silatcha_russelle_et_klein_arthur_icc

February 6, 2024

1 Projet de diagnostics de diabètes (Silatcha Russelle et Klein Arthur)

1.1 Initialisation de l'environnement

[57]:

```
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
!wget -q http://archive.apache.org/dist/spark/spark-3.1.1/spark-3.1.
 ↪1-bin-hadoop3.2.tgz
!tar xf spark-3.1.1-bin-hadoop3.2.tgz
!pip install -q findspark
```

[58]:

```
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.1.1-bin-hadoop3.2"
```

[59]:

```
import findspark
findspark.init()
```

[60]:

```
import pyspark
from pyspark.sql import SparkSession

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import pyspark.sql.functions as f
import seaborn as sns
from sklearn.metrics import confusion_matrix, roc_curve, auc
from pyspark.ml import Pipeline
from pyspark.ml.feature import StandardScaler, VectorAssembler
from pyspark.ml.classification import DecisionTreeClassifier, GBTClassifier,␣
 ↪RandomForestClassifier, MultilayerPerceptronClassifier
from pyspark.ml.evaluation import BinaryClassificationEvaluator
from pyspark.ml.tuning import ParamGridBuilder, CrossValidator
```

1.2 Import des données

Import du fichier diabetes.csv

[61]:

```
from google.colab import files
files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving diabetes.csv to diabetes (1).csv
```

