BIG DATA ANALYTICS LAB

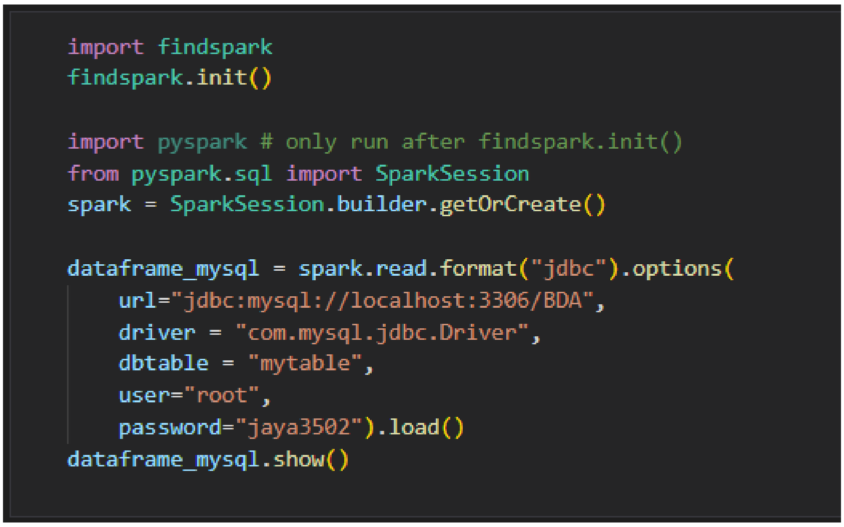
Name : Jayasuriya E,Sanjay Krishna R
Roll No.: 1934012 , 1934036

Dataset:
Goal: The Aim is to Classify whether The Loan Will Get Sanctioned or not.
Link: https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset

Features:

- Loan_Id
- Gender
- Married
- Dependents
- Education
- Self-Employed
- ApplicationIncome
- CoapplicantIncome
- LoanAmount
- Loan_Amount_Term
- Credit_History
- Property_Area
- Loan_Status

Connecting to database:



```
import findspark
findspark.init()

import pyspark # only run after findspark.init()
from pyspark.sql import SparkSession
spark = SparkSession.builder.getOrCreate()

dataframe_mysql = spark.read.format("jdbc").options(
    url="jdbc:mysql://localhost:3306/BDA",
    driver = "com.mysql.jdbc.Driver",
    dbtable = "mytable",
    user="root",
    password="jaya3502").load()
dataframe_mysql.show()
```

The MySQL database is connected.

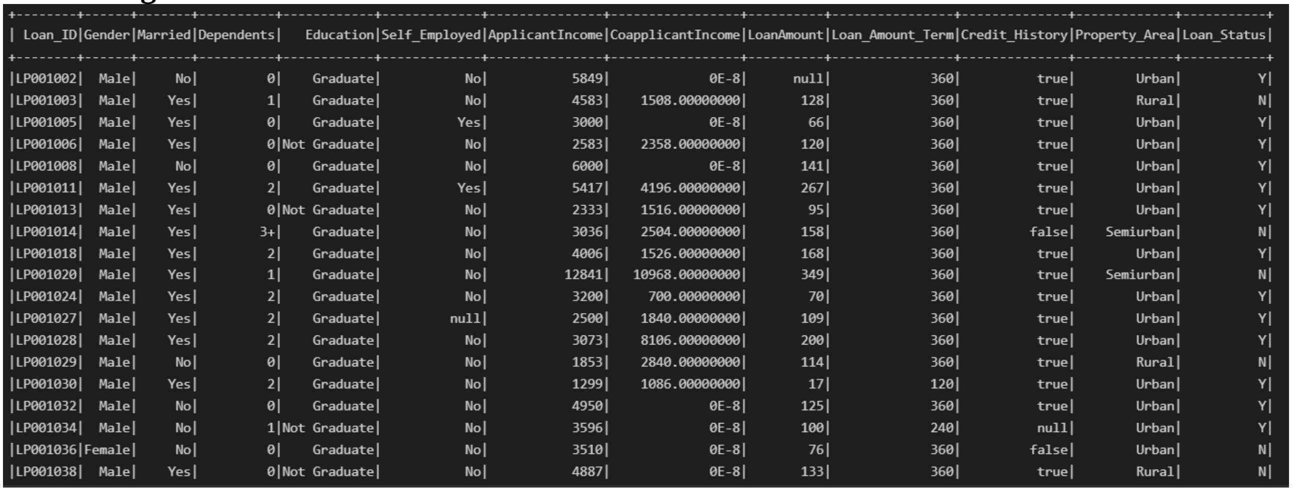
Importing Libraries:

The Libraries used are,

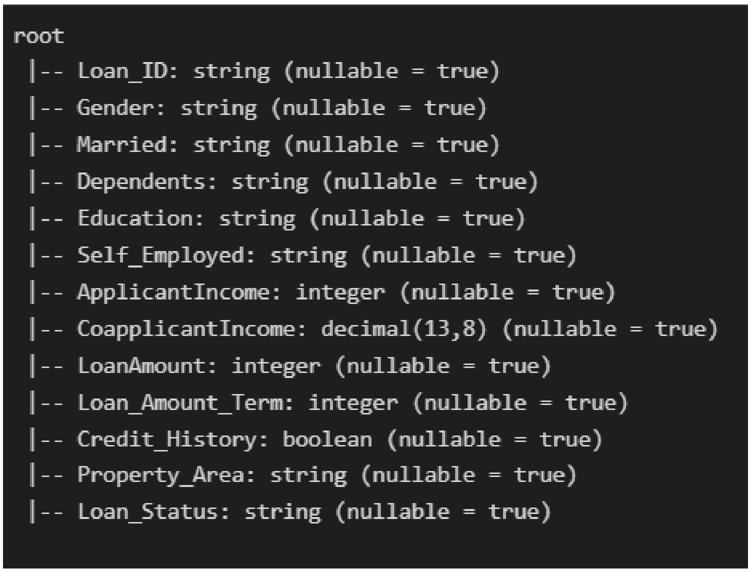
- pyspark
- findspark
- pandas
- numpy

- matplotlib
- sklearn
- seaborn

Visualizing database and Schema of database:

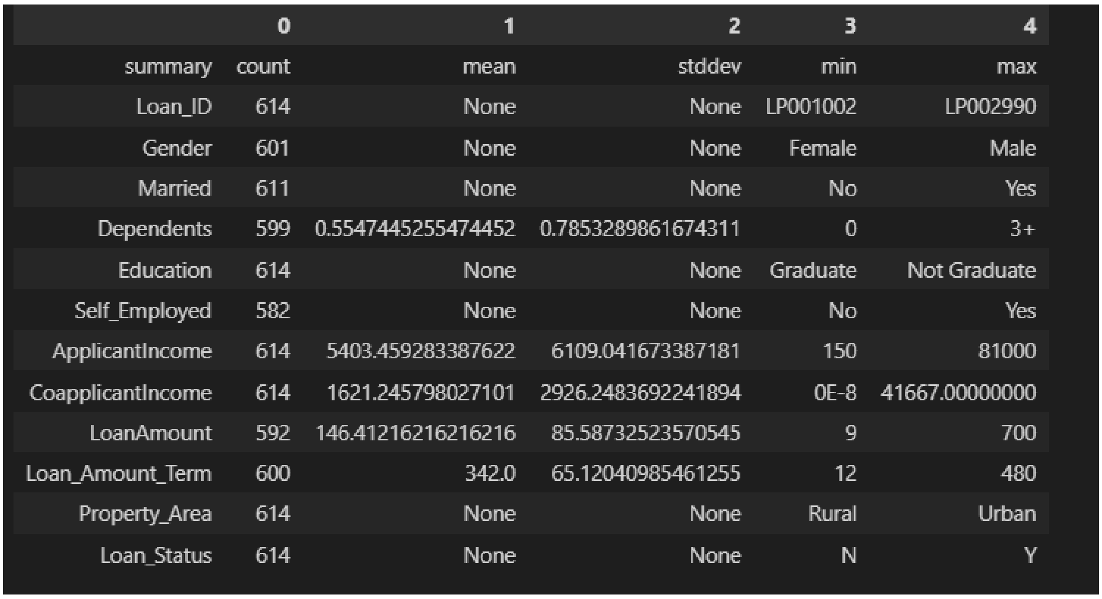


```
+--------+------+-------+----------+------------+-------------+---------------+-----------------+----------+----------------+--------------+-------------+-----------+
| Loan_ID|Gender|Married|Dependents|   Education|Self_Employed|ApplicantIncome|CoapplicantIncome|LoanAmount|Loan_Amount_Term|Credit_History|Property_Area|Loan_Status|
+--------+------+-------+----------+------------+-------------+---------------+-----------------+----------+----------------+--------------+-------------+-----------+
|LP001002|  Male|     No|         0|    Graduate|           No|           5849|             0E-8|      null|             360|          true|        Urban|          Y|
|LP001003|  Male|    Yes|         1|    Graduate|           No|           4583|    1508.00000000|       128|             360|          true|        Rural|          N|
|LP001005|  Male|    Yes|         0|    Graduate|          Yes|           3000|             0E-8|        66|             360|          true|        Urban|          Y|
|LP001006|  Male|    Yes|         0|Not Graduate|           No|           2583|    2358.00000000|       120|             360|          true|        Urban|          Y|
|LP001008|  Male|     No|         0|    Graduate|           No|           6000|             0E-8|       141|             360|          true|        Urban|          Y|
|LP001011|  Male|    Yes|         2|    Graduate|          Yes|           5417|    4196.00000000|       267|             360|          true|        Urban|          Y|
|LP001013|  Male|    Yes|         0|Not Graduate|           No|           2333|    1516.00000000|        95|             360|          true|        Urban|          Y|
|LP001014|  Male|    Yes|        3+|    Graduate|           No|           3036|    2504.00000000|       158|             360|         false|    Semiurban|          N|
|LP001018|  Male|    Yes|         2|    Graduate|           No|           4006|    1526.00000000|       168|             360|          true|        Urban|          Y|
|LP001020|  Male|    Yes|         1|    Graduate|           No|          12841|   10968.00000000|       349|             360|          true|    Semiurban|          N|
|LP001024|  Male|    Yes|         2|    Graduate|           No|           3200|     700.00000000|        70|             360|          true|        Urban|          Y|
|LP001027|  Male|    Yes|         2|    Graduate|         null|           2500|    1840.00000000|       109|             360|          true|        Urban|          Y|
|LP001028|  Male|    Yes|         2|    Graduate|           No|           3073|    8106.00000000|       200|             360|          true|        Urban|          Y|
|LP001029|  Male|     No|         0|    Graduate|           No|           1853|    2840.00000000|       114|             360|          true|        Rural|          N|
|LP001030|  Male|    Yes|         2|    Graduate|           No|           1299|    1086.00000000|        17|             120|          true|        Urban|          Y|
|LP001032|  Male|     No|         0|    Graduate|           No|           4950|             0E-8|       125|             360|          true|        Urban|          Y|
|LP001034|  Male|     No|         1|Not Graduate|           No|           3596|             0E-8|       100|             240|          null|        Urban|          Y|
|LP001036|Female|     No|         0|    Graduate|           No|           3510|             0E-8|        76|             360|         false|        Urban|          N|
|LP001038|  Male|    Yes|         0|Not Graduate|           No|           4887|             0E-8|       133|             360|          true|        Rural|          N|
```



```
root
 |-- Loan_ID: string (nullable = true)
 |-- Gender: string (nullable = true)
 |-- Married: string (nullable = true)
 |-- Dependents: string (nullable = true)
 |-- Education: string (nullable = true)
 |-- Self_Employed: string (nullable = true)
 |-- ApplicantIncome: integer (nullable = true)
 |-- CoapplicantIncome: decimal(13,8) (nullable = true)
 |-- LoanAmount: integer (nullable = true)
 |-- Loan_Amount_Term: integer (nullable = true)
 |-- Credit_History: boolean (nullable = true)
 |-- Property_Area: string (nullable = true)
 |-- Loan_Status: string (nullable = true)
```

