---------+--------------+---------------+------------+------+-------
+----------------+------+----------+-------------+--------------------+---------
----------+------------------+---------------------+
only showing top 20 rows
```

[ ]:
```
#Determine if downampling is needed
trainDF.groupBy("income").count().show()
```

```
+------+-----+
|income|count|
+------+-----+
|  >50K| 6264|
| <=50K|19812|
+------+-----+
```

[ ]:
```
#downsampling if needed, commented out for now.
major_df = trainDF.filter(trainDF["income"] == " <=50K")
minor_df = trainDF.filter(trainDF["income"] == " >50K")


ratio = int(major_df.count()/minor_df.count())

sampled_majority_df = trainDF.sample(False, 1/ratio)
```

```
trainDF_2 = sampled_majority_df.unionAll(minor_df)
trainDF_2.show()
```

```
+----------+---------+--------------+----------------+--------------+------+---
----+--------------+------+----------+-------------+--------------------+-------
------------+------------------+---------------------+
| workclass|education|marital-status|      occupation|  relationship|  race|
sex|native-country|income|age_Scaled|fnlwgt_Scaled|education-num_Scaled|capital-
gain_Scaled|capital-loss_Scaled|hours-per-week_Scaled|
+----------+---------+--------------+----------------+--------------+------+---
----+--------------+------+----------+-------------+--------------------+-------
------------+------------------+---------------------+
|         ?|     12th| Never-married|               ?|     Own-child| White|
Female| United-States| <=50K|       0.0|        0.015|               0.467|
0.0|               0.0|               0.245|
|         ?|     11th| Never-married|               ?|     Own-child| Black|
Female| United-States| <=50K|       0.0|        0.025|                 0.4|
0.0|               0.0|               0.398|
|         ?|     10th| Never-married|               ?|     Own-child| White|
Male| United-States| <=50K|       0.0|        0.053|               0.333|
0.0|               0.0|               0.398|
|         ?|     12th| Never-married|               ?|     Own-child| White|
Male| United-States| <=50K|       0.0|        0.062|               0.467|
0.0|               0.0|               0.398|
|         ?|     10th| Never-married|               ?|     Own-child| White|
Male| United-States| <=50K|       0.0|        0.068|               0.333|
0.0|               0.0|               0.398|
|         ?|     11th| Never-married|               ?|     Own-child| White|
Female| United-States| <=50K|       0.0|         0.07|                 0.4|
0.0|               0.0|               0.173|
|         ?|     10th| Never-married|               ?|     Own-child| White|
Male| United-States| <=50K|       0.0|        0.086|               0.333|
0.0|               0.0|               0.194|
|         ?|     11th| Never-married|               ?| Other-relative| White|
Female| United-States| <=50K|       0.0|         0.09|                 0.4|
0.0|               0.0|               0.245|
|         ?|     10th| Never-married|               ?| Other-relative| White|
Male| United-States| <=50K|       0.0|        0.101|               0.333|
0.0|               0.0|               0.112|
|         ?|     10th| Never-married|               ?|     Own-child| White|
Male| United-States| <=50K|       0.0|        0.135|               0.333|
0.0|               0.0|               0.398|
|         ?|     11th| Never-married|               ?|     Own-child| Black|
Female| United-States| <=50K|       0.0|        0.139|                 0.4|
0.0|               0.0|               0.194|
|         ?|     10th| Never-married|               ?|     Own-child| White|
Female| United-States| <=50K|       0.0|        0.199|               0.333|
0.341|               0.0|               0.316|
```

```
|         ?|     10th| Never-married|              ?|      Own-child| White|
Female| United-States| <=50K|       0.0|        0.422|              0.333|
0.0|               0.0|               0.163|
| Local-gov|      9th| Never-married|   Other-service|      Own-child| Black|
Male| United-States| <=50K|       0.0|        0.013|              0.267|
0.0|               0.0|               0.082|
| Local-gov|     10th| Never-married|   Other-service|      Own-child| White|
Female| United-States| <=50K|       0.0|        0.019|              0.333|
0.0|               0.0|               0.245|
| Local-gov|     11th| Never-married|    Adm-clerical|      Own-child| White|
Female| United-States| <=50K|       0.0|        0.092|                0.4|
0.0|               0.0|               0.112|
| Local-gov|     11th| Never-married| Protective-serv|      Own-child| White|
Male| United-States| <=50K|       0.0|        0.111|                0.4|
0.0|               0.0|               0.296|
| Local-gov|     11th| Never-married|  Prof-specialty|      Own-child| White|
Female|   Puerto-Rico| <=50K|       0.0|        0.159|                0.4|
0.0|               0.0|               0.194|
|   Private|     10th| Never-married|   Other-service|      Own-child| White|
Female| United-States| <=50K|       0.0|        0.009|              0.333|
0.0|               0.0|               0.092|
|   Private|      9th| Never-married|   Other-service|      Own-child| White|
Male| United-States| <=50K|       0.0|        0.011|              0.267|
0.0|               0.0|               0.153|
+----------+---------+--------------+----------------+---------------+------+---
----+--------------+------+----------+-------------+--------------------+-------
------------+------------------+---------------------+
only showing top 20 rows
```

[ ]:
```
trainDF_2.groupBy("income").count().show()
```