[61]:

```
{'diabetes (1).csv': b'Pregnancies,Glucose,BloodPressure,SkinThickness,Insulin,B
MI,DiabetesPedigreeFunction,Age,Outcome\r\n6,148,72,35,0,33.6,0.627,50,1\r\n1,85
,66,29,0,26.6,0.351,31,0\r\n8,183,64,0,0,23.3,0.672,32,1\r\n1,89,66,23,94,28.1,0
.167,21,0\r\n0,137,40,35,168,43.1,2.288,33,1\r\n5,116,74,0,0,25.6,0.201,30,0\r\n
3,78,50,32,88,31,0.248,26,1\r\n10,115,0,0,0,35.3,0.134,29,0\r\n2,197,70,45,543,3
0.5,0.158,53,1\r\n8,125,96,0,0,0,0.232,54,1\r\n4,110,92,0,0,37.6,0.191,30,0\r\n1
0,168,74,0,0,38,0.537,34,1\r\n10,139,80,0,0,27.1,1.441,57,0\r\n1,189,60,23,846,3
0.1,0.398,59,1\r\n5,166,72,19,175,25.8,0.587,51,1\r\n7,100,0,0,0,30,0.484,32,1\r
\n0,118,84,47,230,45.8,0.551,31,1\r\n7,107,74,0,0,29.6,0.254,31,1\r\n1,103,30,38
,83,43.3,0.183,33,0\r\n1,115,70,30,96,34.6,0.529,32,1\r\n3,126,88,41,235,39.3,0.
704,27,0\r\n8,99,84,0,0,35.4,0.388,50,0\r\n7,196,90,0,0,39.8,0.451,41,1\r\n9,119
,80,35,0,29,0.263,29,1\r\n11,143,94,33,146,36.6,0.254,51,1\r\n10,125,70,26,115,3
1.1,0.205,41,1\r\n7,147,76,0,0,39.4,0.257,43,1\r\n1,97,66,15,140,23.2,0.487,22,0
\r\n13,145,82,19,110,22.2,0.245,57,0\r\n5,117,92,0,0,34.1,0.337,38,0\r\n5,109,75
,26,0,36,0.546,60,0\r\n3,158,76,36,245,31.6,0.851,28,1\r\n3,88,58,11,54,24.8,0.2
67,22,0\r\n6,92,92,0,0,19.9,0.188,28,0\r\n10,122,78,31,0,27.6,0.512,45,0\r\n4,10
3,60,33,192,24,0.966,33,0\r\n11,138,76,0,0,33.2,0.42,35,0\r\n9,102,76,37,0,32.9,
0.665,46,1\r\n2,90,68,42,0,38.2,0.503,27,1\r\n4,111,72,47,207,37.1,1.39,56,1\r\n
3,180,64,25,70,34,0.271,26,0\r\n7,133,84,0,0,40.2,0.696,37,0\r\n7,106,92,18,0,22
.7,0.235,48,0\r\n9,171,110,24,240,45.4,0.721,54,1\r\n7,159,64,0,0,27.4,0.294,40,
0\r\n0,180,66,39,0,42,1.893,25,1\r\n1,146,56,0,0,29.7,0.564,29,0\r\n2,71,70,27,0
,28,0.586,22,0\r\n7,103,66,32,0,39.1,0.344,31,1\r\n7,105,0,0,0,0,0.305,24,0\r\n1
,103,80,11,82,19.4,0.491,22,0\r\n1,101,50,15,36,24.2,0.526,26,0\r\n5,88,66,21,23
,24.4,0.342,30,0\r\n8,176,90,34,300,33.7,0.467,58,1\r\n7,150,66,42,342,34.7,0.71
8,42,0\r\n1,73,50,10,0,23,0.248,21,0\r\n7,187,68,39,304,37.7,0.254,41,1\r\n0,100
,88,60,110,46.8,0.962,31,0\r\n0,146,82,0,0,40.5,1.781,44,0\r\n0,105,64,41,142,41
.5,0.173,22,0\r\n2,84,0,0,0,0,0.304,21,0\r\n8,133,72,0,0,32.9,0.27,39,1\r\n5,44,
62,0,0,25,0.587,36,0\r\n2,141,58,34,128,25.4,0.699,24,0\r\n7,114,66,0,0,32.8,0.2
58,42,1\r\n5,99,74,27,0,29,0.203,32,0\r\n0,109,88,30,0,32.5,0.855,38,1\r\n2,109,
92,0,0,42.7,0.845,54,0\r\n1,95,66,13,38,19.6,0.334,25,0\r\n4,146,85,27,100,28.9,
0.189,27,0\r\n2,100,66,20,90,32.9,0.867,28,1\r\n5,139,64,35,140,28.6,0.411,26,0\
r\n13,126,90,0,0,43.4,0.583,42,1\r\n4,129,86,20,270,35.1,0.231,23,0\r\n1,79,75,3
0,0,32,0.396,22,0\r\n1,0,48,20,0,24.7,0.14,22,0\r\n7,62,78,0,0,32.6,0.391,41,0\r
\n5,95,72,33,0,37.7,0.37,27,0\r\n0,131,0,0,0,43.2,0.27,26,1\r\n2,112,66,22,0,25,
0.307,24,0\r\n3,113,44,13,0,22.4,0.14,22,0\r\n2,74,0,0,0,0,0.102,22,0\r\n7,83,78
,26,71,29.3,0.767,36,0\r\n0,101,65,28,0,24.6,0.237,22,0\r\n5,137,108,0,0,48.8,0.
227,37,1\r\n2,110,74,29,125,32.4,0.698,27,0\r\n13,106,72,54,0,36.6,0.178,45,0\r\
n2,100,68,25,71,38.5,0.324,26,0\r\n15,136,70,32,110,37.1,0.153,43,1\r\n1,107,68,
19,0,26.5,0.165,24,0\r\n1,80,55,0,0,19.1,0.258,21,0\r\n4,123,80,15,176,32,0.443,
34,0\r\n7,81,78,40,48,46.7,0.261,42,0\r\n4,134,72,0,0,23.8,0.277,60,1\r\n2,142,8
2,18,64,24.7,0.761,21,0\r\n6,144,72,27,228,33.9,0.255,40,0\r\n2,92,62,28,0,31.6,
0.13,24,0\r\n1,71,48,18,76,20.4,0.323,22,0\r\n6,93,50,30,64,28.7,0.356,23,0\r\n1
,122,90,51,220,49.7,0.325,31,1\r\n1,163,72,0,0,39,1.222,33,1\r\n1,151,60,0,0,26.
1,0.179,22,0\r\n0,125,96,0,0,22.5,0.262,21,0\r\n1,81,72,18,40,26.6,0.283,24,0\r\
n2,85,65,0,0,39.6,0.93,27,0\r\n1,126,56,29,152,28.7,0.801,21,0\r\n1,96,122,0,0,2
2.4,0.207,27,0\r\n4,144,58,28,140,29.5,0.287,37,0\r\n3,83,58,31,18,34.3,0.336,25
,0\r\n0,95,85,25,36,37.4,0.247,24,1\r\n3,171,72,33,135,33.3,0.199,24,1\r\n8,155,
62,26,495,34,0.543,46,1\r\n1,89,76,34,37,31.2,0.192,23,0\r\n4,76,62,0,0,34,0.391
,25,0\r\n7,160,54,32,175,30.5,0.588,39,1\r\n4,146,92,0,0,31.2,0.539,61,1\r\n5,12
4,74,0,0,34,0.22,38,1\r\n5,78,48,0,0,33.7,0.654,25,0\r\n4,97,60,23,0,28.2,0.443,
22,0\r\n4,99,76,15,51,23.2,0.223,21,0\r\n0,162,76,56,100,53.2,0.759,25,1\r\n6,11
1,64,39,0,34.2,0.26,24,0\r\n2,107,74,30,100,33.6,0.404,23,0\r\n5,132,80,0,0,26.8
,0.186,69,0\r\n0,113,76,0,0,33.3,0.278,23,1\r\n1,88,30,42,99,55,0.496,26,1\r\n3,
120,70,30,135,42.9,0.452,30,0\r\n1,118,58,36,94,33.3,0.261,23,0\r\n1,117,88,24,1
45,34.5,0.403,40,1\r\n0,105,84,0,0,27.9,0.741,62,1\r\n4,173,70,14,168,29.7,0.361
,33,1\r\n9,122,56,0,0,33.3,1.114,33,1\r\n3,170,64,37,225,34.5,0.356,30,1\r\n8,84
,74,31,0,38.3,0.457,39,0\r\n2,96,68,13,49,21.1,0.647,26,0\r\n2,125,60,20,140,33.
8,0.088,31,0\r\n0,100,70,26,50,30.8,0.597,21,0\r\n0,93,60,25,92,28.7,0.532,22,0\
r\n0,129,80,0,0,31.2,0.703,29,0\r\n5,105,72,29,325,36.9,0.159,28,0\r\n3,128,78,0
,0,21.1,0.268,55,0\r\n5,106,82,30,0,39.5,0.286,38,0\r\n2,108,52,26,63,32.5,0.318
,22,0\r\n10,108,66,0,0,32.4,0.272,42,1\r\n4,154,62,31,284,32.8,0.237,23,0\r\n0,1
02,75,23,0,0,0.572,21,0\r\n9,57,80,37,0,32.8,0.096,41,0\r\n2,106,64,35,119,30.5,
1.4,34,0\r\n5,147,78,0,0,33.7,0.218,65,0\r\n2,90,70,17,0,27.3,0.085,22,0\r\n1,13
6,74,50,204,37.4,0.399,24,0\r\n4,114,65,0,0,21.9,0.432,37,0\r\n9,156,86,28,155,3
4.3,1.189,42,1\r\n1,153,82,42,485,40.6,0.687,23,0\r\n8,188,78,0,0,47.9,0.137,43,
1\r\n7,152,88,44,0,50,0.337,36,1\r\n2,99,52,15,94,24.6,0.637,21,0\r\n1,109,56,21
,135,25.2,0.833,23,0\r\n2,88,74,19,53,29,0.229,22,0\r\n17,163,72,41,114,40.9,0.8
17,47,1\r\n4,151,90,38,0,29.7,0.294,36,0\r\n7,102,74,40,105,37.2,0.204,45,0\r\n0
,114,80,34,285,44.2,0.167,27,0\r\n2,100,64,23,0,29.7,0.368,21,0\r\n0,131,88,0,0,
31.6,0.743,32,1\r\n6,104,74,18,156,29.9,0.722,41,1\r\n3,148,66,25,0,32.5,0.256,2
2,0\r\n4,120,68,0,0,29.6,0.709,34,0\r\n4,110,66,0,0,31.9,0.471,29,0\r\n3,111,90,
12,78,28.4,0.495,29,0\r\n6,102,82,0,0,30.8,0.18,36,1\r\n6,134,70,23,130,35.4,0.5
42,29,1\r\n2,87,0,23,0,28.9,0.773,25,0\r\n1,79,60,42,48,43.5,0.678,23,0\r\n2,75,
64,24,55,29.7,0.37,33,0\r\n8,179,72,42,130,32.7,0.719,36,1\r\n6,85,78,0,0,31.2,0
.382,42,0\r\n0,129,110,46,130,67.1,0.319,26,1\r\n5,143,78,0,0,45,0.19,47,0\r\n5,
130,82,0,0,39.1,0.956,37,1\r\n6,87,80,0,0,23.2,0.084,32,0\r\n0,119,64,18,92,34.9
,0.725,23,0\r\n1,0,74,20,23,27.7,0.299,21,0\r\n5,73,60,0,0,26.8,0.268,27,0\r\n4,
141,74,0,0,27.6,0.244,40,0\r\n7,194,68,28,0,35.9,0.745,41,1\r\n8,181,68,36,495,3
0.1,0.615,60,1\r\n1,128,98,41,58,32,1.321,33,1\r\n8,109,76,39,114,27.9,0.64,31,1
\r\n5,139,80,35,160,31.6,0.361,25,1\r\n3,111,62,0,0,22.6,0.142,21,0\r\n9,123,70,
44,94,33.1,0.374,40,0\r\n7,159,66,0,0,30.4,0.383,36,1\r\n11,135,0,0,0,52.3,0.578
,40,1\r\n8,85,55,20,0,24.4,0.136,42,0\r\n5,158,84,41,210,39.4,0.395,29,1\r\n1,10
5,58,0,0,24.3,0.187,21,0\r\n3,107,62,13,48,22.9,0.678,23,1\r\n4,109,64,44,99,34.
8,0.905,26,1\r\n4,148,60,27,318,30.9,0.15,29,1\r\n0,113,80,16,0,31,0.874,21,0\r\
n1,138,82,0,0,40.1,0.236,28,0\r\n0,108,68,20,0,27.3,0.787,32,0\r\n2,99,70,16,44,
