Financial Fraud Detection Using Pyspark

September 5, 2022

1 FINANCIAL FRAUD DETCTION USING PYSPARK

[1]:
```
#load_ext nb_black
from pyspark.sql import SparkSession
#session all related to df not rdd
import pyspark.sql.functions as F
import pyspark.sql.types as T

spark = SparkSession.builder.getOrCreate()
```

[2]:
```
spark
```

[2]:
```
<pyspark.sql.session.SparkSession at 0x2a978289130>
```

[7]:
```
df = spark.read.csv("financial.csv", inferSchema=True, header=True)
```

[8]:
```
df.printSchema()
```

```
root
 |-- step: integer (nullable = true)
 |-- type: string (nullable = true)
 |-- amount: double (nullable = true)
 |-- nameOrig: string (nullable = true)
 |-- oldbalanceOrg: double (nullable = true)
 |-- newbalanceOrig: double (nullable = true)
 |-- nameDest: string (nullable = true)
 |-- oldbalanceDest: double (nullable = true)
 |-- newbalanceDest: double (nullable = true)
 |-- isFraud: integer (nullable = true)
 |-- isFlaggedFraud: integer (nullable = true)
```

[9]:
```
df.show(2)
```

```
+----+-------+-------+----------+-------------+--------------+----------+-----
---------+--------------+-------+--------------+
|step|   type| amount|   nameOrig|oldbalanceOrg|newbalanceOrig|
nameDest|oldbalanceDest|newbalanceDest|isFraud|isFlaggedFraud|
+----+-------+-------+----------+-------------+--------------+----------+-----
```

```
---------+-------------+-------+-------------+
|   1|PAYMENT|9839.64|C1231006815|     170136.0|     160296.36|M1979787155|
0.0|           0.0|      0|            0|
|   1|PAYMENT|1864.28|C1666544295|      21249.0|      19384.72|M2044282225|
0.0|           0.0|      0|            0|
+----+-------+-------+----------+-------------+-------------+----------+-----
---------+-------------+-------+-------------+
only showing top 2 rows
```

[10]:
```
df = df.select("type", "amount", "oldbalanceOrg", "newbalanceOrig", "isFraud")
```

[11]:
```
df.show(2)
```

```
+-------+-------+-------------+-------------+-------+
|   type| amount|oldbalanceOrg|newbalanceOrig|isFraud|
+-------+-------+-------------+-------------+-------+
|PAYMENT|9839.64|     170136.0|     160296.36|      0|
|PAYMENT|1864.28|      21249.0|      19384.72|      0|
+-------+-------+-------------+-------------+-------+
only showing top 2 rows
```

[12]:
```
df.printSchema()
```

```
root
 |-- type: string (nullable = true)
 |-- amount: double (nullable = true)
 |-- oldbalanceOrg: double (nullable = true)
 |-- newbalanceOrig: double (nullable = true)
 |-- isFraud: integer (nullable = true)
```

[13]:
```
df.count() , len(df.columns)
```

[13]:
```
(6362620, 5)
```

[14]:
```
df.select('amount','oldbalanceOrg','newbalanceOrig','isFraud').describe().show()
```

```
+-------+-----------------+----------------+----------------+---------------
----+
|summary|           amount|   oldbalanceOrg|   newbalanceOrig|
isFraud|
+-------+-----------------+----------------+----------------+---------------
----+
|  count|          6362620|         6362620|         6362620|
6362620|
|   mean|179861.90354913412|833883.1040744719|855113.6685785714|0.00129082044818
0152|
```

```
| stddev|
603858.2314629498|2888242.673037545|2924048.502954253|0.035904796801604424|
|    min|               0.0|              0.0|              0.0|
0|
|    max|     9.244551664E7|    5.958504037E7|    4.958504037E7|
1|
+-------+------------------+-----------------+-----------------+----------------
----+
```