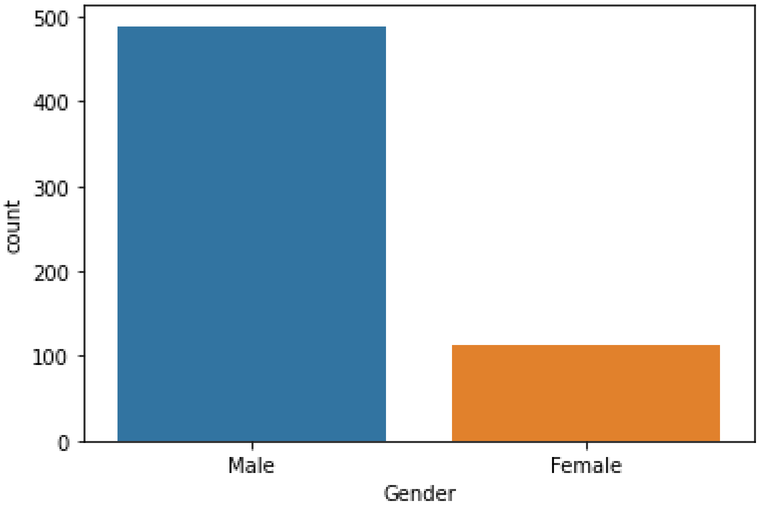
Describe the datatbase:

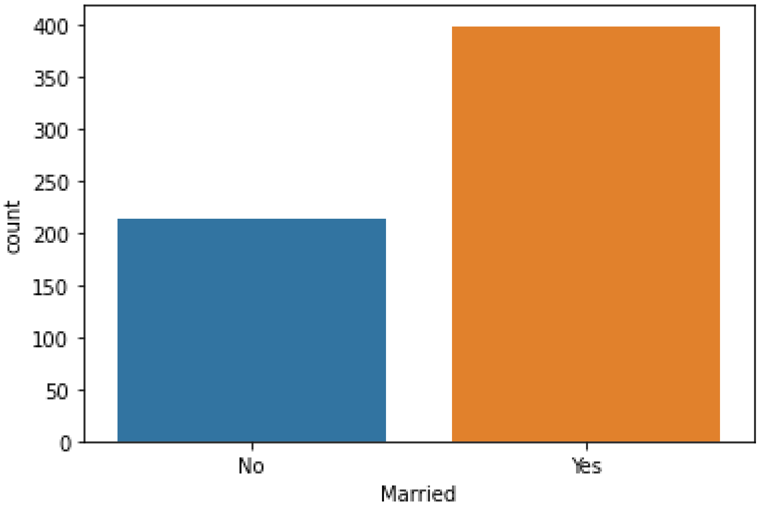


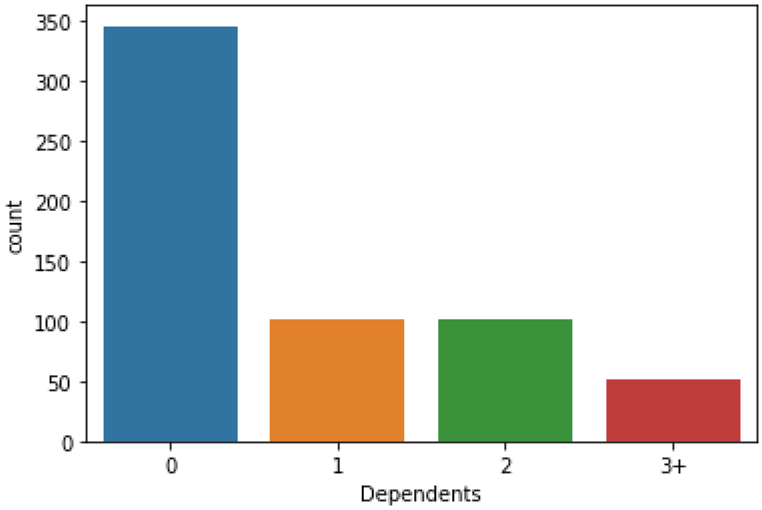
	0	1	2	3	4
summary	count	mean	stddev	min	max
Loan_ID	614	None	None	LP001002	LP002990
Gender	601	None	None	Female	Male
Married	611	None	None	No	Yes
Dependents	599	0.5547445255474452	0.7853289861674311	0	3+
Education	614	None	None	Graduate	Not Graduate
Self_Employed	582	None	None	No	Yes
ApplicantIncome	614	5403.459283387622	6109.041673387181	150	81000
CoapplicantIncome	614	1621.245798027101	2926.2483692241894	0E-8	41667.00000000
LoanAmount	592	146.41216216216216	85.58732523570545	9	700
Loan_Amount_Term	600	342.0	65.12040985461255	12	480
Property_Area	614	None	None	Rural	Urban
Loan_Status	614	None	None	N	Y

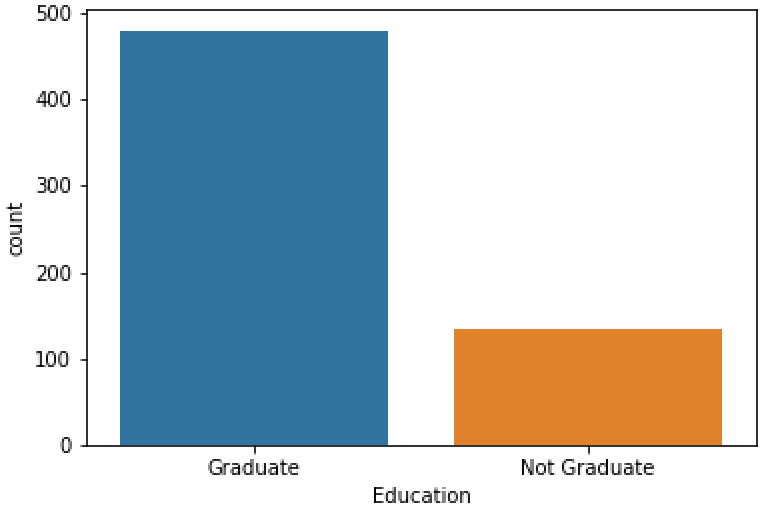
Exploratory Data Analysis:

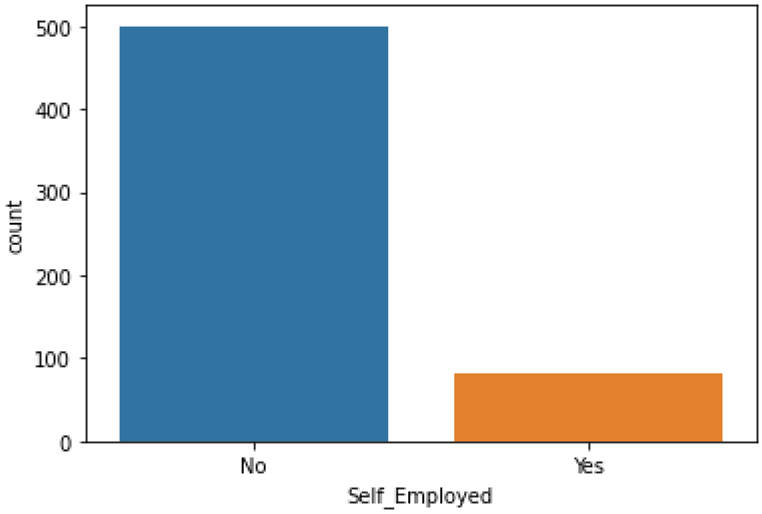
Categorical features:

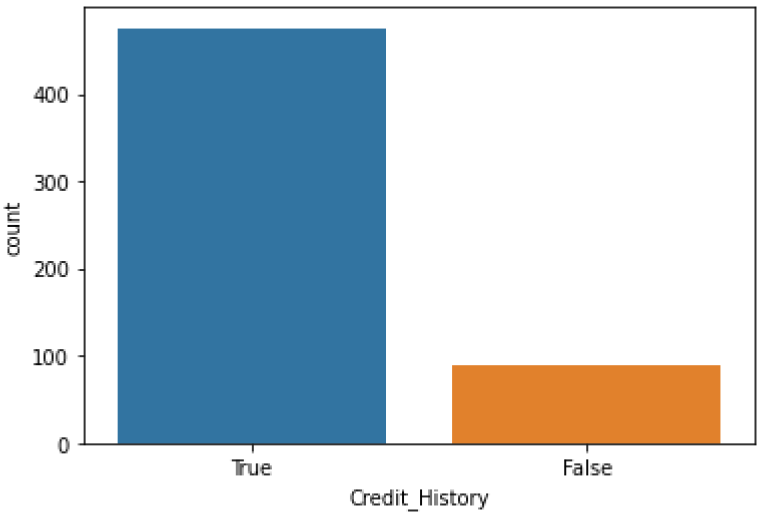


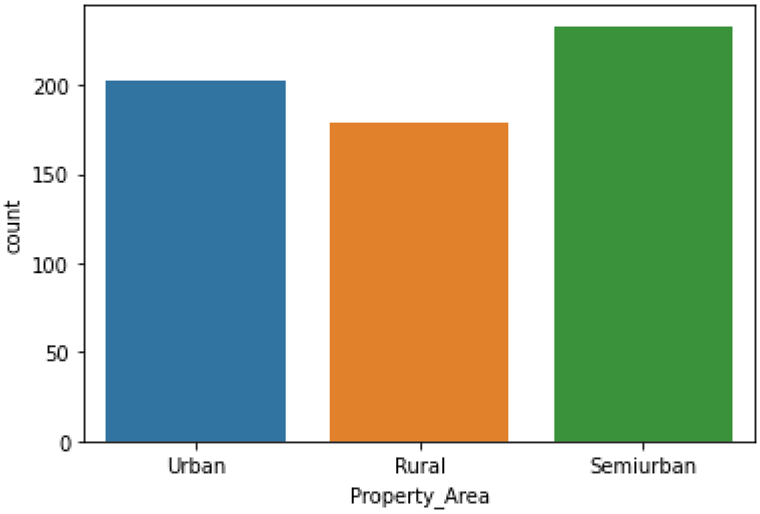


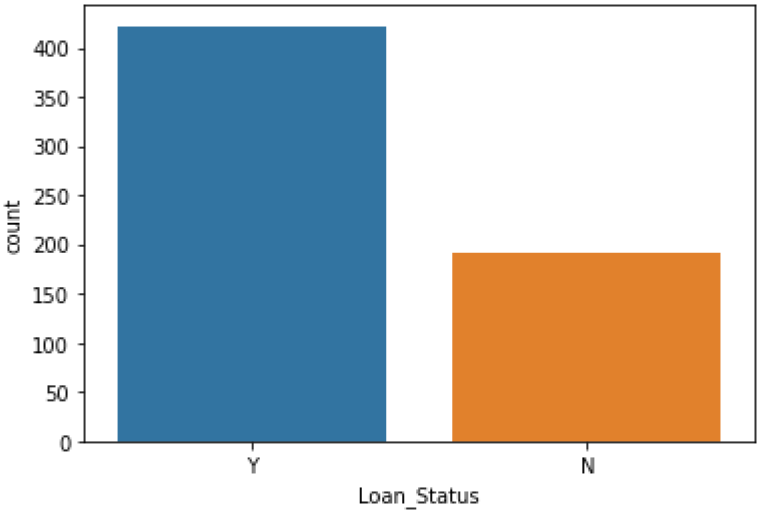

count
350
300
250
200
150
100
50
0
0
1
2
3+
Dependents


count
500
400
300
200
100
0
Graduate
Not Graduate
Education

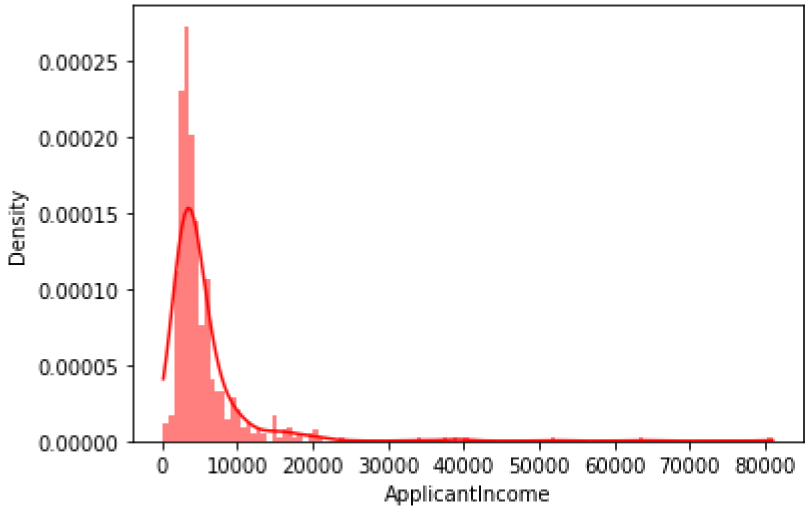

count
500
400
300
200
100
0
No
Yes
Self_Employed

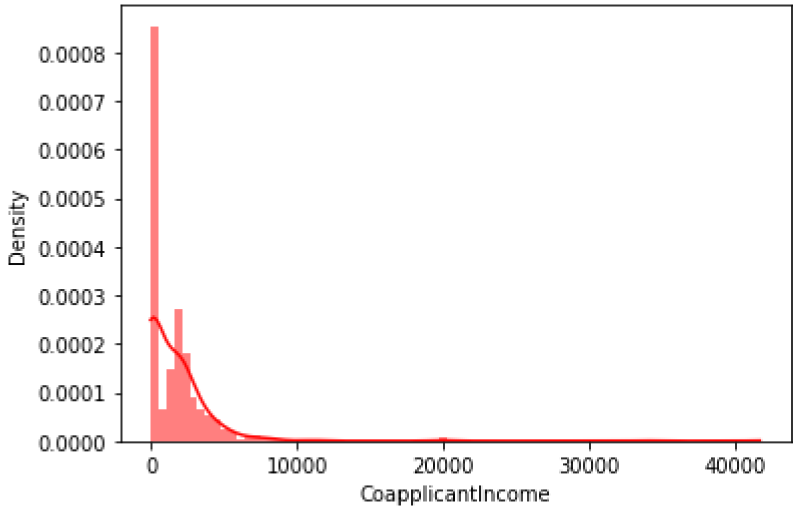

count
400
300
200
100
0
True
False
Credit_History

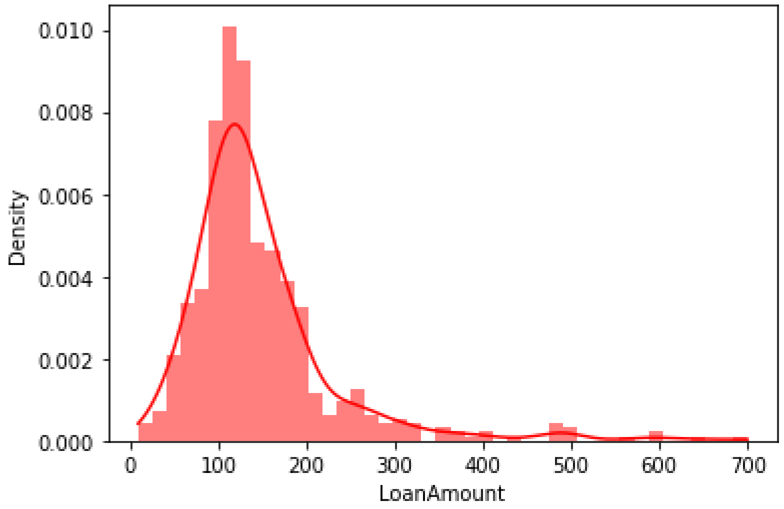

count
200
150
100
50
0
Urban
Rural
Semiurban
Property_Area

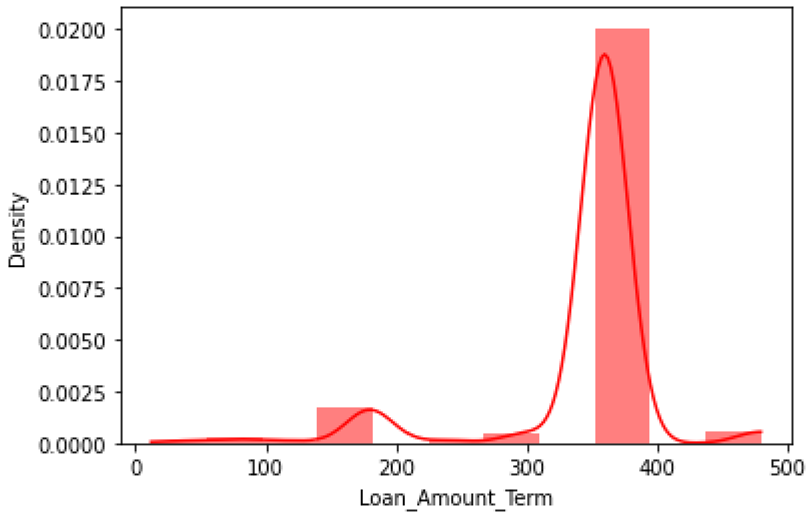

count
400
350
300
250
200
150
100
50
0
Y
N
Loan_Status

Continuous features:





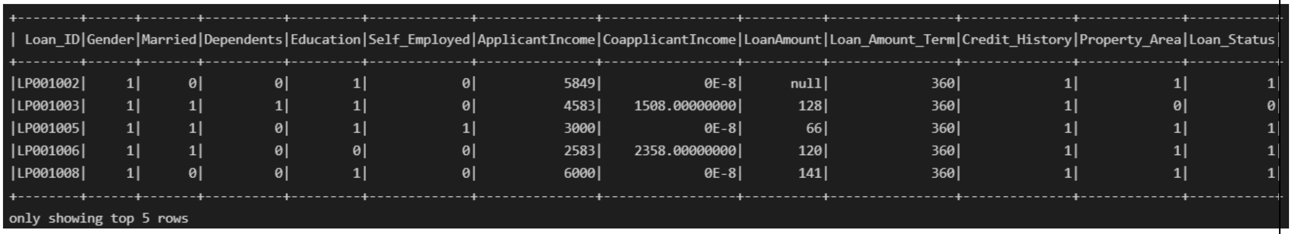




Preprocessing:

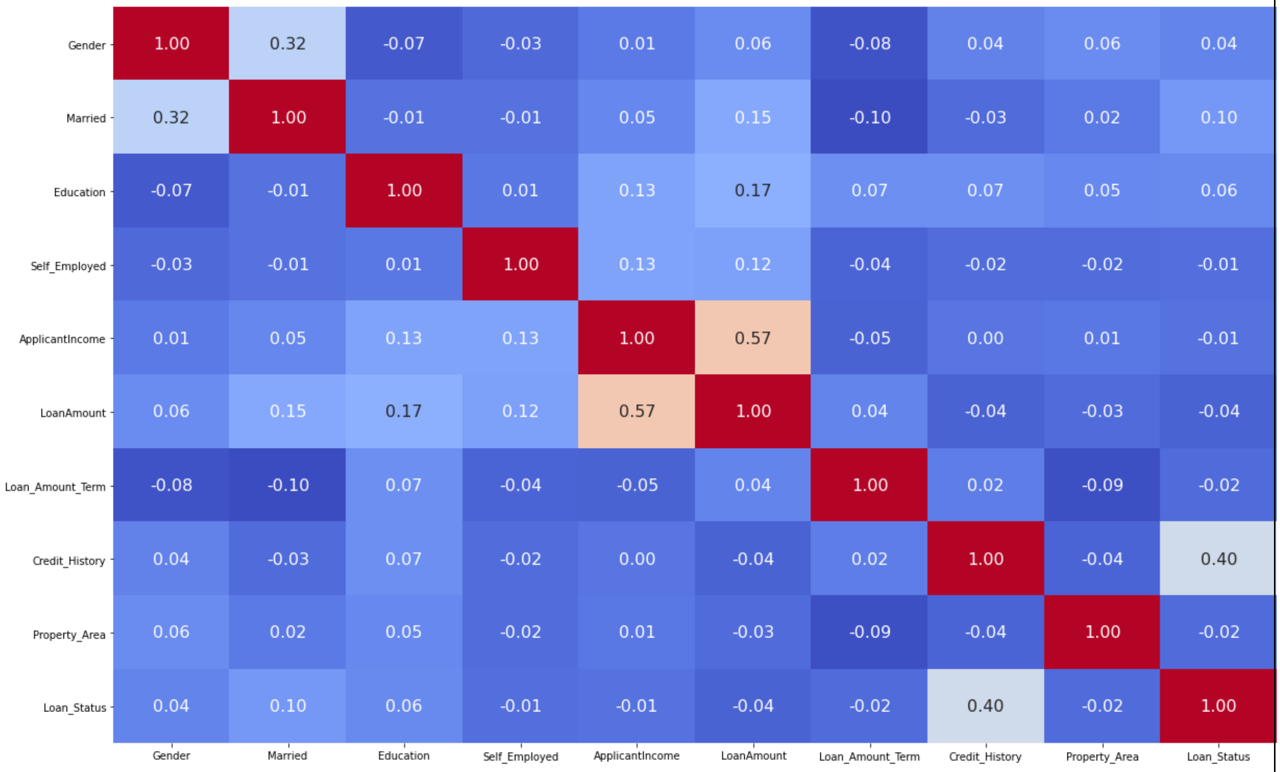
- □ Encoding

```
from pyspark.sql.functions import col, when
dataframe_mysql =
dataframe_mysql.withColumn(dataframe_mysql.columns[1],
when(col(dataframe_mysql.columns[1]) == "Male", 1).otherwise(0))
dataframe_mysql =
dataframe_mysql.withColumn(dataframe_mysql.columns[2],
when(col(dataframe_mysql.columns[2]) == "Yes", 1).otherwise(0))
dataframe_mysql =
dataframe_mysql.withColumn(dataframe_mysql.columns[4],
when(col(dataframe_mysql.columns[4]) == "Graduate", 1).otherwise(0))
dataframe_mysql =
dataframe_mysql.withColumn(dataframe_mysql.columns[5],
when(col(dataframe_mysql.columns[5]) == "Yes", 1).otherwise(0))
dataframe_mysql =
dataframe_mysql.withColumn(dataframe_mysql.columns[10],
when(col(dataframe_mysql.columns[10]) == "true", 1).otherwise(0))
dataframe_mysql =
dataframe_mysql.withColumn(dataframe_mysql.columns[11],
when(col(dataframe_mysql.columns[11]) == "Urban", 1).otherwise(0))
dataframe_mysql =
dataframe_mysql.withColumn(dataframe_mysql.columns[12],
when(col(dataframe_mysql.columns[12]) == "Y", 1).otherwise(0))
dataframe_mysql=dataframe_mysql.na.replace('0E-8', '0')
dataframe_mysql.show(5)
```



```
+--------+------+-------+----------+---------+-------------+---------------+-----------------+----------+----------------+--------------+-------------+-----------+
| Loan_ID|Gender|Married|Dependents|Education|Self_Employed|ApplicantIncome|CoapplicantIncome|LoanAmount|Loan_Amount_Term|Credit_History|Property_Area|Loan_Status|
+--------+------+-------+----------+---------+-------------+---------------+-----------------+----------+----------------+--------------+-------------+-----------+
|LP001002|     1|      0|         0|        1|            0|           5849|             0E-8|      null|             360|             1|            1|          1|
|LP001003|     1|      1|         1|        1|            0|           4583|    1508.00000000|       128|             360|             1|            0|          0|
|LP001005|     1|      1|         0|        1|            1|           3000|             0E-8|        66|             360|             1|            1|          1|
|LP001006|     1|      1|         0|        0|            0|           2583|    2358.00000000|       120|             360|             1|            1|          1|
|LP001008|     1|      0|         0|        1|            0|           6000|             0E-8|       141|             360|             1|            1|          1|
+--------+------+-------+----------+---------+-------------+---------------+-----------------+----------+----------------+--------------+-------------+-----------+
only showing top 5 rows
```

□ Correlation:



	Gender	Married	Education	Self_Employed	ApplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
Gender	1.00	0.32	-0.07	-0.03	0.01	0.06	-0.08	0.04	0.06	0.04
Married	0.32	1.00	-0.01	-0.01	0.05	0.15	-0.10	-0.03	0.02	0.10
Education	-0.07	-0.01	1.00	0.01	0.13	0.17	0.07	0.07	0.05	0.06
Self_Employed	-0.03	-0.01	0.01	1.00	0.13	0.12	-0.04	-0.02	-0.02	-0.01
ApplicantIncome	0.01	0.05	0.13	0.13	1.00	0.57	-0.05	0.00	0.01	-0.01
LoanAmount	0.06	0.15	0.17	0.12	0.57	1.00	0.04	-0.04	-0.03	-0.04
Loan_Amount_Term	-0.08	-0.10	0.07	-0.04	-0.05	0.04	1.00	0.02	-0.09	-0.02
Credit_History	0.04	-0.03	0.07	-0.02	0.00	-0.04	0.02	1.00	-0.04	0.40
Property_Area	0.06	0.02	0.05	-0.02	0.01	-0.03	-0.09	-0.04	1.00	-0.02
Loan_Status	0.04	0.10	0.06	-0.01	-0.01	-0.04	-0.02	0.40	-0.02	1.00

Loanamount and Applicantincome are positively correlated.

Train_Test Split:

```
features = dataframe_mysql.drop('Loan_Status')
output = assembler.transform(dataframe_mysql)
output= output.select("features", "Loan_Status")
train_df,test_df = output.randomSplit([0.7, 0.3])
```

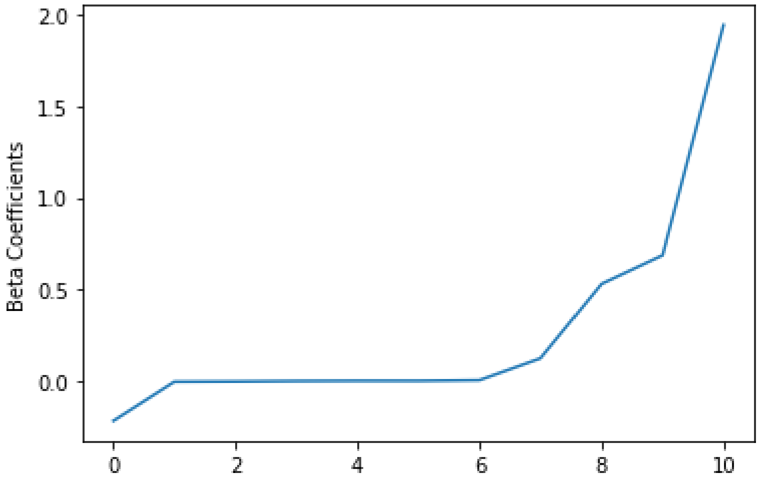
Model Building:

1. Logistic Regression:

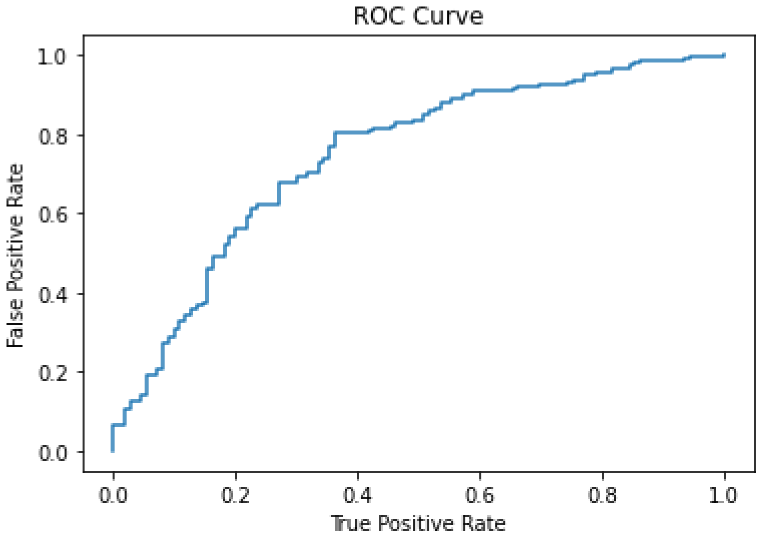
Fitting for Training data:

```
from pyspark.ml.classification import LogisticRegression
lr = LogisticRegression(featuresCol = 'features', labelCol =
'Loan_Status', maxIter=10)
lrModel = lr.fit(train_df)
```

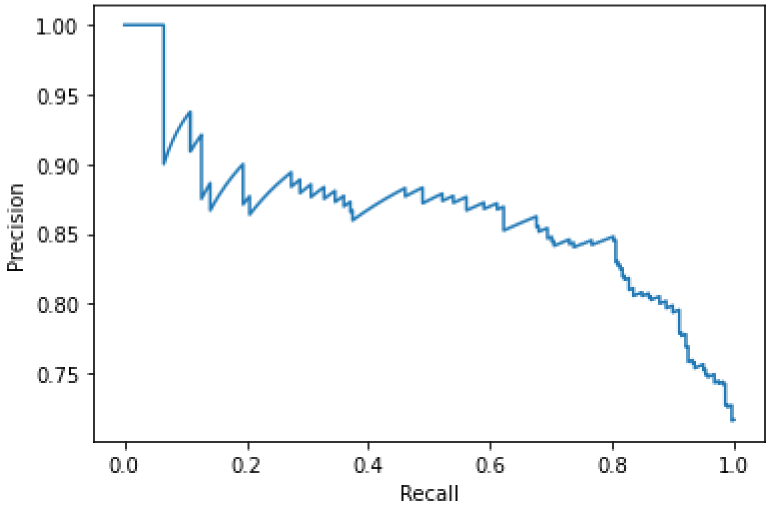
Beta coefficients:



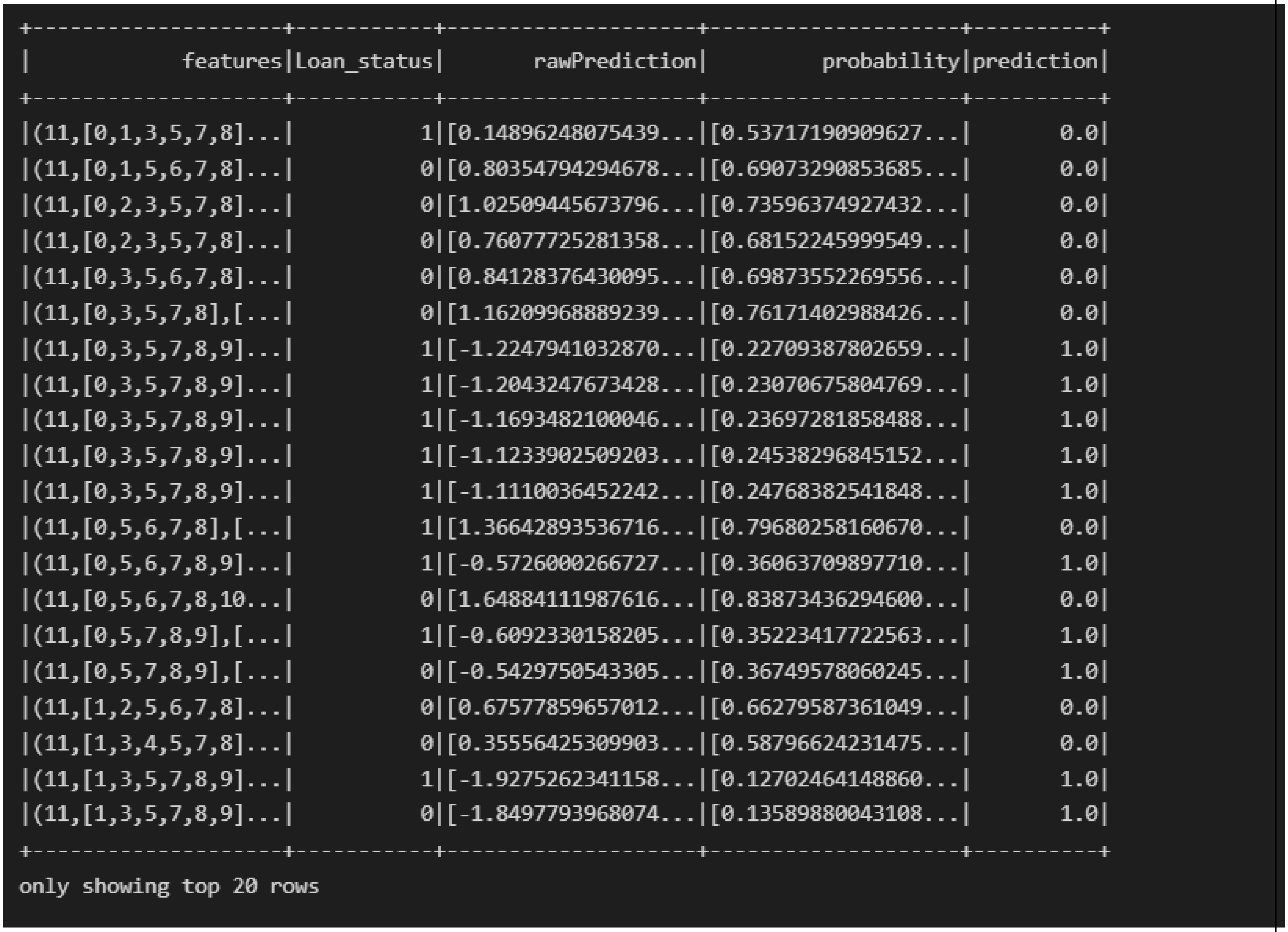
ROC curve:



PR curve:

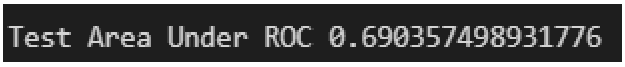


Predicted the values for Test Dataset



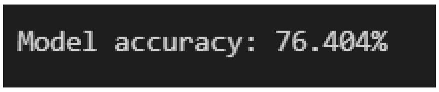
```
+--------------------+-----------+--------------------+--------------------+----------+
|            features|Loan_status|       rawPrediction|         probability|prediction|
+--------------------+-----------+--------------------+--------------------+----------+
|(11,[0,1,3,5,7,8]...|          1|[0.14896248075439...|[0.53717190909627...|       0.0|
|(11,[0,1,5,6,7,8]...|          0|[0.80354794294678...|[0.69073290853685...|       0.0|
|(11,[0,2,3,5,7,8]...|          0|[1.02509445673796...|[0.73596374927432...|       0.0|
|(11,[0,2,3,5,7,8]...|          0|[0.76077725281358...|[0.68152245999549...|       0.0|
|(11,[0,3,5,6,7,8]...|          0|[0.84128376430095...|[0.69873552269556...|       0.0|
|(11,[0,3,5,7,8],[...|          0|[1.16209968889239...|[0.76171402988426...|       0.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.2247941032870...|[0.22709387802659...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.2043247673428...|[0.23070675804769...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.1693482100046...|[0.23697281858488...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.1233902509203...|[0.24538296845152...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.1110036452242...|[0.24768382541848...|       1.0|
|(11,[0,5,6,7,8],[...|          1|[1.36642893536716...|[0.79680258160670...|       0.0|
|(11,[0,5,6,7,8,9]...|          1|[-0.5726000266727...|[0.36063709897710...|       1.0|
|(11,[0,5,6,7,8,10...|          0|[1.64884111987616...|[0.83873436294600...|       0.0|
|(11,[0,5,7,8,9],[...|          1|[-0.6092330158205...|[0.35223417722563...|       1.0|
|(11,[0,5,7,8,9],[...|          0|[-0.5429750543305...|[0.36749578060245...|       1.0|
|(11,[1,2,5,6,7,8]...|          0|[0.67577859657012...|[0.66279587361049...|       0.0|
|(11,[1,3,4,5,7,8]...|          0|[0.35556425309903...|[0.58796624231475...|       0.0|
|(11,[1,3,5,7,8,9]...|          1|[-1.9275262341158...|[0.12702464148860...|       1.0|
|(11,[1,3,5,7,8,9]...|          0|[-1.8497793968074...|[0.13589880043108...|       1.0|
+--------------------+-----------+--------------------+--------------------+----------+
only showing top 20 rows
```

ROC for test:



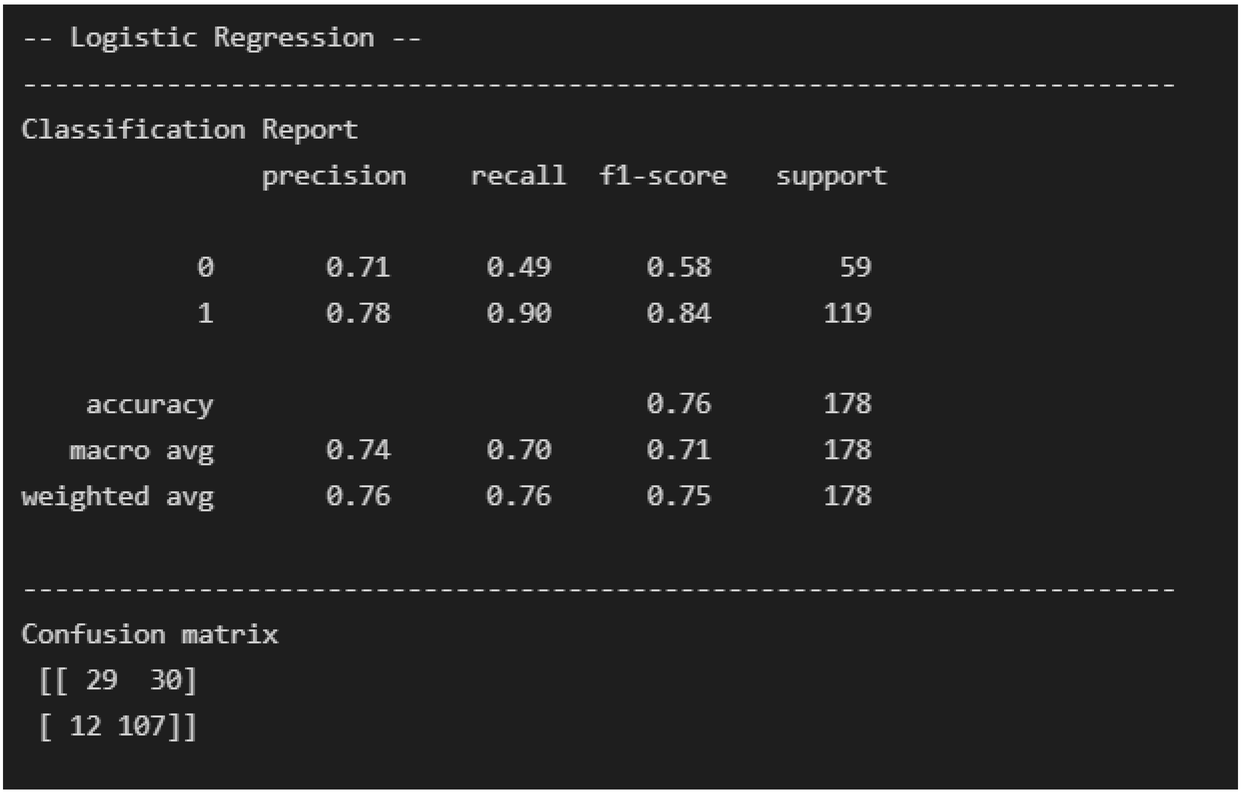
```
Test Area Under ROC 0.690357498931776
```

Accuracy:



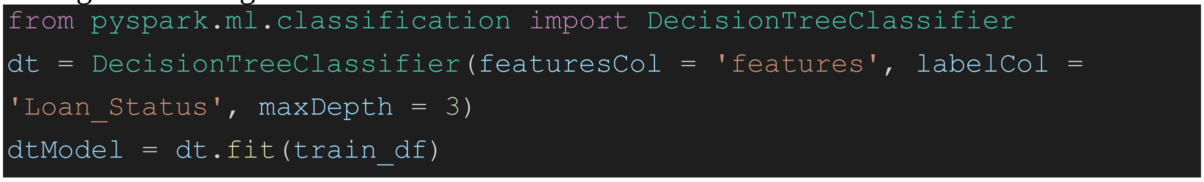
```
Model accuracy: 76.404%
```

Classification Report and Confusion matrix:



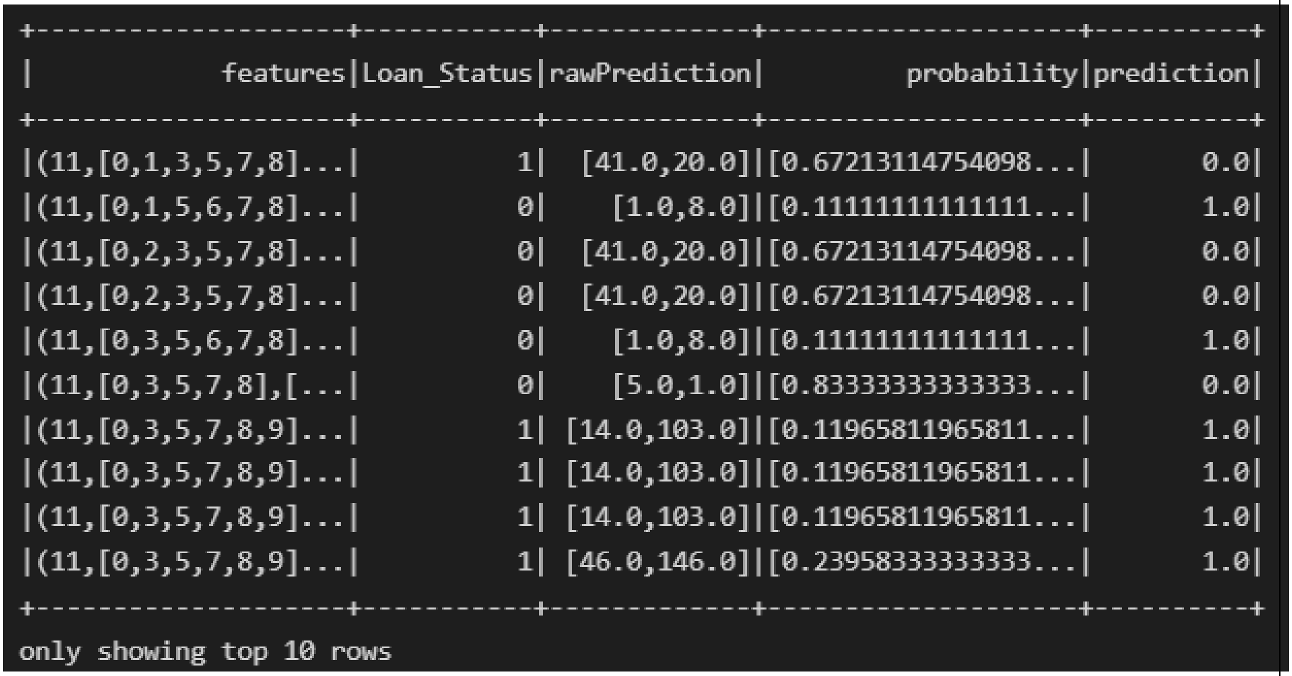
```
-- Logistic Regression --
--------------------------------------------------------------------------
Classification Report
               precision    recall  f1-score   support

           0       0.71      0.49      0.58        59
           1       0.78      0.90      0.84       119

    accuracy                           0.76       178
   macro avg       0.74      0.70      0.71       178
weighted avg       0.76      0.76      0.75       178


--------------------------------------------------------------------------
Confusion matrix
 [[ 29  30]
 [ 12 107]]
```

2. **Decision Tree**
Fitting for Training data:



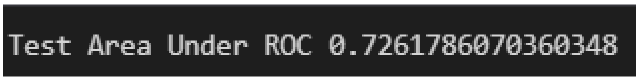
```
from pyspark.ml.classification import DecisionTreeClassifier
dt = DecisionTreeClassifier(featuresCol = 'features', labelCol =
'Loan_Status', maxDepth = 3)
dtModel = dt.fit(train_df)
```

Predicted the values for Test Dataset



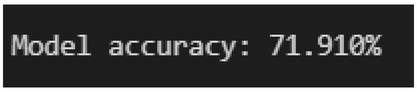
```
+--------------------+-----------+-------------+--------------------+----------+
|            features|Loan_Status|rawPrediction|         probability|prediction|
+--------------------+-----------+-------------+--------------------+----------+
|(11,[0,1,3,5,7,8]...|          1|  [41.0,20.0]|[0.67213114754098...|       0.0|
|(11,[0,1,5,6,7,8]...|          0|    [1.0,8.0]|[0.11111111111111...|       1.0|
|(11,[0,2,3,5,7,8]...|          0|  [41.0,20.0]|[0.67213114754098...|       0.0|
|(11,[0,2,3,5,7,8]...|          0|  [41.0,20.0]|[0.67213114754098...|       0.0|
|(11,[0,3,5,6,7,8]...|          0|    [1.0,8.0]|[0.11111111111111...|       1.0|
|(11,[0,3,5,7,8],[...|          0|    [5.0,1.0]|[0.83333333333333...|       0.0|
|(11,[0,3,5,7,8,9]...|          1| [14.0,103.0]|[0.11965811965811...|       1.0|
|(11,[0,3,5,7,8,9]...|          1| [14.0,103.0]|[0.11965811965811...|       1.0|
|(11,[0,3,5,7,8,9]...|          1| [14.0,103.0]|[0.11965811965811...|       1.0|
|(11,[0,3,5,7,8,9]...|          1| [46.0,146.0]|[0.23958333333333...|       1.0|
+--------------------+-----------+-------------+--------------------+----------+
only showing top 10 rows
```

ROC for test:



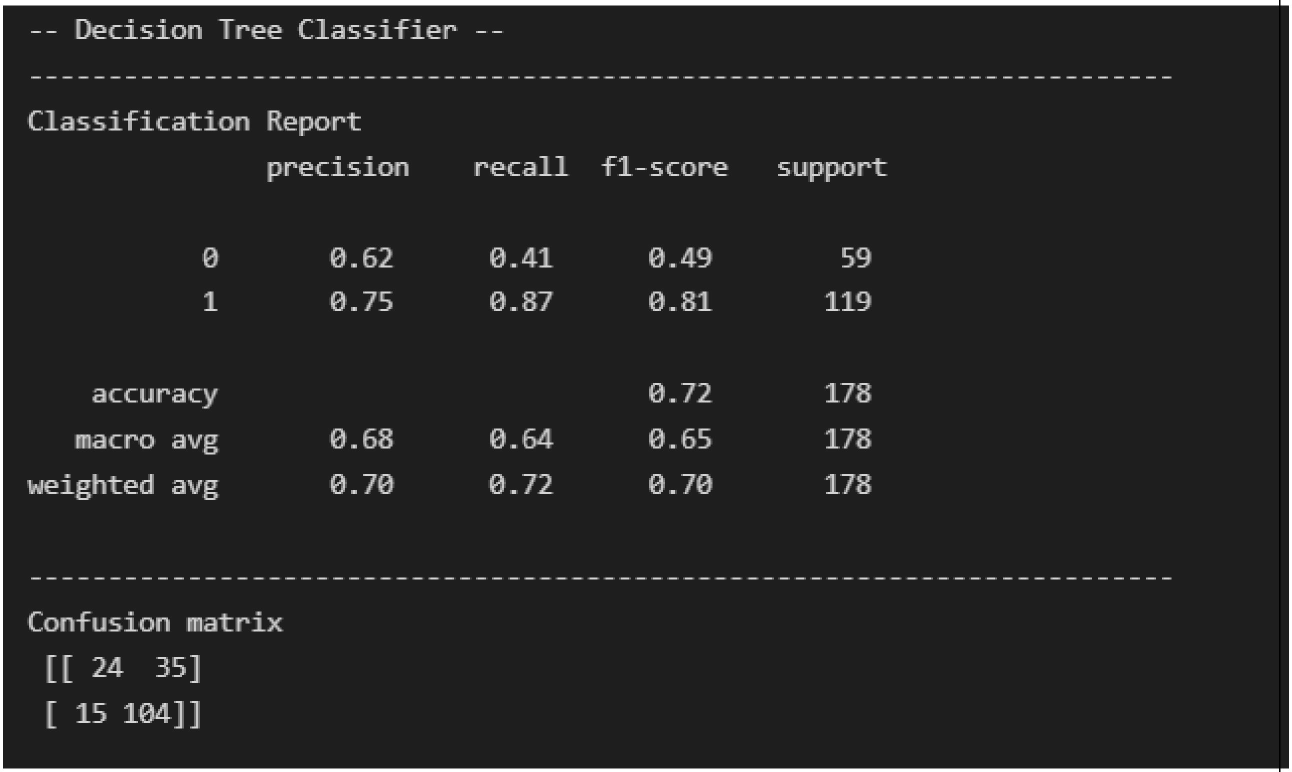
```
Test Area Under ROC 0.7261786070360348
```