20.4,0.235,27,0\r\n6,103,72,32,190,37.7,0.324,55,0\r\n5,111,72,28,0,23.9,0.407,2
7,0\r\n8,196,76,29,280,37.5,0.605,57,1\r\n5,162,104,0,0,37.7,0.151,52,1\r\n1,96,
64,27,87,33.2,0.289,21,0\r\n7,184,84,33,0,35.5,0.355,41,1\r\n2,81,60,22,0,27.7,0
```

```
.29,25,0\r\n0,147,85,54,0,42.8,0.375,24,0\r\n7,179,95,31,0,34.2,0.164,60,0\r\n0,
140,65,26,130,42.6,0.431,24,1\r\n9,112,82,32,175,34.2,0.26,36,1\r\n12,151,70,40,
271,41.8,0.742,38,1\r\n5,109,62,41,129,35.8,0.514,25,1\r\n6,125,68,30,120,30,0.4
64,32,0\r\n5,85,74,22,0,29,1.224,32,1\r\n5,112,66,0,0,37.8,0.261,41,1\r\n0,177,6
0,29,478,34.6,1.072,21,1\r\n2,158,90,0,0,31.6,0.805,66,1\r\n7,119,0,0,0,25.2,0.2
09,37,0\r\n7,142,60,33,190,28.8,0.687,61,0\r\n1,100,66,15,56,23.6,0.666,26,0\r\n
1,87,78,27,32,34.6,0.101,22,0\r\n0,101,76,0,0,35.7,0.198,26,0\r\n3,162,52,38,0,3
7.2,0.652,24,1\r\n4,197,70,39,744,36.7,2.329,31,0\r\n0,117,80,31,53,45.2,0.089,2
4,0\r\n4,142,86,0,0,44,0.645,22,1\r\n6,134,80,37,370,46.2,0.238,46,1\r\n1,79,80,
25,37,25.4,0.583,22,0\r\n4,122,68,0,0,35,0.394,29,0\r\n3,74,68,28,45,29.7,0.293,
23,0\r\n4,171,72,0,0,43.6,0.479,26,1\r\n7,181,84,21,192,35.9,0.586,51,1\r\n0,179
,90,27,0,44.1,0.686,23,1\r\n9,164,84,21,0,30.8,0.831,32,1\r\n0,104,76,0,0,18.4,0
.582,27,0\r\n1,91,64,24,0,29.2,0.192,21,0\r\n4,91,70,32,88,33.1,0.446,22,0\r\n3,
139,54,0,0,25.6,0.402,22,1\r\n6,119,50,22,176,27.1,1.318,33,1\r\n2,146,76,35,194
,38.2,0.329,29,0\r\n9,184,85,15,0,30,1.213,49,1\r\n10,122,68,0,0,31.2,0.258,41,0
\r\n0,165,90,33,680,52.3,0.427,23,0\r\n9,124,70,33,402,35.4,0.282,34,0\r\n1,111,
86,19,0,30.1,0.143,23,0\r\n9,106,52,0,0,31.2,0.38,42,0\r\n2,129,84,0,0,28,0.284,
27,0\r\n2,90,80,14,55,24.4,0.249,24,0\r\n0,86,68,32,0,35.8,0.238,25,0\r\n12,92,6
2,7,258,27.6,0.926,44,1\r\n1,113,64,35,0,33.6,0.543,21,1\r\n3,111,56,39,0,30.1,0
.557,30,0\r\n2,114,68,22,0,28.7,0.092,25,0\r\n1,193,50,16,375,25.9,0.655,24,0\r\
n11,155,76,28,150,33.3,1.353,51,1\r\n3,191,68,15,130,30.9,0.299,34,0\r\n3,141,0,
0,0,30,0.761,27,1\r\n4,95,70,32,0,32.1,0.612,24,0\r\n3,142,80,15,0,32.4,0.2,63,0
\r\n4,123,62,0,0,32,0.226,35,1\r\n5,96,74,18,67,33.6,0.997,43,0\r\n0,138,0,0,0,3
6.3,0.933,25,1\r\n2,128,64,42,0,40,1.101,24,0\r\n0,102,52,0,0,25.1,0.078,21,0\r\
n2,146,0,0,0,27.5,0.24,28,1\r\n10,101,86,37,0,45.6,1.136,38,1\r\n2,108,62,32,56,
25.2,0.128,21,0\r\n3,122,78,0,0,23,0.254,40,0\r\n1,71,78,50,45,33.2,0.422,21,0\r
\n13,106,70,0,0,34.2,0.251,52,0\r\n2,100,70,52,57,40.5,0.677,25,0\r\n7,106,60,24
,0,26.5,0.296,29,1\r\n0,104,64,23,116,27.8,0.454,23,0\r\n5,114,74,0,0,24.9,0.744
,57,0\r\n2,108,62,10,278,25.3,0.881,22,0\r\n0,146,70,0,0,37.9,0.334,28,1\r\n10,1
29,76,28,122,35.9,0.28,39,0\r\n7,133,88,15,155,32.4,0.262,37,0\r\n7,161,86,0,0,3
0.4,0.165,47,1\r\n2,108,80,0,0,27,0.259,52,1\r\n7,136,74,26,135,26,0.647,51,0\r\
n5,155,84,44,545,38.7,0.619,34,0\r\n1,119,86,39,220,45.6,0.808,29,1\r\n4,96,56,1
7,49,20.8,0.34,26,0\r\n5,108,72,43,75,36.1,0.263,33,0\r\n0,78,88,29,40,36.9,0.43
4,21,0\r\n0,107,62,30,74,36.6,0.757,25,1\r\n2,128,78,37,182,43.3,1.224,31,1\r\n1
,128,48,45,194,40.5,0.613,24,1\r\n0,161,50,0,0,21.9,0.254,65,0\r\n6,151,62,31,12
0,35.5,0.692,28,0\r\n2,146,70,38,360,28,0.337,29,1\r\n0,126,84,29,215,30.7,0.52,
24,0\r\n14,100,78,25,184,36.6,0.412,46,1\r\n8,112,72,0,0,23.6,0.84,58,0\r\n0,167
,0,0,0,32.3,0.839,30,1\r\n2,144,58,33,135,31.6,0.422,25,1\r\n5,77,82,41,42,35.8,
0.156,35,0\r\n5,115,98,0,0,52.9,0.209,28,1\r\n3,150,76,0,0,21,0.207,37,0\r\n2,12
0,76,37,105,39.7,0.215,29,0\r\n10,161,68,23,132,25.5,0.326,47,1\r\n0,137,68,14,1
48,24.8,0.143,21,0\r\n0,128,68,19,180,30.5,1.391,25,1\r\n2,124,68,28,205,32.9,0.
875,30,1\r\n6,80,66,30,0,26.2,0.313,41,0\r\n0,106,70,37,148,39.4,0.605,22,0\r\n2
,155,74,17,96,26.6,0.433,27,1\r\n3,113,50,10,85,29.5,0.626,25,0\r\n7,109,80,31,0
,35.9,1.127,43,1\r\n2,112,68,22,94,34.1,0.315,26,0\r\n3,99,80,11,64,19.3,0.284,3
0,0\r\n3,182,74,0,0,30.5,0.345,29,1\r\n3,115,66,39,140,38.1,0.15,28,0\r\n6,194,7
8,0,0,23.5,0.129,59,1\r\n4,129,60,12,231,27.5,0.527,31,0\r\n3,112,74,30,0,31.6,0
.197,25,1\r\n0,124,70,20,0,27.4,0.254,36,1\r\n13,152,90,33,29,26.8,0.731,43,1\r\
n2,112,75,32,0,35.7,0.148,21,0\r\n1,157,72,21,168,25.6,0.123,24,0\r\n1,122,64,32
,156,35.1,0.692,30,1\r\n10,179,70,0,0,35.1,0.2,37,0\r\n2,102,86,36,120,45.5,0.12
7,23,1\r\n6,105,70,32,68,30.8,0.122,37,0\r\n8,118,72,19,0,23.1,1.476,46,0\r\n2,8
7,58,16,52,32.7,0.166,25,0\r\n1,180,0,0,0,43.3,0.282,41,1\r\n12,106,80,0,0,23.6,
0.137,44,0\r\n1,95,60,18,58,23.9,0.26,22,0\r\n0,165,76,43,255,47.9,0.259,26,0\r\
n0,117,0,0,0,33.8,0.932,44,0\r\n5,115,76,0,0,31.2,0.343,44,1\r\n9,152,78,34,171,
34.2,0.893,33,1\r\n7,178,84,0,0,39.9,0.331,41,1\r\n1,130,70,13,105,25.9,0.472,22
,0\r\n1,95,74,21,73,25.9,0.673,36,0\r\n1,0,68,35,0,32,0.389,22,0\r\n5,122,86,0,0
,34.7,0.29,33,0\r\n8,95,72,0,0,36.8,0.485,57,0\r\n8,126,88,36,108,38.5,0.349,49,
0\r\n1,139,46,19,83,28.7,0.654,22,0\r\n3,116,0,0,0,23.5,0.187,23,0\r\n3,99,62,19
,74,21.8,0.279,26,0\r\n5,0,80,32,0,41,0.346,37,1\r\n4,92,80,0,0,42.2,0.237,29,0\
r\n4,137,84,0,0,31.2,0.252,30,0\r\n3,61,82,28,0,34.4,0.243,46,0\r\n1,90,62,12,43
,27.2,0.58,24,0\r\n3,90,78,0,0,42.7,0.559,21,0\r\n9,165,88,0,0,30.4,0.302,49,1\r
\n1,125,50,40,167,33.3,0.962,28,1\r\n13,129,0,30,0,39.9,0.569,44,1\r\n12,88,74,4
0,54,35.3,0.378,48,0\r\n1,196,76,36,249,36.5,0.875,29,1\r\n5,189,64,33,325,31.2,
0.583,29,1\r\n5,158,70,0,0,29.8,0.207,63,0\r\n5,103,108,37,0,39.2,0.305,65,0\r\n
4,146,78,0,0,38.5,0.52,67,1\r\n4,147,74,25,293,34.9,0.385,30,0\r\n5,99,54,28,83,
34,0.499,30,0\r\n6,124,72,0,0,27.6,0.368,29,1\r\n0,101,64,17,0,21,0.252,21,0\r\n
3,81,86,16,66,27.5,0.306,22,0\r\n1,133,102,28,140,32.8,0.234,45,1\r\n3,173,82,48
,465,38.4,2.137,25,1\r\n0,118,64,23,89,0,1.731,21,0\r\n0,84,64,22,66,35.8,0.545,
21,0\r\n2,105,58,40,94,34.9,0.225,25,0\r\n2,122,52,43,158,36.2,0.816,28,0\r\n12,
140,82,43,325,39.2,0.528,58,1\r\n0,98,82,15,84,25.2,0.299,22,0\r\n1,87,60,37,75,
37.2,0.509,22,0\r\n4,156,75,0,0,48.3,0.238,32,1\r\n0,93,100,39,72,43.4,1.021,35,
0\r\n1,107,72,30,82,30.8,0.821,24,0\r\n0,105,68,22,0,20,0.236,22,0\r\n1,109,60,8
,182,25.4,0.947,21,0\r\n1,90,62,18,59,25.1,1.268,25,0\r\n1,125,70,24,110,24.3,0.
221,25,0\r\n1,119,54,13,50,22.3,0.205,24,0\r\n5,116,74,29,0,32.3,0.66,35,1\r\n8,
105,100,36,0,43.3,0.239,45,1\r\n5,144,82,26,285,32,0.452,58,1\r\n3,100,68,23,81,
31.6,0.949,28,0\r\n1,100,66,29,196,32,0.444,42,0\r\n5,166,76,0,0,45.7,0.34,27,1\
r\n1,131,64,14,415,23.7,0.389,21,0\r\n4,116,72,12,87,22.1,0.463,37,0\r\n4,158,78
,0,0,32.9,0.803,31,1\r\n2,127,58,24,275,27.7,1.6,25,0\r\n3,96,56,34,115,24.7,0.9
44,39,0\r\n0,131,66,40,0,34.3,0.196,22,1\r\n3,82,70,0,0,21.1,0.389,25,0\r\n3,193
,70,31,0,34.9,0.241,25,1\r\n4,95,64,0,0,32,0.161,31,1\r\n6,137,61,0,0,24.2,0.151
,55,0\r\n5,136,84,41,88,35,0.286,35,1\r\n9,72,78,25,0,31.6,0.28,38,0\r\n5,168,64
,0,0,32.9,0.135,41,1\r\n2,123,48,32,165,42.1,0.52,26,0\r\n4,115,72,0,0,28.9,0.37