[15]:
```
# null values in each column
data_agg = df.agg(*[F.count(F.when(F.isnull(c), c)).alias(c) for c in df.
 ↪columns])
data_agg.show()
```

```
+----+------+-------------+--------------+-------+
|type|amount|oldbalanceOrg|newbalanceOrig|isFraud|
+----+------+-------------+--------------+-------+
|   0|     0|            0|             0|      0|
+----+------+-------------+--------------+-------+
```

[16]:
```
# value counts of Type column
df.groupBy('type').count().show()
```

```
+--------+-------+
|    type|  count|
+--------+-------+
|TRANSFER| 532909|
| CASH_IN|1399284|
|CASH_OUT|2237500|
| PAYMENT|2151495|
|   DEBIT|  41432|
+--------+-------+
```

[17]:
```
train, test = df.randomSplit([0.7, 0.3], seed=7)
```

[18]:
```
print(f"Train set length: {train.count()} records")
print(f"Test set length: {test.count()} records")
```

```
Train set length: 4451877 records
Test set length: 1910743 records
```

[19]:
```
train.show(2)
```

```
+-------+------+-------------+--------------+-------+
|   type|amount|oldbalanceOrg|newbalanceOrig|isFraud|
+-------+------+-------------+--------------+-------+
|CASH_IN|  1.42|   1270713.41|    1270714.83|      0|
```

```
|CASH_IN|  4.35|   4136277.22|    4136281.57|      0|
+-------+------+-------------+--------------+-------+
only showing top 2 rows
```

[20]:
```
train.dtypes
```

[20]:
```
[('type', 'string'),
 ('amount', 'double'),
 ('oldbalanceOrg', 'double'),
 ('newbalanceOrig', 'double'),
 ('isFraud', 'int')]
```

[21]:
```
train.show(2)
```

```
+-------+------+-------------+--------------+-------+
|   type|amount|oldbalanceOrg|newbalanceOrig|isFraud|
+-------+------+-------------+--------------+-------+
|CASH_IN|  1.42|   1270713.41|    1270714.83|      0|
|CASH_IN|  4.35|   4136277.22|    4136281.57|      0|
+-------+------+-------------+--------------+-------+
only showing top 2 rows
```

[22]:
```
catCols = [x for (x, dataType) in train.dtypes if dataType == "string"]
numCols = [ x for (x, dataType) in train.dtypes if (dataType == "double") ]
#numCols = [ x for (x, dataType) in train.dtypes if ((dataType == "double") &
 ↪(x != "isFraud")) ]
#skip the "isFraud" but
```

[23]:
```
print(numCols)
print(catCols)
```

```
['amount', 'oldbalanceOrg', 'newbalanceOrig']
['type']
```

[24]:
```
train.agg(F.countDistinct("type")).show()
```

```
+-----------+
|count(type)|
+-----------+
|          5|
+-----------+
```

[25]:
```
train.groupBy("type").count().show()
```

```
+--------+-------+
|    type|  count|
+--------+-------+
```

```
|TRANSFER| 372791|
| CASH_IN| 979384|
|CASH_OUT|1565928|
| PAYMENT|1504894|
|   DEBIT|  28880|
+--------+-------+
```

[26]:
```
from pyspark.ml.feature import (
    OneHotEncoder,
    StringIndexer,
)
```

[27]:
```
df = df.select("type","isFraud")
```

[28]:
```
#catCols are the cols with string
string_indexer = [
    StringIndexer(inputCol=x, outputCol=x + "_StringIndexer",␣
 ↪handleInvalid="skip")
    for x in catCols
]
```

[29]:
```
string_indexe=string_indexer[0].fit(df).transform(df)
string_indexe.show()
```

```
+--------+-------+------------------+
|    type|isFraud|type_StringIndexer|
+--------+-------+------------------+
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
|TRANSFER|      1|               3.0|
|CASH_OUT|      1|               0.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
|   DEBIT|      0|               4.0|
|   DEBIT|      0|               4.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
|CASH_OUT|      0|               0.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
| PAYMENT|      0|               1.0|
|TRANSFER|      0|               3.0|
```

```
+--------+------+-----------------+
only showing top 20 rows
```