```
+------+-----+
|income|count|
+------+-----+
|  >50K| 8363|
| <=50K| 6684|
+------+-----+
```

[ ]:
```
trainDF_2.printSchema()
```

```
root
 |-- workclass: string (nullable = true)
 |-- education: string (nullable = true)
 |-- marital-status: string (nullable = true)
 |-- occupation: string (nullable = true)
 |-- relationship: string (nullable = true)
 |-- race: string (nullable = true)
```

```
 |-- sex: string (nullable = true)
 |-- native-country: string (nullable = true)
 |-- income: string (nullable = true)
 |-- age_Scaled: double (nullable = true)
 |-- fnlwgt_Scaled: double (nullable = true)
 |-- education-num_Scaled: double (nullable = true)
 |-- capital-gain_Scaled: double (nullable = true)
 |-- capital-loss_Scaled: double (nullable = true)
 |-- hours-per-week_Scaled: double (nullable = true)
```

[ ]:
```
trainDF_2.display()
```

[ ]:
```
testDF.groupBy("income").count().show()
```

```
+------+-----+
|income|count|
+------+-----+
|  >50K| 1577|
| <=50K| 4908|
+------+-----+
```

[ ]:
```
major_df2 = testDF.filter(testDF["income"] == " <=50K")
minor_df2 = testDF.filter(testDF["income"] == " >50K")


ratio = int(major_df2.count()/minor_df2.count())

sampled_majority_df2 = testDF.sample(False, 1/ratio)
testDF_2 = sampled_majority_df2.unionAll(minor_df2)
testDF_2.show()
```

```
+---------+------------+--------------+------------------+--------------+-----
-+-------+----------------+------+----------+-------------+--------------------
+------------------+------------------+--------------------+
|workclass|    education|marital-status|        occupation|   relationship|
race|    sex|  native-country|income|age_Scaled|fnlwgt_Scaled|education-
num_Scaled|capital-gain_Scaled|capital-loss_Scaled|hours-per-week_Scaled|
+---------+------------+--------------+------------------+--------------+-----
-+-------+----------------+------+----------+-------------+--------------------
+------------------+------------------+--------------------+
|        ?|         10th| Never-married|                 ?|      Own-child|
White| Female|   United-States| <=50K|       0.0|        0.087|
0.333|               0.0|               0.0|               0.143|
|        ?|         10th| Never-married|                 ?|      Own-child|
White| Female|   United-States| <=50K|       0.0|        0.168|
0.333|               0.0|               0.0|               0.143|
|        ?|         11th| Never-married|                 ?|      Own-child|
```

```
Black| Female| Trinadad&Tobago| <=50K|       0.0|        0.448|
0.4|               0.0|               0.0|                0.398|
|  Private|          9th| Never-married|      Craft-repair|      Own-child|
White| Female|   United-States| <=50K|       0.0|        0.041|
0.267|               0.0|               0.0|                0.153|
|  Private|          9th| Never-married|     Other-service|      Unmarried|
White| Female|   United-States| <=50K|       0.0|        0.059|
0.267|               0.0|               0.0|                0.194|
|  Private|         10th| Never-married|     Other-service|      Own-child|
White|   Male|   United-States| <=50K|       0.0|        0.078|
0.333|               0.0|               0.0|                0.296|
|  Private|         11th| Never-married| Handlers-cleaners|      Own-child|
White|   Male|   United-States| <=50K|       0.0|        0.096|
0.4|               0.0|               0.0|                0.245|
|  Private|         10th| Never-married|     Other-service| Other-relative|
White|   Male|   United-States| <=50K|       0.0|        0.113|
0.333|               0.0|               0.0|                0.143|
|  Private|         11th| Never-married|      Craft-repair|      Own-child|
White|   Male|   United-States| <=50K|       0.0|        0.126|
0.4|               0.0|               0.0|                0.163|
|  Private|         11th| Never-married|             Sales|      Own-child|
White| Female|   United-States| <=50K|       0.0|        0.128|
0.4|               0.0|               0.0|                0.143|
|  Private|         11th| Never-married|             Sales|      Own-child|
White| Female|   United-States| <=50K|       0.0|         0.14|
0.4|               0.0|               0.0|                0.071|
|  Private|         10th| Never-married|     Other-service|      Own-child|
White| Female|   United-States| <=50K|       0.0|        0.142|
0.333|               0.0|             0.368|                0.112|
|  Private|      HS-grad| Never-married|   Farming-fishing|      Own-child|
Black|   Male|   United-States| <=50K|       0.0|        0.153|
0.533|               0.0|               0.0|                0.398|
|  Private|         10th| Never-married| Handlers-cleaners|      Own-child|
White| Female|   United-States| <=50K|       0.0|        0.171|
0.333|               0.0|               0.0|                0.224|
|  Private|         11th| Never-married|      Adm-clerical|      Own-child|
White| Female|   United-States| <=50K|       0.0|        0.181|
0.4|               0.0|               0.0|                0.071|
|  Private|         11th| Never-married|             Sales|      Own-child|
White|   Male|   United-States| <=50K|       0.0|        0.183|
0.4|               0.0|               0.0|                0.296|
|  Private|         11th| Never-married|    Prof-specialty|      Own-child|
White|   Male|   United-States| <=50K|       0.0|        0.189|
0.4|               0.0|               0.0|                0.398|
|  Private|         10th| Never-married|      Adm-clerical|      Own-child|
White| Female|   United-States| <=50K|       0.0|        0.196|
0.333|               0.0|               0.0|                0.194|
|        ?|         11th| Never-married|                 ?|      Own-child|
```