6,46,1\r\n0,101,62,0,0,21.9,0.336,25,0\r\n8,197,74,0,0,25.9,1.191,39,1\r\n1,172,
68,49,579,42.4,0.702,28,1\r\n6,102,90,39,0,35.7,0.674,28,0\r\n1,112,72,30,176,34
.4,0.528,25,0\r\n1,143,84,23,310,42.4,1.076,22,0\r\n1,143,74,22,61,26.2,0.256,21
,0\r\n0,138,60,35,167,34.6,0.534,21,1\r\n3,173,84,33,474,35.7,0.258,22,1\r\n1,97
,68,21,0,27.2,1.095,22,0\r\n4,144,82,32,0,38.5,0.554,37,1\r\n1,83,68,0,0,18.2,0.
624,27,0\r\n3,129,64,29,115,26.4,0.219,28,1\r\n1,119,88,41,170,45.3,0.507,26,0\r
\n2,94,68,18,76,26,0.561,21,0\r\n0,102,64,46,78,40.6,0.496,21,0\r\n2,115,64,22,0
,30.8,0.421,21,0\r\n8,151,78,32,210,42.9,0.516,36,1\r\n4,184,78,39,277,37,0.264,
31,1\r\n0,94,0,0,0,0,0.256,25,0\r\n1,181,64,30,180,34.1,0.328,38,1\r\n0,135,94,4
6,145,40.6,0.284,26,0\r\n1,95,82,25,180,35,0.233,43,1\r\n2,99,0,0,0,22.2,0.108,2
3,0\r\n3,89,74,16,85,30.4,0.551,38,0\r\n1,80,74,11,60,30,0.527,22,0\r\n2,139,75,
0,0,25.6,0.167,29,0\r\n1,90,68,8,0,24.5,1.138,36,0\r\n0,141,0,0,0,42.4,0.205,29,
1\r\n12,140,85,33,0,37.4,0.244,41,0\r\n5,147,75,0,0,29.9,0.434,28,0\r\n1,97,70,1
```

```
5,0,18.2,0.147,21,0\r\n6,107,88,0,0,36.8,0.727,31,0\r\n0,189,104,25,0,34.3,0.435
,41,1\r\n2,83,66,23,50,32.2,0.497,22,0\r\n4,117,64,27,120,33.2,0.23,24,0\r\n8,10
8,70,0,0,30.5,0.955,33,1\r\n4,117,62,12,0,29.7,0.38,30,1\r\n0,180,78,63,14,59.4,
2.42,25,1\r\n1,100,72,12,70,25.3,0.658,28,0\r\n0,95,80,45,92,36.5,0.33,26,0\r\n0
,104,64,37,64,33.6,0.51,22,1\r\n0,120,74,18,63,30.5,0.285,26,0\r\n1,82,64,13,95,
21.2,0.415,23,0\r\n2,134,70,0,0,28.9,0.542,23,1\r\n0,91,68,32,210,39.9,0.381,25,
0\r\n2,119,0,0,0,19.6,0.832,72,0\r\n2,100,54,28,105,37.8,0.498,24,0\r\n14,175,62
,30,0,33.6,0.212,38,1\r\n1,135,54,0,0,26.7,0.687,62,0\r\n5,86,68,28,71,30.2,0.36
4,24,0\r\n10,148,84,48,237,37.6,1.001,51,1\r\n9,134,74,33,60,25.9,0.46,81,0\r\n9
,120,72,22,56,20.8,0.733,48,0\r\n1,71,62,0,0,21.8,0.416,26,0\r\n8,74,70,40,49,35
.3,0.705,39,0\r\n5,88,78,30,0,27.6,0.258,37,0\r\n10,115,98,0,0,24,1.022,34,0\r\n
0,124,56,13,105,21.8,0.452,21,0\r\n0,74,52,10,36,27.8,0.269,22,0\r\n0,97,64,36,1
00,36.8,0.6,25,0\r\n8,120,0,0,0,30,0.183,38,1\r\n6,154,78,41,140,46.1,0.571,27,0
\r\n1,144,82,40,0,41.3,0.607,28,0\r\n0,137,70,38,0,33.2,0.17,22,0\r\n0,119,66,27
,0,38.8,0.259,22,0\r\n7,136,90,0,0,29.9,0.21,50,0\r\n4,114,64,0,0,28.9,0.126,24,
0\r\n0,137,84,27,0,27.3,0.231,59,0\r\n2,105,80,45,191,33.7,0.711,29,1\r\n7,114,7
6,17,110,23.8,0.466,31,0\r\n8,126,74,38,75,25.9,0.162,39,0\r\n4,132,86,31,0,28,0
.419,63,0\r\n3,158,70,30,328,35.5,0.344,35,1\r\n0,123,88,37,0,35.2,0.197,29,0\r\
n4,85,58,22,49,27.8,0.306,28,0\r\n0,84,82,31,125,38.2,0.233,23,0\r\n0,145,0,0,0,
44.2,0.63,31,1\r\n0,135,68,42,250,42.3,0.365,24,1\r\n1,139,62,41,480,40.7,0.536,
21,0\r\n0,173,78,32,265,46.5,1.159,58,0\r\n4,99,72,17,0,25.6,0.294,28,0\r\n8,194
,80,0,0,26.1,0.551,67,0\r\n2,83,65,28,66,36.8,0.629,24,0\r\n2,89,90,30,0,33.5,0.
292,42,0\r\n4,99,68,38,0,32.8,0.145,33,0\r\n4,125,70,18,122,28.9,1.144,45,1\r\n3
,80,0,0,0,0,0.174,22,0\r\n6,166,74,0,0,26.6,0.304,66,0\r\n5,110,68,0,0,26,0.292,
30,0\r\n2,81,72,15,76,30.1,0.547,25,0\r\n7,195,70,33,145,25.1,0.163,55,1\r\n6,15
4,74,32,193,29.3,0.839,39,0\r\n2,117,90,19,71,25.2,0.313,21,0\r\n3,84,72,32,0,37
.2,0.267,28,0\r\n6,0,68,41,0,39,0.727,41,1\r\n7,94,64,25,79,33.3,0.738,41,0\r\n3
,96,78,39,0,37.3,0.238,40,0\r\n10,75,82,0,0,33.3,0.263,38,0\r\n0,180,90,26,90,36
.5,0.314,35,1\r\n1,130,60,23,170,28.6,0.692,21,0\r\n2,84,50,23,76,30.4,0.968,21,
0\r\n8,120,78,0,0,25,0.409,64,0\r\n12,84,72,31,0,29.7,0.297,46,1\r\n0,139,62,17,
210,22.1,0.207,21,0\r\n9,91,68,0,0,24.2,0.2,58,0\r\n2,91,62,0,0,27.3,0.525,22,0\
r\n3,99,54,19,86,25.6,0.154,24,0\r\n3,163,70,18,105,31.6,0.268,28,1\r\n9,145,88,
34,165,30.3,0.771,53,1\r\n7,125,86,0,0,37.6,0.304,51,0\r\n13,76,60,0,0,32.8,0.18
,41,0\r\n6,129,90,7,326,19.6,0.582,60,0\r\n2,68,70,32,66,25,0.187,25,0\r\n3,124,
80,33,130,33.2,0.305,26,0\r\n6,114,0,0,0,0,0.189,26,0\r\n9,130,70,0,0,34.2,0.652
,45,1\r\n3,125,58,0,0,31.6,0.151,24,0\r\n3,87,60,18,0,21.8,0.444,21,0\r\n1,97,64
,19,82,18.2,0.299,21,0\r\n3,116,74,15,105,26.3,0.107,24,0\r\n0,117,66,31,188,30.
8,0.493,22,0\r\n0,111,65,0,0,24.6,0.66,31,0\r\n2,122,60,18,106,29.8,0.717,22,0\r
\n0,107,76,0,0,45.3,0.686,24,0\r\n1,86,66,52,65,41.3,0.917,29,0\r\n6,91,0,0,0,29
.8,0.501,31,0\r\n1,77,56,30,56,33.3,1.251,24,0\r\n4,132,0,0,0,32.9,0.302,23,1\r\
n0,105,90,0,0,29.6,0.197,46,0\r\n0,57,60,0,0,21.7,0.735,67,0\r\n0,127,80,37,210,
36.3,0.804,23,0\r\n3,129,92,49,155,36.4,0.968,32,1\r\n8,100,74,40,215,39.4,0.661
,43,1\r\n3,128,72,25,190,32.4,0.549,27,1\r\n10,90,85,32,0,34.9,0.825,56,1\r\n4,8
4,90,23,56,39.5,0.159,25,0\r\n1,88,78,29,76,32,0.365,29,0\r\n8,186,90,35,225,34.
5,0.423,37,1\r\n5,187,76,27,207,43.6,1.034,53,1\r\n4,131,68,21,166,33.1,0.16,28,
0\r\n1,164,82,43,67,32.8,0.341,50,0\r\n4,189,110,31,0,28.5,0.68,37,0\r\n1,116,70
,28,0,27.4,0.204,21,0\r\n3,84,68,30,106,31.9,0.591,25,0\r\n6,114,88,0,0,27.8,0.2
```

```
47,66,0\r\n1,88,62,24,44,29.9,0.422,23,0\r\n1,84,64,23,115,36.9,0.471,28,0\r\n7,
124,70,33,215,25.5,0.161,37,0\r\n1,97,70,40,0,38.1,0.218,30,0\r\n8,110,76,0,0,27
.8,0.237,58,0\r\n11,103,68,40,0,46.2,0.126,42,0\r\n11,85,74,0,0,30.1,0.3,35,0\r\
n6,125,76,0,0,33.8,0.121,54,1\r\n0,198,66,32,274,41.3,0.502,28,1\r\n1,87,68,34,7
7,37.6,0.401,24,0\r\n6,99,60,19,54,26.9,0.497,32,0\r\n0,91,80,0,0,32.4,0.601,27,
0\r\n2,95,54,14,88,26.1,0.748,22,0\r\n1,99,72,30,18,38.6,0.412,21,0\r\n6,92,62,3
2,126,32,0.085,46,0\r\n4,154,72,29,126,31.3,0.338,37,0\r\n0,121,66,30,165,34.3,0
.203,33,1\r\n3,78,70,0,0,32.5,0.27,39,0\r\n2,130,96,0,0,22.6,0.268,21,0\r\n3,111
,58,31,44,29.5,0.43,22,0\r\n2,98,60,17,120,34.7,0.198,22,0\r\n1,143,86,30,330,30
.1,0.892,23,0\r\n1,119,44,47,63,35.5,0.28,25,0\r\n6,108,44,20,130,24,0.813,35,0\
r\n2,118,80,0,0,42.9,0.693,21,1\r\n10,133,68,0,0,27,0.245,36,0\r\n2,197,70,99,0,
34.7,0.575,62,1\r\n0,151,90,46,0,42.1,0.371,21,1\r\n6,109,60,27,0,25,0.206,27,0\
r\n12,121,78,17,0,26.5,0.259,62,0\r\n8,100,76,0,0,38.7,0.19,42,0\r\n8,124,76,24,
600,28.7,0.687,52,1\r\n1,93,56,11,0,22.5,0.417,22,0\r\n8,143,66,0,0,34.9,0.129,4
1,1\r\n6,103,66,0,0,24.3,0.249,29,0\r\n3,176,86,27,156,33.3,1.154,52,1\r\n0,73,0
,0,0,21.1,0.342,25,0\r\n11,111,84,40,0,46.8,0.925,45,1\r\n2,112,78,50,140,39.4,0
.175,24,0\r\n3,132,80,0,0,34.4,0.402,44,1\r\n2,82,52,22,115,28.5,1.699,25,0\r\n6
,123,72,45,230,33.6,0.733,34,0\r\n0,188,82,14,185,32,0.682,22,1\r\n0,67,76,0,0,4
5.3,0.194,46,0\r\n1,89,24,19,25,27.8,0.559,21,0\r\n1,173,74,0,0,36.8,0.088,38,1\
r\n1,109,38,18,120,23.1,0.407,26,0\r\n1,108,88,19,0,27.1,0.4,24,0\r\n6,96,0,0,0,
23.7,0.19,28,0\r\n1,124,74,36,0,27.8,0.1,30,0\r\n7,150,78,29,126,35.2,0.692,54,1
\r\n4,183,0,0,0,28.4,0.212,36,1\r\n1,124,60,32,0,35.8,0.514,21,0\r\n1,181,78,42,
293,40,1.258,22,1\r\n1,92,62,25,41,19.5,0.482,25,0\r\n0,152,82,39,272,41.5,0.27,
27,0\r\n1,111,62,13,182,24,0.138,23,0\r\n3,106,54,21,158,30.9,0.292,24,0\r\n3,17
4,58,22,194,32.9,0.593,36,1\r\n7,168,88,42,321,38.2,0.787,40,1\r\n6,105,80,28,0,