[30]:
```
one_hot_encoder = [
    OneHotEncoder(
        inputCols=[f"{x}_StringIndexer" for x in catCols],
        outputCols=[f"{x}_OneHotEncoder" for x in catCols],
    )
]
```

[31]:
```
from pyspark.ml.feature import VectorAssembler
```

[32]:
```
assemblerInput = [x for x in numCols]
assemblerInput += [f"{x}_OneHotEncoder" for x in catCols]
assemblerInput
```

[32]:
```
['amount', 'oldbalanceOrg', 'newbalanceOrig', 'type_OneHotEncoder']
```

[33]:
```
vector_assembler = VectorAssembler(
    inputCols=assemblerInput, outputCol="VectorAssembler_features"
)
```

[34]:
```
stages = []
stages += string_indexer
stages += one_hot_encoder
stages += [vector_assembler]
```

[35]:
```
stages
```

[35]:
```
[StringIndexer_27a4f739ca00,
 OneHotEncoder_312d975a7740,
 VectorAssembler_850465fa6f68]
```

[36]:
```
#%%time
from pyspark.ml import Pipeline

pipeline = Pipeline().setStages(stages)
model = pipeline.fit(train)

pp_df = model.transform(train)
```

[37]:
```
pp_df.select(
    "type", "amount", "oldbalanceOrg", "newbalanceOrig",␣
 ↪"VectorAssembler_features",
).show(truncate=False)
```

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

```
|type   |amount|oldbalanceOrg|newbalanceOrig|VectorAssembler_features
|
+-------+------+-------------+--------------+-----------------------------------
----------------+
|CASH_IN|1.42  |1270713.41   |1270714.83
|[1.42,1270713.41,1270714.83,0.0,0.0,1.0,0.0]       |
|CASH_IN|4.35  |4136277.22   |4136281.57
|[4.35,4136277.22,4136281.57,0.0,0.0,1.0,0.0]       |
|CASH_IN|4.71  |50198.0      |50202.71
|[4.71,50198.0,50202.71,0.0,0.0,1.0,0.0]            |
|CASH_IN|5.19  |18104.0      |18109.19
|[5.19,18104.0,18109.19,0.0,0.0,1.0,0.0]            |
|CASH_IN|5.66  |5061561.06   |5061566.72
|[5.66,5061561.06,5061566.72,0.0,0.0,1.0,0.0]       |
|CASH_IN|6.5   |1696433.45   |1696439.95
|[6.5,1696433.45,1696439.95,0.0,0.0,1.0,0.0]        |
|CASH_IN|8.29  |20392.0      |20400.29
|[8.29,20392.0,20400.29,0.0,0.0,1.0,0.0]            |
|CASH_IN|9.02  |2416260.59   |2416269.61
|[9.02,2416260.59,2416269.61,0.0,0.0,1.0,0.0]       |
|CASH_IN|12.18 |299322.0     |299334.18
|[12.18,299322.0,299334.18,0.0,0.0,1.0,0.0]         |
|CASH_IN|13.2  |106204.0     |106217.2
|[13.2,106204.0,106217.2,0.0,0.0,1.0,0.0]           |
|CASH_IN|14.36 |613030.46    |613044.82
|[14.36,613030.46,613044.82,0.0,0.0,1.0,0.0]        |
|CASH_IN|14.4  |1.143460813E7|1.143462253E7
|[14.4,1.143460813E7,1.143462253E7,0.0,0.0,1.0,0.0] |
|CASH_IN|16.89 |0.0          |16.89         |(7,[0,2,5],[16.89,16.89,1.0])
|
|CASH_IN|17.33 |8964056.72   |8964074.05
|[17.33,8964056.72,8964074.05,0.0,0.0,1.0,0.0]      |
|CASH_IN|17.53 |1111294.85   |1111312.37
|[17.53,1111294.85,1111312.37,0.0,0.0,1.0,0.0]      |
|CASH_IN|21.15 |2729078.29   |2729099.44
|[21.15,2729078.29,2729099.44,0.0,0.0,1.0,0.0]      |
|CASH_IN|21.57 |104362.0     |104383.57
|[21.57,104362.0,104383.57,0.0,0.0,1.0,0.0]         |
|CASH_IN|22.31 |0.0          |22.31         |(7,[0,2,5],[22.31,22.31,1.0])
|
|CASH_IN|22.67 |405940.0     |405962.67
|[22.67,405940.0,405962.67,0.0,0.0,1.0,0.0]         |
|CASH_IN|23.36 |2.442828608E7|2.442830944E7
|[23.36,2.442828608E7,2.442830944E7,0.0,0.0,1.0,0.0]|
+-------+------+-------------+--------------+-----------------------------------
----------------+
only showing top 20 rows
```