Accuracy:



```
Model accuracy: 71.910%
```

Classification Report and Confusion matrix



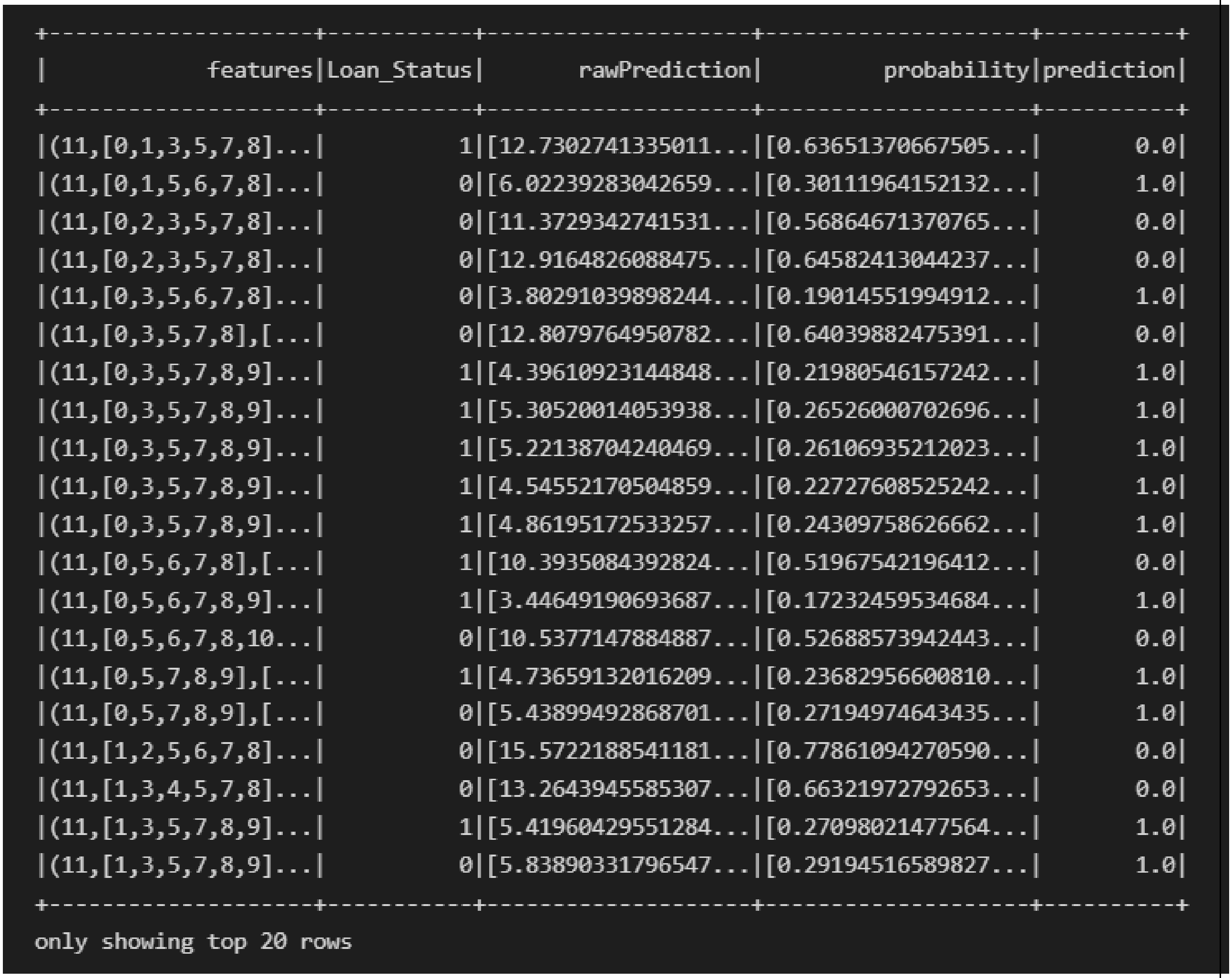
```
-- Decision Tree Classifier --
----------------------------------------------------------------------
Classification Report
               precision    recall  f1-score   support

           0       0.62      0.41      0.49        59
           1       0.75      0.87      0.81       119

    accuracy                           0.72       178
   macro avg       0.68      0.64      0.65       178
weighted avg       0.70      0.72      0.70       178

----------------------------------------------------------------------
Confusion matrix
 [[ 24  35]
 [ 15 104]]
```

3. **Random Forest**

Fitting for Training data:

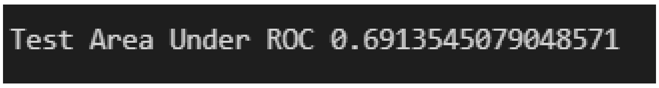
```
from pyspark.ml.classification import RandomForestClassifier
rf = RandomForestClassifier(featuresCol = 'features', labelCol = 
'Loan_Status')
rfModel = rf.fit(train_df)
```

Predicted the values for Test Dataset



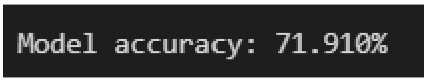
```
+--------------------+-----------+--------------------+--------------------+----------+
|            features|Loan_Status|       rawPrediction|         probability|prediction|
+--------------------+-----------+--------------------+--------------------+----------+
|(11,[0,1,3,5,7,8]...|          1|[12.7302741335011...|[0.63651370667505...|       0.0|
|(11,[0,1,5,6,7,8]...|          0|[6.02239283042659...|[0.30111964152132...|       1.0|
|(11,[0,2,3,5,7,8]...|          0|[11.3729342741531...|[0.56864671370765...|       0.0|
|(11,[0,2,3,5,7,8]...|          0|[12.9164826088475...|[0.64582413044237...|       0.0|
|(11,[0,3,5,6,7,8]...|          0|[3.80291039898244...|[0.19014551994912...|       1.0|
|(11,[0,3,5,7,8],[...|          0|[12.8079764950782...|[0.64039882475391...|       0.0|
|(11,[0,3,5,7,8,9]...|          1|[4.39610923144848...|[0.21980546157242...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[5.30520014053938...|[0.26526000702696...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[5.22138704240469...|[0.26106935212023...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[4.54552170504859...|[0.22727608525242...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[4.86195172533257...|[0.24309758626662...|       1.0|
|(11,[0,5,6,7,8],[...|          1|[10.3935084392824...|[0.51967542196412...|       0.0|
|(11,[0,5,6,7,8,9]...|          1|[3.44649190693687...|[0.17232459534684...|       1.0|
|(11,[0,5,6,7,8,10...|          0|[10.5377147884887...|[0.52688573942443...|       0.0|
|(11,[0,5,7,8,9],[...|          1|[4.73659132016209...|[0.23682956600810...|       1.0|
|(11,[0,5,7,8,9],[...|          0|[5.43899492868701...|[0.27194974643435...|       1.0|
|(11,[1,2,5,6,7,8]...|          0|[15.5722188541181...|[0.77861094270590...|       0.0|
|(11,[1,3,4,5,7,8]...|          0|[13.2643945585307...|[0.66321972792653...|       0.0|
|(11,[1,3,5,7,8,9]...|          1|[5.41960429551284...|[0.27098021477564...|       1.0|
|(11,[1,3,5,7,8,9]...|          0|[5.83890331796547...|[0.29194516589827...|       1.0|
+--------------------+-----------+--------------------+--------------------+----------+
only showing top 20 rows
```

ROC for test:



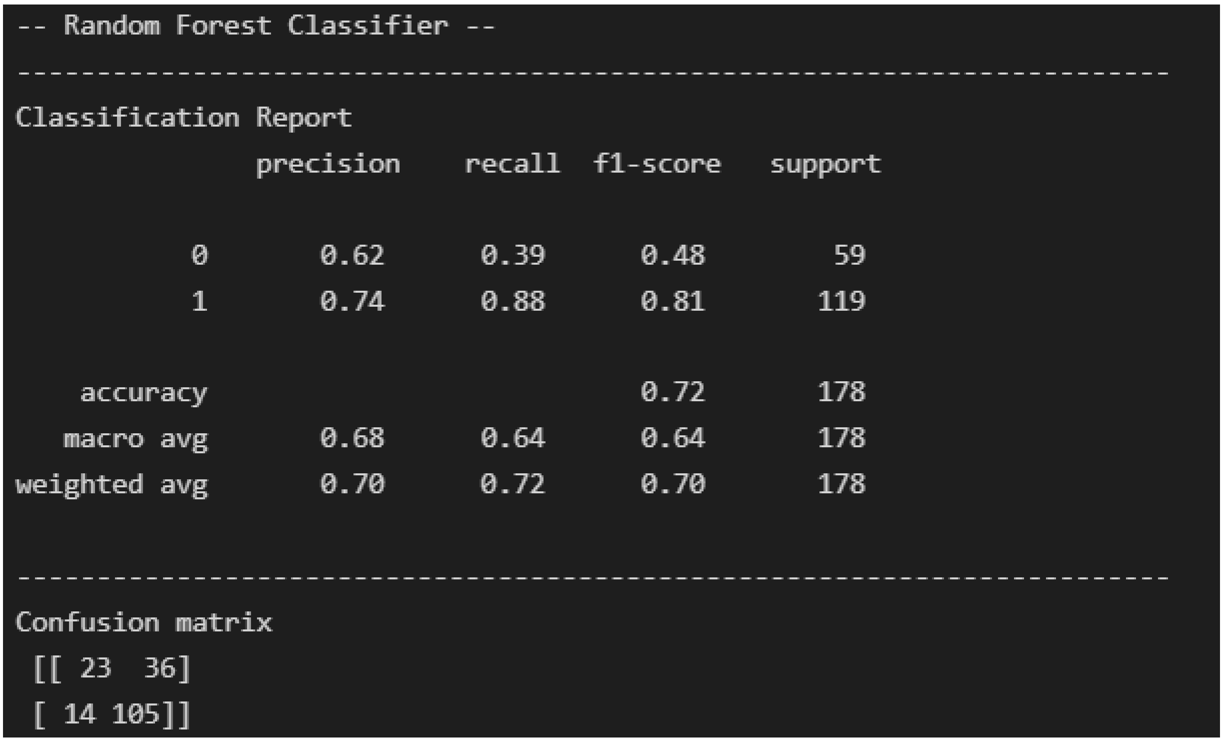
```
Test Area Under ROC 0.6913545079048571
```

Accuracy:



```
Model accuracy: 71.910%
```

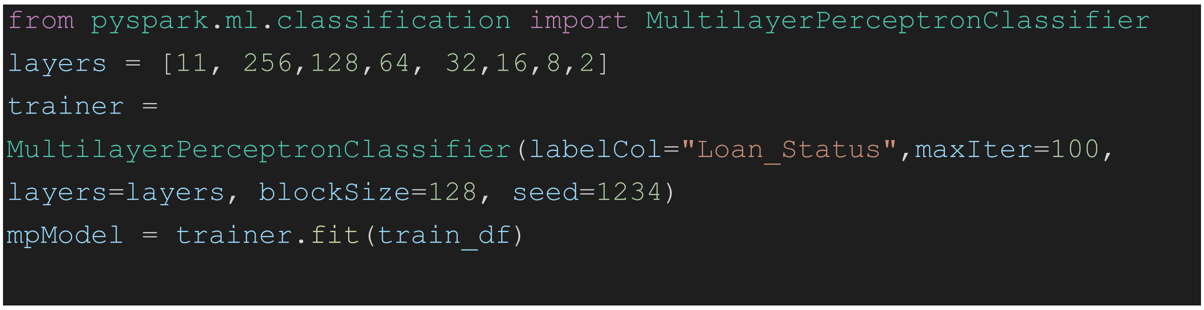
Classification Report and Confusion matrix



```
-- Random Forest Classifier --
----------------------------------------------------------------------
Classification Report
               precision    recall  f1-score   support

           0       0.62      0.39      0.48        59
           1       0.74      0.88      0.81       119

    accuracy                           0.72       178
   macro avg       0.68      0.64      0.64       178
weighted avg       0.70      0.72      0.70       178

----------------------------------------------------------------------
Confusion matrix
 [[ 23  36]
 [ 14 105]]
```