32.5,0.878,26,0\r\n11,138,74,26,144,36.1,0.557,50,1\r\n3,106,72,0,0,25.8,0.207,2
7,0\r\n6,117,96,0,0,28.7,0.157,30,0\r\n2,68,62,13,15,20.1,0.257,23,0\r\n9,112,82
,24,0,28.2,1.282,50,1\r\n0,119,0,0,0,32.4,0.141,24,1\r\n2,112,86,42,160,38.4,0.2
46,28,0\r\n2,92,76,20,0,24.2,1.698,28,0\r\n6,183,94,0,0,40.8,1.461,45,0\r\n0,94,
70,27,115,43.5,0.347,21,0\r\n2,108,64,0,0,30.8,0.158,21,0\r\n4,90,88,47,54,37.7,
0.362,29,0\r\n0,125,68,0,0,24.7,0.206,21,0\r\n0,132,78,0,0,32.4,0.393,21,0\r\n5,
128,80,0,0,34.6,0.144,45,0\r\n4,94,65,22,0,24.7,0.148,21,0\r\n7,114,64,0,0,27.4,
0.732,34,1\r\n0,102,78,40,90,34.5,0.238,24,0\r\n2,111,60,0,0,26.2,0.343,23,0\r\n
1,128,82,17,183,27.5,0.115,22,0\r\n10,92,62,0,0,25.9,0.167,31,0\r\n13,104,72,0,0
,31.2,0.465,38,1\r\n5,104,74,0,0,28.8,0.153,48,0\r\n2,94,76,18,66,31.6,0.649,23,
0\r\n7,97,76,32,91,40.9,0.871,32,1\r\n1,100,74,12,46,19.5,0.149,28,0\r\n0,102,86
,17,105,29.3,0.695,27,0\r\n4,128,70,0,0,34.3,0.303,24,0\r\n6,147,80,0,0,29.5,0.1
78,50,1\r\n4,90,0,0,0,28,0.61,31,0\r\n3,103,72,30,152,27.6,0.73,27,0\r\n2,157,74
,35,440,39.4,0.134,30,0\r\n1,167,74,17,144,23.4,0.447,33,1\r\n0,179,50,36,159,37
.8,0.455,22,1\r\n11,136,84,35,130,28.3,0.26,42,1\r\n0,107,60,25,0,26.4,0.133,23,
0\r\n1,91,54,25,100,25.2,0.234,23,0\r\n1,117,60,23,106,33.8,0.466,27,0\r\n5,123,
74,40,77,34.1,0.269,28,0\r\n2,120,54,0,0,26.8,0.455,27,0\r\n1,106,70,28,135,34.2
,0.142,22,0\r\n2,155,52,27,540,38.7,0.24,25,1\r\n2,101,58,35,90,21.8,0.155,22,0\
r\n1,120,80,48,200,38.9,1.162,41,0\r\n11,127,106,0,0,39,0.19,51,0\r\n3,80,82,31,
70,34.2,1.292,27,1\r\n10,162,84,0,0,27.7,0.182,54,0\r\n1,199,76,43,0,42.9,1.394,
22,1\r\n8,167,106,46,231,37.6,0.165,43,1\r\n9,145,80,46,130,37.9,0.637,40,1\r\n6
,115,60,39,0,33.7,0.245,40,1\r\n1,112,80,45,132,34.8,0.217,24,0\r\n4,145,82,18,0
```

```
,32.5,0.235,70,1\r\n10,111,70,27,0,27.5,0.141,40,1\r\n6,98,58,33,190,34,0.43,43,
0\r\n9,154,78,30,100,30.9,0.164,45,0\r\n6,165,68,26,168,33.6,0.631,49,0\r\n1,99,
58,10,0,25.4,0.551,21,0\r\n10,68,106,23,49,35.5,0.285,47,0\r\n3,123,100,35,240,5
7.3,0.88,22,0\r\n8,91,82,0,0,35.6,0.587,68,0\r\n6,195,70,0,0,30.9,0.328,31,1\r\n
9,156,86,0,0,24.8,0.23,53,1\r\n0,93,60,0,0,35.3,0.263,25,0\r\n3,121,52,0,0,36,0.
127,25,1\r\n2,101,58,17,265,24.2,0.614,23,0\r\n2,56,56,28,45,24.2,0.332,22,0\r\n
0,162,76,36,0,49.6,0.364,26,1\r\n0,95,64,39,105,44.6,0.366,22,0\r\n4,125,80,0,0,
32.3,0.536,27,1\r\n5,136,82,0,0,0,0.64,69,0\r\n2,129,74,26,205,33.2,0.591,25,0\r
\n3,130,64,0,0,23.1,0.314,22,0\r\n1,107,50,19,0,28.3,0.181,29,0\r\n1,140,74,26,1
80,24.1,0.828,23,0\r\n1,144,82,46,180,46.1,0.335,46,1\r\n8,107,80,0,0,24.6,0.856
,34,0\r\n13,158,114,0,0,42.3,0.257,44,1\r\n2,121,70,32,95,39.1,0.886,23,0\r\n7,1
29,68,49,125,38.5,0.439,43,1\r\n2,90,60,0,0,23.5,0.191,25,0\r\n7,142,90,24,480,3
0.4,0.128,43,1\r\n3,169,74,19,125,29.9,0.268,31,1\r\n0,99,0,0,0,25,0.253,22,0\r\
n4,127,88,11,155,34.5,0.598,28,0\r\n4,118,70,0,0,44.5,0.904,26,0\r\n2,122,76,27,
200,35.9,0.483,26,0\r\n6,125,78,31,0,27.6,0.565,49,1\r\n1,168,88,29,0,35,0.905,5
2,1\r\n2,129,0,0,0,38.5,0.304,41,0\r\n4,110,76,20,100,28.4,0.118,27,0\r\n6,80,80
,36,0,39.8,0.177,28,0\r\n10,115,0,0,0,0,0.261,30,1\r\n2,127,46,21,335,34.4,0.176
,22,0\r\n9,164,78,0,0,32.8,0.148,45,1\r\n2,93,64,32,160,38,0.674,23,1\r\n3,158,6
4,13,387,31.2,0.295,24,0\r\n5,126,78,27,22,29.6,0.439,40,0\r\n10,129,62,36,0,41.
2,0.441,38,1\r\n0,134,58,20,291,26.4,0.352,21,0\r\n3,102,74,0,0,29.5,0.121,32,0\
r\n7,187,50,33,392,33.9,0.826,34,1\r\n3,173,78,39,185,33.8,0.97,31,1\r\n10,94,72
,18,0,23.1,0.595,56,0\r\n1,108,60,46,178,35.5,0.415,24,0\r\n5,97,76,27,0,35.6,0.
378,52,1\r\n4,83,86,19,0,29.3,0.317,34,0\r\n1,114,66,36,200,38.1,0.289,21,0\r\n1
,149,68,29,127,29.3,0.349,42,1\r\n5,117,86,30,105,39.1,0.251,42,0\r\n1,111,94,0,
0,32.8,0.265,45,0\r\n4,112,78,40,0,39.4,0.236,38,0\r\n1,116,78,29,180,36.1,0.496
,25,0\r\n0,141,84,26,0,32.4,0.433,22,0\r\n2,175,88,0,0,22.9,0.326,22,0\r\n2,92,5
2,0,0,30.1,0.141,22,0\r\n3,130,78,23,79,28.4,0.323,34,1\r\n8,120,86,0,0,28.4,0.2
59,22,1\r\n2,174,88,37,120,44.5,0.646,24,1\r\n2,106,56,27,165,29,0.426,22,0\r\n2
,105,75,0,0,23.3,0.56,53,0\r\n4,95,60,32,0,35.4,0.284,28,0\r\n0,126,86,27,120,27
.4,0.515,21,0\r\n8,65,72,23,0,32,0.6,42,0\r\n2,99,60,17,160,36.6,0.453,21,0\r\n1
,102,74,0,0,39.5,0.293,42,1\r\n11,120,80,37,150,42.3,0.785,48,1\r\n3,102,44,20,9
4,30.8,0.4,26,0\r\n1,109,58,18,116,28.5,0.219,22,0\r\n9,140,94,0,0,32.7,0.734,45
,1\r\n13,153,88,37,140,40.6,1.174,39,0\r\n12,100,84,33,105,30,0.488,46,0\r\n1,14
7,94,41,0,49.3,0.358,27,1\r\n1,81,74,41,57,46.3,1.096,32,0\r\n3,187,70,22,200,36
.4,0.408,36,1\r\n6,162,62,0,0,24.3,0.178,50,1\r\n4,136,70,0,0,31.2,1.182,22,1\r\
n1,121,78,39,74,39,0.261,28,0\r\n3,108,62,24,0,26,0.223,25,0\r\n0,181,88,44,510,
43.3,0.222,26,1\r\n8,154,78,32,0,32.4,0.443,45,1\r\n1,128,88,39,110,36.5,1.057,3
7,1\r\n7,137,90,41,0,32,0.391,39,0\r\n0,123,72,0,0,36.3,0.258,52,1\r\n1,106,76,0
,0,37.5,0.197,26,0\r\n6,190,92,0,0,35.5,0.278,66,1\r\n2,88,58,26,16,28.4,0.766,2
2,0\r\n9,170,74,31,0,44,0.403,43,1\r\n9,89,62,0,0,22.5,0.142,33,0\r\n10,101,76,4
8,180,32.9,0.171,63,0\r\n2,122,70,27,0,36.8,0.34,27,0\r\n5,121,72,23,112,26.2,0.
245,30,0\r\n1,126,60,0,0,30.1,0.349,47,1\r\n1,93,70,31,0,30.4,0.315,23,0'}
```

```
[62]: !ls
```

```
'diabetes (1).csv'   sample_data                  spark-3.1.1-bin-hadoop3.2.tgz
 diabetes.csv        spark-3.1.1-bin-hadoop3.2   spark-3.1.1-bin-hadoop3.2.tgz.1
```

```
[63]: spark = SparkSession.builder.master("local[*]").getOrCreate()
      dataset = spark.read.csv('diabetes.csv',inferSchema=True, header =True)
```

```
[64]: dataset.show(10)
```

```
+-----------+-------+-------------+-------------+-------+----+------------------
------+---+-------+
|Pregnancies|Glucose|BloodPressure|SkinThickness|Insulin|
BMI|DiabetesPedigreeFunction|Age|Outcome|
+-----------+-------+-------------+-------------+-------+----+------------------
------+---+-------+
|          6|    148|           72|           35|      0|33.6|
0.627| 50|      1|
|          1|     85|           66|           29|      0|26.6|
0.351| 31|      0|
|          8|    183|           64|            0|      0|23.3|
0.672| 32|      1|
|          1|     89|           66|           23|     94|28.1|
0.167| 21|      0|
|          0|    137|           40|           35|    168|43.1|
2.288| 33|      1|
|          5|    116|           74|            0|      0|25.6|
0.201| 30|      0|
|          3|     78|           50|           32|     88|31.0|
0.248| 26|      1|
|         10|    115|            0|            0|      0|35.3|
0.134| 29|      0|
|          2|    197|           70|           45|    543|30.5|
0.158| 53|      1|
|          8|    125|           96|            0|      0| 0.0|
0.232| 54|      1|
+-----------+-------+-------------+-------------+-------+----+------------------
------+---+-------+
only showing top 10 rows
```