[38]:
```
pp_df.show()
```

```
+-------+------+-------------+--------------+-------+-----------------+--------
---------+------------------------+
|   type|amount|oldbalanceOrg|newbalanceOrig|isFraud|type_StringIndexer|type_One
HotEncoder|VectorAssembler_features|
+-------+------+-------------+--------------+-------+-----------------+--------
---------+------------------------+
|CASH_IN|  1.42|   1270713.41|    1270714.83|      0|              2.0|
(4,[2],[1.0])|    [1.42,1270713.41,…|
|CASH_IN|  4.35|   4136277.22|    4136281.57|      0|              2.0|
(4,[2],[1.0])|    [4.35,4136277.22,…|
|CASH_IN|  4.71|      50198.0|      50202.71|      0|              2.0|
(4,[2],[1.0])|    [4.71,50198.0,502…|
|CASH_IN|  5.19|      18104.0|      18109.19|      0|              2.0|
(4,[2],[1.0])|    [5.19,18104.0,181…|
|CASH_IN|  5.66|   5061561.06|    5061566.72|      0|              2.0|
(4,[2],[1.0])|    [5.66,5061561.06,…|
|CASH_IN|   6.5|   1696433.45|    1696439.95|      0|              2.0|
(4,[2],[1.0])|    [6.5,1696433.45,1…|
|CASH_IN|  8.29|      20392.0|      20400.29|      0|              2.0|
(4,[2],[1.0])|    [8.29,20392.0,204…|
|CASH_IN|  9.02|   2416260.59|    2416269.61|      0|              2.0|
(4,[2],[1.0])|    [9.02,2416260.59,…|
|CASH_IN| 12.18|     299322.0|     299334.18|      0|              2.0|
(4,[2],[1.0])|    [12.18,299322.0,2…|
|CASH_IN|  13.2|     106204.0|      106217.2|      0|              2.0|
(4,[2],[1.0])|    [13.2,106204.0,10…|
|CASH_IN| 14.36|    613030.46|     613044.82|      0|              2.0|
(4,[2],[1.0])|    [14.36,613030.46,…|
|CASH_IN|  14.4|1.143460813E7| 1.143462253E7|      0|              2.0|
(4,[2],[1.0])|    [14.4,1.143460813…|
|CASH_IN| 16.89|          0.0|         16.89|      0|              2.0|
(4,[2],[1.0])|    (7,[0,2,5],[16.89…|
|CASH_IN| 17.33|   8964056.72|    8964074.05|      0|              2.0|
(4,[2],[1.0])|    [17.33,8964056.72…|
|CASH_IN| 17.53|   1111294.85|    1111312.37|      0|              2.0|
(4,[2],[1.0])|    [17.53,1111294.85…|
|CASH_IN| 21.15|   2729078.29|    2729099.44|      0|              2.0|
(4,[2],[1.0])|    [21.15,2729078.29…|
|CASH_IN| 21.57|     104362.0|     104383.57|      0|              2.0|
(4,[2],[1.0])|    [21.57,104362.0,1…|
|CASH_IN| 22.31|          0.0|         22.31|      0|              2.0|
(4,[2],[1.0])|    (7,[0,2,5],[22.31…|
|CASH_IN| 22.67|     405940.0|     405962.67|      0|              2.0|
(4,[2],[1.0])|    [22.67,405940.0,4…|
|CASH_IN| 23.36|2.442828608E7| 2.442830944E7|      0|              2.0|
(4,[2],[1.0])|    [23.36,2.44282860…|
```

```
+-------+------+------------+-------------+-------+-----------------+--------
----------+-------------------------+
only showing top 20 rows
```