```
White|   Male|   United-States| <=50K|     0.014|        0.188|
0.4|                0.0|                0.0|                0.245|
|        ?| Some-college| Never-married|                 ?|      Own-child|
White|   Male|   United-States| <=50K|     0.014|        0.197|
0.6|                0.0|                0.0|                0.398|
+---------+-------------+--------------+------------------+---------------+-----
-+-------+----------------+------+----------+-------------+--------------------
+-------------------+-------------------+--------------------+
only showing top 20 rows
```

[ ]:
```
testDF_2.groupBy("income").count().show()
```

```
+------+-----+
|income|count|
+------+-----+
|  >50K| 2114|
| <=50K| 1639|
+------+-----+
```

[ ]:
```
testDF_2.display()
```

```
Databricks visualization. Run in Databricks to view.

Databricks visualization. Run in Databricks to view.

Databricks visualization. Run in Databricks to view.
```

[ ]:
```
categoricalCols = ["workclass", "education", "marital-status", "occupation",␣
 ↪"relationship", "race", "sex", "native-country"]

#use transformer to convert string columns to label indexes, and encoder to map␣
 ↪binary indices to column of binary vectors
stringIndexer = StringIndexer(inputCols=categoricalCols, outputCols=[x + "Index"␣
 ↪for x in categoricalCols], handleInvalid= "keep")
encoder = OneHotEncoder(inputCols=stringIndexer.getOutputCols(), outputCols=[x +␣
 ↪"OHE" for x in categoricalCols])

#convert label (income) from string to numeric value
labelToIndex = StringIndexer(inputCol="income", outputCol="label")


numericCols = ["age_Scaled", "fnlwgt_Scaled", "education-num_Scaled",␣
 ↪"capital-gain_Scaled", "capital-loss_Scaled", "hours-per-week_Scaled"]
assemblerInputs = [c + "OHE" for c in categoricalCols] + numericCols
assembler = VectorAssembler(inputCols=assemblerInputs, outputCol="features")
```

```
stages = [stringIndexer, encoder, labelToIndex, assembler]
partialPipeline = Pipeline().setStages(stages)
pipelineModel = partialPipeline.fit(trainDF_2)
trainDF_fitted = pipelineModel.transform(trainDF_2)
```

```
Downloading artifacts:   0%|          | 0/25 [00:00<?, ?it/s]
Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]
```

[ ]:
```
testDF_fitted = pipelineModel.transform(testDF_2)
```

[ ]:
```
lrModel = LogisticRegression(family="binomial").fit(trainDF_fitted)
```

```
Downloading artifacts:   0%|          | 0/9 [00:00<?, ?it/s]
Uploading artifacts:   0%|          | 0/4 [00:00<?, ?it/s]
```

[ ]:
```
train_preds = lrModel.transform(trainDF_fitted)
test_preds = lrModel.transform(testDF_fitted)
```

[ ]:
```
display(train_preds)
```

[ ]:
```
display(test_preds)
```

[ ]:
```
display(lrModel, train_preds.drop("prediction", "rawPrediction", "probability"),␣
↪"ROC")
```

[ ]:
```
display(lrModel, test_preds.drop("prediction", "rawPrediction", "probability"),␣
↪"ROC")
```

[ ]:
```
bcEvaluator = BinaryClassificationEvaluator(metricName="areaUnderROC")
print(f"Area under ROC curve: {bcEvaluator.evaluate(train_preds)}")

mcEvaluator = MulticlassClassificationEvaluator(metricName="accuracy")
print(f"Accuracy: {mcEvaluator.evaluate(train_preds)}")

mcEvaluatorf1 = MulticlassClassificationEvaluator(metricName="f1")
print(f"f1: {mcEvaluatorf1.evaluate(train_preds)}")
```

```
Area under ROC curve: 0.9099875269891968
Accuracy: 0.8343191333820695
f1: 0.8333702522705072
```

[ ]:
```
bcEvaluator = BinaryClassificationEvaluator(metricName="areaUnderROC")
print(f"Area under ROC curve: {bcEvaluator.evaluate(test_preds)}")

mcEvaluator = MulticlassClassificationEvaluator(metricName="accuracy")
print(f"Accuracy: {mcEvaluator.evaluate(test_preds)}")

mcEvaluatorf1 = MulticlassClassificationEvaluator(metricName="f1")
```

```
print(f"f1: {mcEvaluatorf1.evaluate(test_preds)}")
```

```
Area under ROC curve: 0.9034123017300023
Accuracy: 0.8326671995736744
f1: 0.8315597473405894
```