1.3 Préparation des données

1.3.1 Types de variables

```
[65]: dataset.dtypes
```

```
[65]: [('Pregnancies', 'int'),
       ('Glucose', 'int'),
       ('BloodPressure', 'int'),
       ('SkinThickness', 'int'),
       ('Insulin', 'int'),
       ('BMI', 'double'),
       ('DiabetesPedigreeFunction', 'double'),
       ('Age', 'int'),
       ('Outcome', 'int')]
```

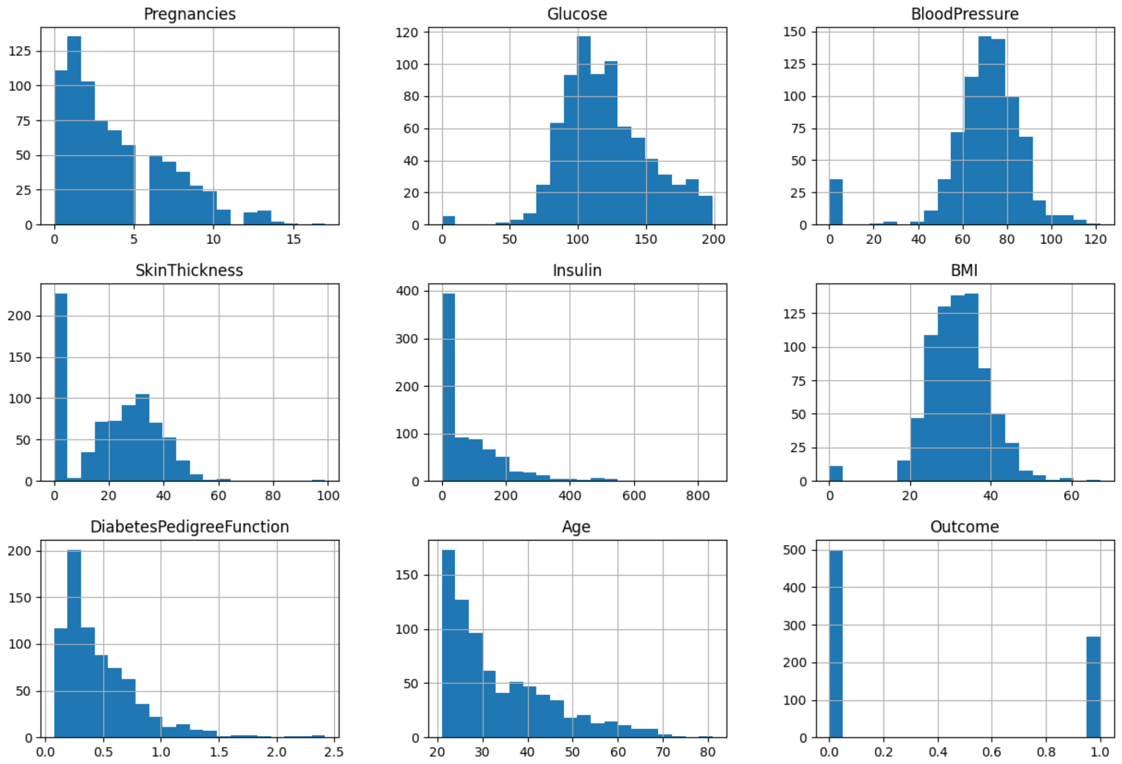
Le type des variables semble être adapté pour la classification.

1.3.2 Recherche des outliers

On va afficher les données sous forme d'histogramme pour rechercher les outliers.

```
[66]: numeric_cols = [col[0] for col in dataset.dtypes if col[1] in ['int', 'double']]
      dataset.select(numeric_cols).toPandas().hist(bins=20, figsize=(15, 10))
```

```
[66]: array([[<Axes: title={'center': 'Pregnancies'}>,
              <Axes: title={'center': 'Glucose'}>,
              <Axes: title={'center': 'BloodPressure'}>],
             [<Axes: title={'center': 'SkinThickness'}>,
              <Axes: title={'center': 'Insulin'}>,
              <Axes: title={'center': 'BMI'}>],
             [<Axes: title={'center': 'DiabetesPedigreeFunction'}>,
              <Axes: title={'center': 'Age'}>,
              <Axes: title={'center': 'Outcome'}>]], dtype=object)
```

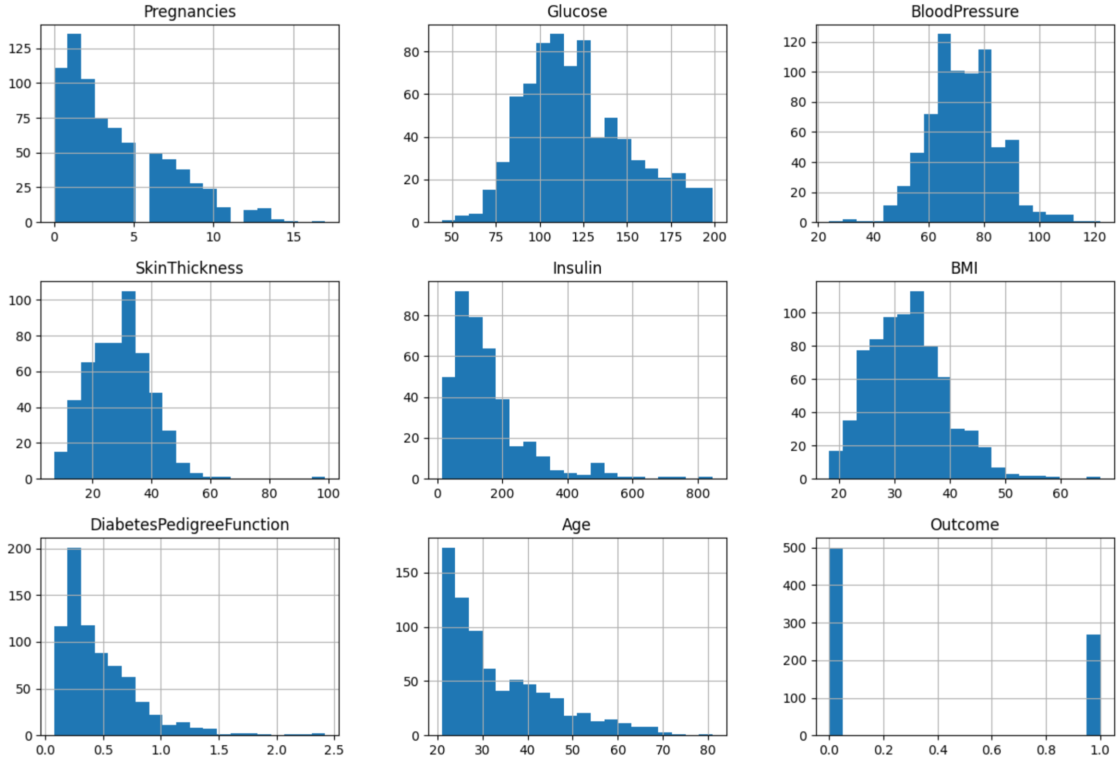


On remarque de nombreuses valeurs en dehors des distributions normales que suivent globalement les courbes pour certaines mesures. On en déduit que des valeurs à 0 signifient que la mesure n'a pas été effectuée. On va donc remplacer ces valeurs par null pour s'assurer qui rend plus transparent l'abscence de mesure.

```
[67]: for column_name in ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',␣
       ↪'BMI']:
        dataset =  dataset.withColumn(column_name, f.when(f.col(column_name) == 0,␣
       ↪None).otherwise(f.col(column_name)))
```

```
[68]: numeric_cols = [col[0] for col in dataset.dtypes if col[1] in ['int', 'double']]
      dataset.select(numeric_cols).toPandas().hist(bins=20, figsize=(15, 10))
```

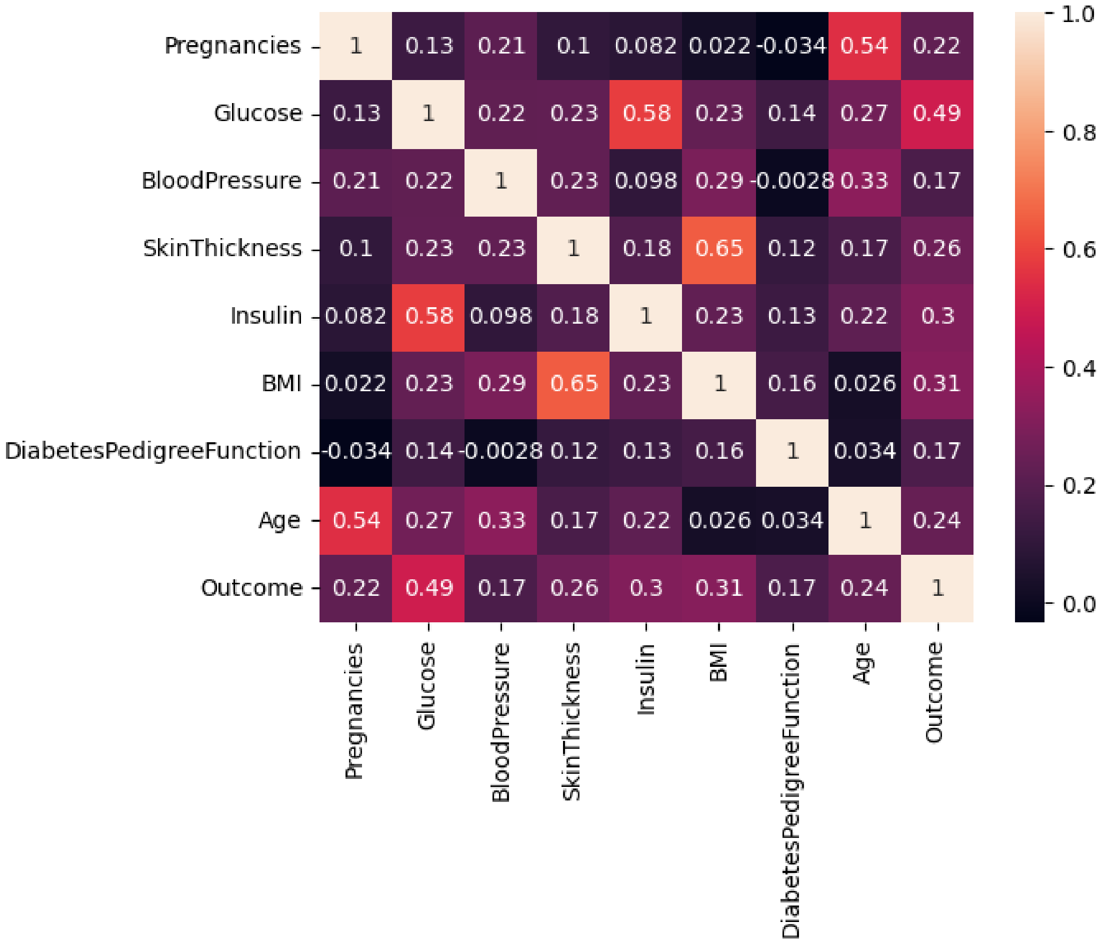
```
[68]: array([[<Axes: title={'center': 'Pregnancies'}>,
              <Axes: title={'center': 'Glucose'}>,
              <Axes: title={'center': 'BloodPressure'}>],
             [<Axes: title={'center': 'SkinThickness'}>,
              <Axes: title={'center': 'Insulin'}>,
              <Axes: title={'center': 'BMI'}>],
             [<Axes: title={'center': 'DiabetesPedigreeFunction'}>,
              <Axes: title={'center': 'Age'}>,
              <Axes: title={'center': 'Outcome'}>]], dtype=object)
```