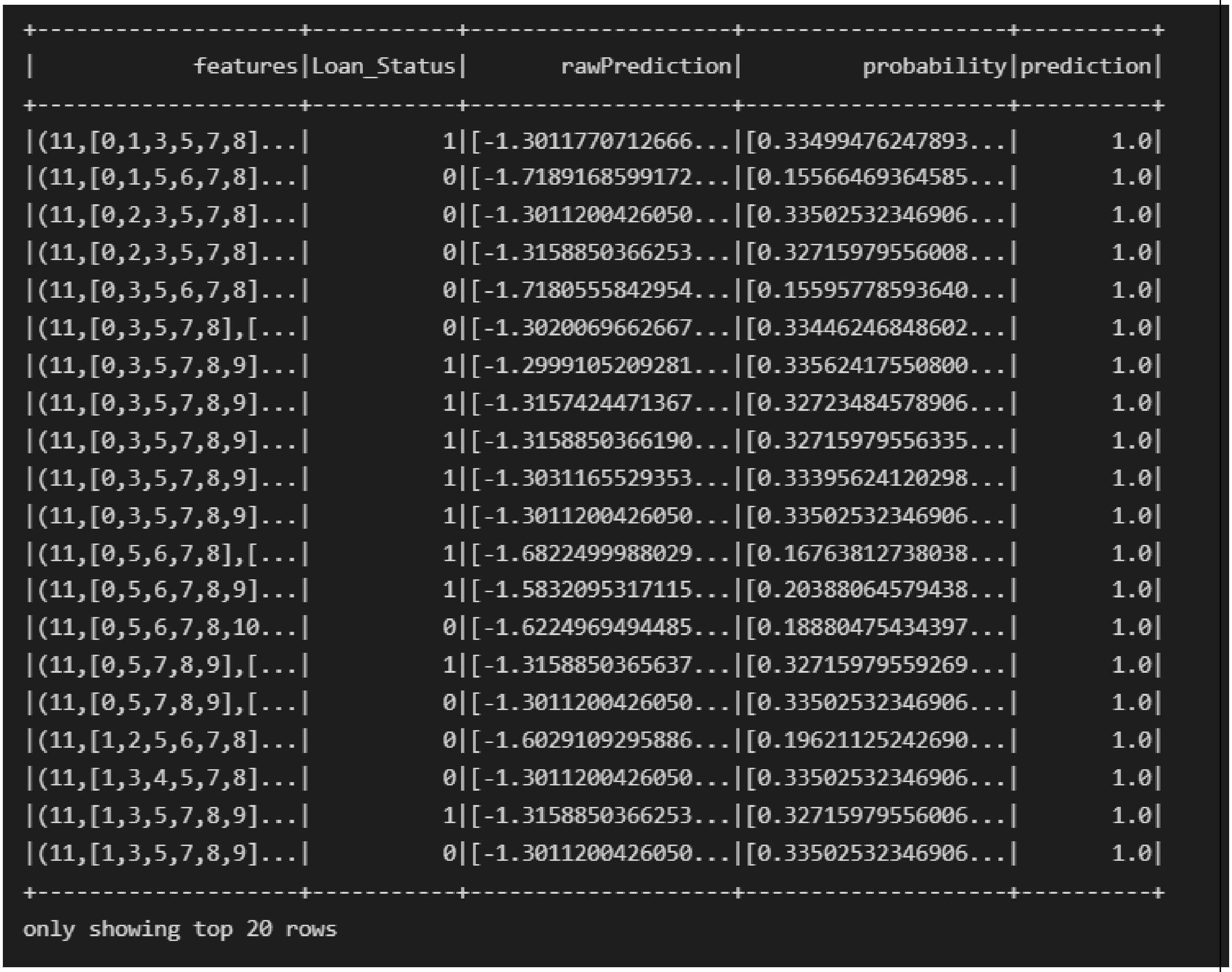
4. **Multi-Layer Perceptron**

Fitting for Training data:



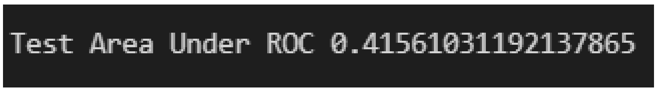
```
from pyspark.ml.classification import MultilayerPerceptronClassifier
layers = [11, 256,128,64, 32,16,8,2]
trainer =
MultilayerPerceptronClassifier(labelCol="Loan_Status",maxIter=100,
layers=layers, blockSize=128, seed=1234)
mpModel = trainer.fit(train_df)
```

Predicted the values for Test Dataset



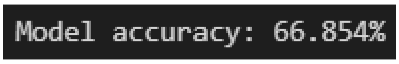
```
+--------------------+-----------+--------------------+--------------------+----------+
|            features|Loan_Status|       rawPrediction|         probability|prediction|
+--------------------+-----------+--------------------+--------------------+----------+
|(11,[0,1,3,5,7,8]...|          1|[-1.3011770712666...|[0.33499476247893...|       1.0|
|(11,[0,1,5,6,7,8]...|          0|[-1.7189168599172...|[0.15566469364585...|       1.0|
|(11,[0,2,3,5,7,8]...|          0|[-1.3011200426050...|[0.33502532346906...|       1.0|
|(11,[0,2,3,5,7,8]...|          0|[-1.3158850366253...|[0.32715979556008...|       1.0|
|(11,[0,3,5,6,7,8]...|          0|[-1.7180555842954...|[0.15595778593640...|       1.0|
|(11,[0,3,5,7,8],[...|          0|[-1.3020069662667...|[0.33446246848602...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.2999105209281...|[0.33562417550800...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.3157424471367...|[0.32723484578906...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.3158850366190...|[0.32715979556335...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.3031165529353...|[0.33395624120298...|       1.0|
|(11,[0,3,5,7,8,9]...|          1|[-1.3011200426050...|[0.33502532346906...|       1.0|
|(11,[0,5,6,7,8],[...|          1|[-1.6822499988029...|[0.16763812738038...|       1.0|
|(11,[0,5,6,7,8,9]...|          1|[-1.5832095317115...|[0.20388064579438...|       1.0|
|(11,[0,5,6,7,8,10...|          0|[-1.6224969494485...|[0.18880475434397...|       1.0|
|(11,[0,5,7,8,9],[...|          1|[-1.3158850365637...|[0.32715979559269...|       1.0|
|(11,[0,5,7,8,9],[...|          0|[-1.3011200426050...|[0.33502532346906...|       1.0|
|(11,[1,2,5,6,7,8]...|          0|[-1.6029109295886...|[0.19621125242690...|       1.0|
|(11,[1,3,4,5,7,8]...|          0|[-1.3011200426050...|[0.33502532346906...|       1.0|
|(11,[1,3,5,7,8,9]...|          1|[-1.3158850366253...|[0.32715979556006...|       1.0|
|(11,[1,3,5,7,8,9]...|          0|[-1.3011200426050...|[0.33502532346906...|       1.0|
+--------------------+-----------+--------------------+--------------------+----------+
only showing top 20 rows
```

ROC for test:



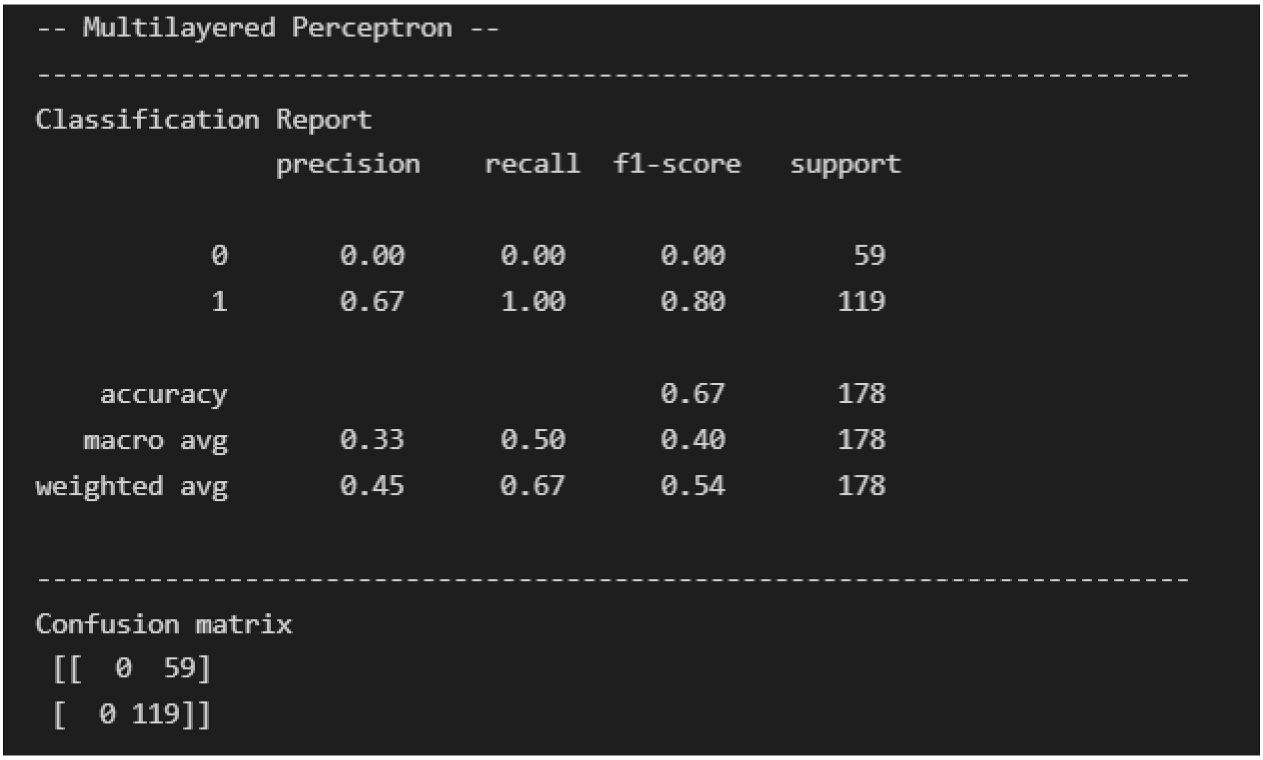
```
Test Area Under ROC 0.41561031192137865
```

Accuracy:



```
Model accuracy: 66.854%
```

Classification Report and Confusion matrix:



```
-- Multilayered Perceptron --
------------------------------------------------------------------------
Classification Report
               precision    recall  f1-score   support


           0       0.00      0.00      0.00        59
           1       0.67      1.00      0.80       119

    accuracy                           0.67       178
   macro avg       0.33      0.50      0.40       178
weighted avg       0.45      0.67      0.54       178


------------------------------------------------------------------------
Confusion matrix
 [[  0  59]
 [  0 119]]
```