[39]:
```
test.count()
```

[39]: 1910743

[40]:
```
df_test=test.where(test.isFraud == 1)
```

[41]:
```
df_test.show()
```

```
+--------+--------+-------------+-------------+-------+
|    type|  amount|oldbalanceOrg|newbalanceOrig|isFraud|
+--------+--------+-------------+-------------+-------+
|CASH_OUT| 1996.17|      1996.17|          0.0|      1|
|CASH_OUT|  2007.0|       2007.0|          0.0|      1|
|CASH_OUT|  2100.0|       2100.0|          0.0|      1|
|CASH_OUT| 4094.07|      4094.07|          0.0|      1|
|CASH_OUT| 4120.14|      4120.14|          0.0|      1|
|CASH_OUT| 4289.18|      4289.18|          0.0|      1|
|CASH_OUT| 4530.71|      4530.71|          0.0|      1|
|CASH_OUT| 6783.47|      6783.47|          0.0|      1|
|CASH_OUT| 6844.73|      6844.73|          0.0|      1|
|CASH_OUT| 7887.75|      7887.75|          0.0|      1|
|CASH_OUT| 7927.06|      7927.06|          0.0|      1|
|CASH_OUT| 8380.79|      8380.79|          0.0|      1|
|CASH_OUT|  8677.0|       8677.0|          0.0|      1|
|CASH_OUT| 9917.27|      9917.27|          0.0|      1|
|CASH_OUT| 10565.0|      10565.0|          0.0|      1|
|CASH_OUT| 12461.0|      12461.0|          0.0|      1|
|CASH_OUT|14016.65|     14016.65|          0.0|      1|
|CASH_OUT| 17246.0|      17246.0|          0.0|      1|
|CASH_OUT|18126.95|     18126.95|          0.0|      1|
|CASH_OUT|18627.02|     18627.02|          0.0|      1|
+--------+--------+-------------+-------------+-------+
only showing top 20 rows
```

[42]:
```
from pyspark.ml.classification import LogisticRegression
```

[43]:
```
data = pp_df.select(
    F.col("VectorAssembler_features").alias("features"),
    F.col("isFraud").alias("label"),
)
```

[44]:
```
data.show(5, truncate=False)
```

```
+-----------------------------------------+-----+
|features                                 |label|
+-----------------------------------------+-----+
|[1.42,1270713.41,1270714.83,0.0,0.0,1.0,0.0]|0    |
|[4.35,4136277.22,4136281.57,0.0,0.0,1.0,0.0]|0    |
|[4.71,50198.0,50202.71,0.0,0.0,1.0,0.0]     |0    |
|[5.19,18104.0,18109.19,0.0,0.0,1.0,0.0]     |0    |
|[5.66,5061561.06,5061566.72,0.0,0.0,1.0,0.0]|0    |
+-----------------------------------------+-----+
only showing top 5 rows
```

[45]:
```
%%time
model = LogisticRegression().fit(data)
data=model.transform(data)
```

```
Wall time: 1min 54s
```