1.3.3 Recherche des corrélations

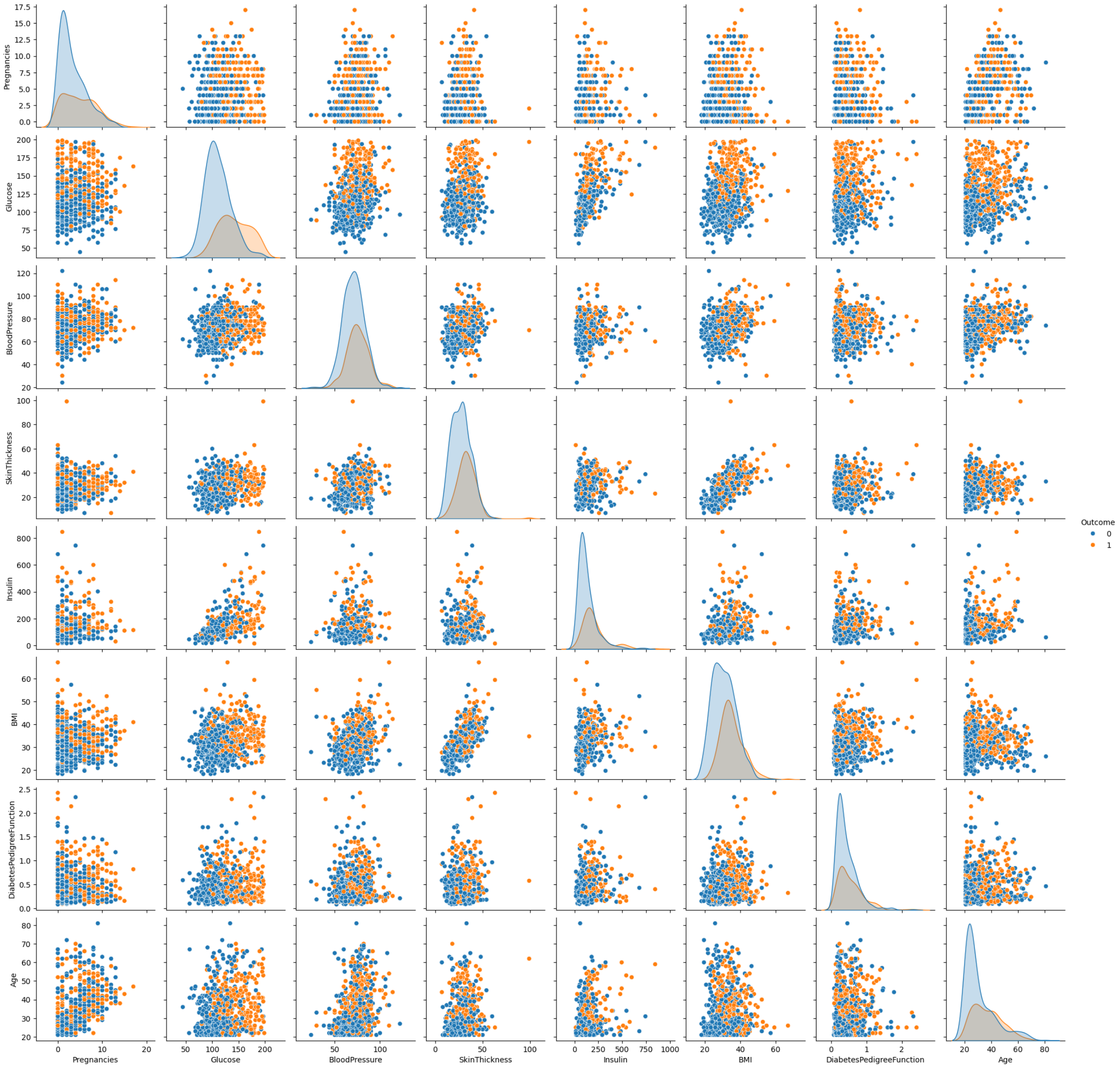
```
[69]: sns.heatmap(dataset.toPandas().corr(), annot = True)
```

```
[69]: <Axes: >
```



```
[70]: sns.pairplot(dataset.select(numeric_cols).toPandas(), hue = 'Outcome')
```

```
[70]: <seaborn.axisgrid.PairGrid at 0x7e5f9d523e80>
```



On remarque une forte corrélation entre le Glucose et l'Outcome, et entre l'âge et le nombre de grossesses. On va donc chercher quelle variable a été le moins remplie pour savoir laquelle on supprime.

[71]:
```
dataset.toPandas().isna().sum()
```

[71]:
```
Pregnancies                   0
Glucose                       5
BloodPressure                35
SkinThickness               227
Insulin                     374
BMI                          11
DiabetesPedigreeFunction      0
Age                           0
Outcome                       0
```

```
dtype: int64
```

On remarque que le SkinThickness n'est pas rempli pour une grande partie de la base de données. Ce sera donc cette variable que l'on va supprimer. On supprimera également grossesses qui a l'air d'avoir un impact plus faible dans les distributions affichées par seaborn au-dessus.

```
[72]: dataset = dataset.drop('SkinThickness').drop('Pregnancies')
```

1.3.4 Gestion des null

```
[73]: 374 / dataset.count()
```

```
[73]: 0.4869791666666667
```

50% de la base n'a pas les données d'Insuline, on enlévera donc également cette variable.

```
[74]: dataset = dataset.drop('Insulin')
```

```
[75]: dataset.toPandas().isna().sum()
```

```
[75]: Glucose                      5
      BloodPressure               35
      BMI                         11
      DiabetesPedigreeFunction     0
      Age                          0
      Outcome                      0
      dtype: int64
```

On a encore quelques individus qui n'ont pas effectué certaines des mesures, et que l'on va retirer pour uniformiser notre base de données.

```
[76]: dataset = dataset.dropna()
```

```
[77]: dataset.toPandas().isna().sum()
```

```
[77]: Glucose                     0
      BloodPressure               0
      BMI                         0
      DiabetesPedigreeFunction    0
      Age                         0
      Outcome                     0
      dtype: int64
```

En dehors de cela, les courbes semblent suivre des lois normales qui est probablement le résultat attendue: on en conclut qu'aucune autre modification n'est nécessaire pour supprimer les individus anormaux.

1.4 Séparation des datasets

On sépare le dataset en 80% pour l'entraînement, et 20% pour le test.

[78]:
```
dataset_train, dataset_test = dataset.randomSplit([0.8, 0.2])
```

1.5 Standardisation des données

On normalise les données pour éviter l'impact d'une différence d'ordre de grandeur.

[79]:
```
features = ['Glucose', 'BloodPressure', 'BMI', 'DiabetesPedigreeFunction',␣
 ↪'Age']
assembler = VectorAssembler(inputCols = features, outputCol = 'features')
dataset_assembled = assembler.transform(dataset)
train_assembled = assembler.transform(dataset_train)

standardScaler = StandardScaler(inputCol='features',␣
 ↪outputCol='scaled_features')
scaler_model = standardScaler.fit(dataset_assembled)
train_norm = scaler_model.transform(train_assembled)
```

[80]:
```
train_norm.show()
```

```
+-------+-------------+----+------------------------+---+-------+---------------
-----+--------------------+
|Glucose|BloodPressure| BMI|DiabetesPedigreeFunction|Age|Outcome|
features|     scaled_features|
+-------+-------------+----+------------------------+---+-------+---------------
-----+--------------------+
|     44|           62|25.0|                   0.587| 36|
0|[44.0,62.0,25.0,0…|[1.43089291370079…|
|     56|           56|24.2|                   0.332| 22|
0|[56.0,56.0,24.2,0…|[1.82113643561919…|
|     57|           60|21.7|                   0.735| 67|
0|[57.0,60.0,21.7,0…|[1.85365672911239…|
|     62|           78|32.6|                   0.391| 41|
0|[62.0,78.0,32.6,0…|[2.01625819657839…|
|     65|           72|32.0|                     0.6| 42|
0|[65.0,72.0,32.0,0…|[2.11381907705799…|
|     67|           76|45.3|                   0.194| 46|
0|[67.0,76.0,45.3,0…|[2.17885966404439…|
|     68|           70|25.0|                   0.187| 25|
0|[68.0,70.0,25.0,0…|[2.21137995753759…|
|     71|           62|21.8|                   0.416| 26|
0|[71.0,62.0,21.8,0…|[2.30894083801719…|
|     71|           70|28.0|                   0.586| 22|
0|[71.0,70.0,28.0,0…|[2.30894083801719…|
|     71|           78|33.2|                   0.422| 21|
0|[71.0,78.0,33.2,0…|[2.30894083801719…|
|     72|            78|31.6|                    0.28| 38|
0|[72.0,78.0,31.6,0…|[2.34146113151039…|
|     73|            50|23.0|                   0.248| 21|
0|[73.0,50.0,23.0,0…|[2.37398142500359…|
|     73|            60|26.8|                   0.268| 27|
0|[73.0,60.0,26.8,0…|[2.37398142500359…|
|     74|            52|27.8|                   0.269| 22|
0|[74.0,52.0,27.8,0…|[2.40650171849679…|
|     74|            68|29.7|                   0.293| 23|
0|[74.0,68.0,29.7,0…|[2.40650171849679…|
|     75|            64|29.7|                    0.37| 33|
0|[75.0,64.0,29.7,0…|[2.43902201198999…|
|     75|            82|33.3|                   0.263| 38|
0|[75.0,82.0,33.3,0…|[2.43902201198999…|
|     76|            60|32.8|                    0.18| 41|
0|[76.0,60.0,32.8,0…|[2.47154230548319…|
|     76|            62|34.0|                   0.391| 25|
0|[76.0,62.0,34.0,0…|[2.47154230548319…|
|     77|            56|33.3|                   1.251| 24|
0|[77.0,56.0,33.3,1…|[2.50406259897639…|
+-------+-------------+----+------------------------+---+-------+---------------
-----+--------------------+
only showing top 20 rows
```

1.6 Définition des modèles

```
[81]: evaluator = BinaryClassificationEvaluator(labelCol = 'Outcome')
```

1.6.1 Arbre de décision

Commençons par créer un arbre de décision.

```
[82]: tree = DecisionTreeClassifier(labelCol = 'Outcome', featuresCol =␣
       ↪'scaled_features')
      pipeline_tree = Pipeline(stages = [assembler, standardScaler, tree])
      params_tree = (ParamGridBuilder()
          .addGrid(tree.maxDepth, [5, 10, 15])
          .addGrid(tree.impurity, ['gini', 'entropy'])
          .addGrid(tree.maxBins, [32, 64])
          .build())
      cv_tree = CrossValidator(
        estimator = pipeline_tree,
        estimatorParamMaps = params_tree,
        evaluator = evaluator,
        numFolds = 5
      )
```

[83]:

```
%%time
cv_model_tree = cv_tree.fit(dataset_train)
```

```
CPU times: user 4.23 s, sys: 877 ms, total: 5.11 s
Wall time: 49.2 s
```

[84]:

```
avg_tree_auc = cv_model_tree.avgMetrics
best_tree_auc = max(avg_tree_auc)
best_tree = cv_model_tree.bestModel
print(best_tree.stages[-1].explainParam('maxDepth'))
print(best_tree.stages[-1].explainParam('impurity'))
print(best_tree.stages[-1].explainParam('maxBins'))
print(best_tree_auc)
```

```
maxDepth: Maximum depth of the tree. (>= 0) E.g., depth 0 means 1 leaf node;
depth 1 means 1 internal node + 2 leaf nodes. (default: 5, current: 5)
impurity: Criterion used for information gain calculation (case-insensitive).
Supported options: entropy, gini (default: gini, current: entropy)
maxBins: Max number of bins for discretizing continuous features.  Must be >=2
and >= number of categories for any categorical feature. (default: 32, current:
32)
0.7145830572076701
```

On constate que le problème ne bénéficie pas d'un arbre plus complexe car augmenter le paramètre maxDepth diminuait la précision.