[46]:
```
data.show()
```

```
+--------------------+-----+--------------------+-----------+----------+
|            features|label|       rawPrediction|probability|prediction|
+--------------------+-----+--------------------+-----------+----------+
|[1.42,1270713.41,…|    0|[197.561087692828…|  [1.0,0.0]|       0.0|
|[4.35,4136277.22,…|    0|[203.627061733643…|  [1.0,0.0]|       0.0|
|[4.71,50198.0,502…|    0|[194.977674963424…|  [1.0,0.0]|       0.0|
|[5.19,18104.0,181…|    0|[194.909763566974…|  [1.0,0.0]|       0.0|
|[5.66,5061561.06,…|    0|[205.585769589595…|  [1.0,0.0]|       0.0|
|[6.5,1696433.45,1…|    0|[198.462517102936…|  [1.0,0.0]|       0.0|
|[8.29,20392.0,204…|    0|[194.914769091311…|  [1.0,0.0]|       0.0|
|[9.02,2416260.59,…|    0|[199.986377907443…|  [1.0,0.0]|       0.0|
|[12.18,299322.0,2…|    0|[195.505411293430…|  [1.0,0.0]|       0.0|
|[13.2,106204.0,10…|    0|[195.096672834587…|  [1.0,0.0]|       0.0|
|[14.36,613030.46,…|    0|[196.169582960010…|  [1.0,0.0]|       0.0|
|[14.4,1.143460813…|    0|[219.076682455198…|  [1.0,0.0]|       0.0|
|(7,[0,2,5],[16.89…|    0|[194.872053563307…|  [1.0,0.0]|       0.0|
|[17.33,8964056.72…|    0|[213.847176778020…|  [1.0,0.0]|       0.0|
|[17.53,1111294.85…|    0|[197.224473828982…|  [1.0,0.0]|       0.0|
|[21.15,2729078.29…|    0|[200.649184912619…|  [1.0,0.0]|       0.0|
|[21.57,104362.0,1…|    0|[195.093211874512…|  [1.0,0.0]|       0.0|
|(7,[0,2,5],[22.31…|    0|[194.872337308676…|  [1.0,0.0]|       0.0|
|[22.67,405940.0,4…|    0|[195.731649264110…|  [1.0,0.0]|       0.0|
|[23.36,2.44282860…|    0|[246.582147143619…|  [1.0,0.0]|       0.0|
+--------------------+-----+--------------------+-----------+----------+
only showing top 20 rows
```

[47]:
```
model = pipeline.fit(df_test)

pp_df_test = model.transform(df_test)
```

[48]:
```
data_test = pp_df_test.select(
    F.col("VectorAssembler_features").alias("features"),
    F.col("isFraud").alias("label"),
)
```

[49]:
```
data_test.show(5, truncate=False)
```

```
+-------------------------+-----+
|features                 |label|
+-------------------------+-----+
|[1996.17,1996.17,0.0,0.0]|1    |
|[2007.0,2007.0,0.0,0.0]  |1    |
|[2100.0,2100.0,0.0,0.0]  |1    |
|[4094.07,4094.07,0.0,0.0]|1    |
|[4120.14,4120.14,0.0,0.0]|1    |
+-------------------------+-----+
only showing top 5 rows
```

[50]:
```
model = LogisticRegression().fit(data_test)
data=model.transform(data_test)
data.show()
```

```
+--------------------+-----+--------------------+----------+----------+
|            features|label|       rawPrediction|probability|prediction|
+--------------------+-----+--------------------+----------+----------+
|[1996.17,1996.17,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[2007.0,2007.0,0…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[2100.0,2100.0,0…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[4094.07,4094.07,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[4120.14,4120.14,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[4289.18,4289.18,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[4530.71,4530.71,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[6783.47,6783.47,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[6844.73,6844.73,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[7887.75,7887.75,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[7927.06,7927.06,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[8380.79,8380.79,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[8677.0,8677.0,0…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[9917.27,9917.27,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[10565.0,10565.0,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[12461.0,12461.0,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[14016.65,14016.6…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[17246.0,17246.0,…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
```

```
|[18126.95,18126.9…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
|[18627.02,18627.0…|    1|[-Infinity,Infinity]|  [0.0,1.0]|       1.0|
+--------------------+-----+--------------------+-----------+----------+
only showing top 20 rows
```

[51]:
```
df.limit
```

[51]:
```
<bound method DataFrame.limit of DataFrame[type: string, isFraud: int]>
```

[52]:
```
model.summary.areaUnderROC
```

[52]:
```
1.0
```

[53]:
```
model.summary.pr.show()
```

```
+------+---------+
|recall|precision|
+------+---------+
|   0.0|      1.0|
|   1.0|      1.0|
+------+---------+
```

[ ]:
```

```