1.6.2 Random Forest

Le second modèle envisagé est une Random Forest.

[85]:

```
forest = RandomForestClassifier(labelCol = 'Outcome', featuresCol =␣
 ↪'scaled_features')
pipeline_forest = Pipeline(stages = [assembler, standardScaler, forest])
params_forest = ParamGridBuilder() \
         .addGrid(forest.featureSubsetStrategy, ['all', 'onethird', 'sqrt',␣
 ↪'log2']) \
         .addGrid(forest.maxDepth, [2, 5, 10]) \
         .addGrid(forest.numTrees, [10, 20, 30]) \
         .addGrid(forest.impurity, ['gini', 'entropy']) \
         .addGrid(forest.maxBins, [32, 64]) \
         .build()
cv_forest = CrossValidator(
  estimator = pipeline_forest,
  estimatorParamMaps = params_forest,
  evaluator = evaluator,
  numFolds = 5
)
```

[86]:

```
%%time
cv_model_forest = cv_forest.fit(dataset_train)
```

```
CPU times: user 47.6 s, sys: 12.4 s, total: 1min
Wall time: 10min 5s
```

[104]:

```
avg_forest_auc = cv_model_forest.avgMetrics
best_forest_auc = max(avg_forest_auc)
best_forest = cv_model_forest.bestModel
print(best_forest.stages[-1].explainParam('featureSubsetStrategy'))
print(best_forest.stages[-1].explainParam('maxDepth'))
print(best_forest.stages[-1].explainParam('numTrees'))
print(best_forest.stages[-1].explainParam('impurity'))
print(best_forest.stages[-1].explainParam('maxBins'))
print(best_forest_auc)
```

```
featureSubsetStrategy: The number of features to consider for splits at each
tree node. Supported options: 'auto' (choose automatically for task: If numTrees
== 1, set to 'all'. If numTrees > 1 (forest), set to 'sqrt' for classification
and to 'onethird' for regression), 'all' (use all features), 'onethird' (use 1/3
of the features), 'sqrt' (use sqrt(number of features)), 'log2' (use log2(number
of features)), 'n' (when n is in the range (0, 1.0], use n * number of features.
When n is in the range (1, number of features), use n features). default =
'auto' (default: auto, current: sqrt)
maxDepth: Maximum depth of the tree. (>= 0) E.g., depth 0 means 1 leaf node;
depth 1 means 1 internal node + 2 leaf nodes. (default: 5, current: 5)
numTrees: Number of trees to train (>= 1). (default: 20, current: 30)
impurity: Criterion used for information gain calculation (case-insensitive).
Supported options: entropy, gini (default: gini, current: entropy)
maxBins: Max number of bins for discretizing continuous features.  Must be >=2
and >= number of categories for any categorical feature. (default: 32, current:
64)
0.8244510830586683
```

1.6.3 Gradient-Boosted Trees

Nous créerons également un Gradient-Boosted Trees pour la comparaison.

[88]:

```
gbt = GBTClassifier(labelCol = 'Outcome', featuresCol = 'scaled_features')
pipeline_gbt = Pipeline(stages = [assembler, standardScaler, gbt])
params_gbt = (ParamGridBuilder()
    .addGrid(gbt.maxDepth, [5, 10])
    .addGrid(gbt.maxIter, [10, 20])
    .addGrid(gbt.stepSize, [0.1, 0.05])
    .build())
cv_gbt = CrossValidator(
  estimator = pipeline_gbt,
  estimatorParamMaps = params_gbt,
  evaluator = evaluator,
  numFolds = 5
)
```

[89]:

```
%%time
cv_model_gbt = cv_gbt.fit(dataset_train)
```

```
CPU times: user 3.54 s, sys: 836 ms, total: 4.37 s
Wall time: 3min
```

[90]:

```
avg_gbt_auc = cv_model_gbt.avgMetrics
best_gbt_auc = max(avg_gbt_auc)
best_gbt = cv_model_gbt.bestModel
print(best_gbt.stages[-1].explainParam('maxDepth'))
print(best_gbt.stages[-1].explainParam('maxIter'))
print(best_gbt.stages[-1].explainParam('stepSize'))
print(best_gbt_auc)
```

```
maxDepth: Maximum depth of the tree. (>= 0) E.g., depth 0 means 1 leaf node;
depth 1 means 1 internal node + 2 leaf nodes. (default: 5, current: 5)
maxIter: max number of iterations (>= 0). (default: 20, current: 20)
stepSize: Step size (a.k.a. learning rate) in interval (0, 1] for shrinking the
contribution of each estimator. (default: 0.1, current: 0.05)
0.7531279156000519
```

1.6.4 Réseau de neurones

Comparons maintenant ces modèles à un réseau de neurone.

[91]:

```
layers = [len(features), 128, 64, 2]

nn = MultilayerPerceptronClassifier(labelCol='Outcome',␣
  ↪featuresCol='scaled_features', layers=layers)
pipeline_nn = Pipeline(stages=[assembler, standardScaler, nn])
params_nn = (ParamGridBuilder()
    .addGrid(nn.maxIter, [50, 100])
    .addGrid(nn.blockSize, [128, 256])
    .addGrid(nn.stepSize, [0.03, 0.1])
    .build())
cv_nn = CrossValidator(
    estimator=pipeline_nn,
    estimatorParamMaps=params_nn,
    evaluator=evaluator,
    numFolds=5
)
```

[92]:

```
%%time
cv_model_nn = cv_nn.fit(dataset_train)
```

```
CPU times: user 2.88 s, sys: 671 ms, total: 3.55 s
Wall time: 2min 13s
```

[93]:

```
avg_nn_auc = cv_model_nn.avgMetrics
best_nn_auc = max(avg_nn_auc)
best_nn = cv_model_nn.bestModel
print(best_nn.stages[-1].explainParam('maxIter'))
print(best_nn.stages[-1].explainParam('blockSize'))
print(best_nn.stages[-1].explainParam('stepSize'))
print(best_nn_auc)
```

```
maxIter: max number of iterations (>= 0). (default: 100, current: 50)
blockSize: block size for stacking input data in matrices. Data is stacked
within partitions. If block size is more than remaining data in a partition then
it is adjusted to the size of this data. (default: 128, current: 128)
stepSize: Step size to be used for each iteration of optimization (>= 0).
(default: 0.03, current: 0.03)
0.8345025554940418
```

1.6.5 Comparaison

On observe que les Random forest sont beaucoup plus efficaces que les arbres de décision pour la cross-validation, et un peu plus efficace que les Gradient Boosted Trees. On observe également que la durée nécessaire pour entraîner un Gradient Boosted Tree est très élevée considérant la grille de paramètres données. Le temps d'entraînement des random forest est probablement dû au grand nombre de paramètres testés.
De son côté, le réseau de neurone utilisé ici est entraîné dans un temps significativement plus court que le gradient-boosted tree. À l'issu de cet entraînement, son taux de précision est supérieur aux trois autres modèles envisagés.

1.7 Évaluation du modèle

1.7.1 Arbre de décision

[94]:

```
%%time
evaluator.evaluate(best_tree.transform(dataset_test))
```

```
CPU times: user 23.7 ms, sys: 3.36 ms, total: 27.1 ms
Wall time: 188 ms
```

[94]:

```
0.7436371100164204
```

1.7.2 Random Forest

[95]:

```
%%time
evaluator.evaluate(best_forest.transform(dataset_test))
```

```
CPU times: user 12.2 ms, sys: 3.64 ms, total: 15.8 ms
Wall time: 158 ms
```

```
[95]: 0.837027914614121
```

1.7.3 Gradient Boosted Trees

```
[96]: %%time
      evaluator.evaluate(best_gbt.transform(dataset_test))
```

```
CPU times: user 17.8 ms, sys: 573 µs, total: 18.4 ms
Wall time: 163 ms
```

```
[96]: 0.7702175697865353
```

1.7.4 Neural network

```
[97]: %%time
      evaluator.evaluate(best_nn.transform(dataset_test))
```

```
CPU times: user 14.3 ms, sys: 708 µs, total: 15 ms
Wall time: 136 ms
```

```
[97]: 0.8357963875205253
```

1.7.5 Comparaison

On observe que la forêt met 2 fois plus de temps à s'exécuter mais reste néanmoins plus efficace. On observe également que le Gradient Boosted Trees, une fois entraînée, est plus rapide et plus précis que les arbres de décisions. Le réseau de neurone semble cependant être la meilleure solution avec un taux de précision 5% supérieur au gradient boosted tree our une durée d'exécution similaire au random forest.

1.7.6 Visualisation

Déterminons les diagnostics réels et prédits par le Random forest et le neural network.

Random forest

```
[98]: predictions = best_tree.transform(dataset_test)
      labels_and_scores = predictions.select("Outcome", "probability") \
        .rdd.map(lambda row: (float(row.Outcome), float(row.probability[1])))
      labels_and_scores = np.array(labels_and_scores.collect())

      true_labels = labels_and_scores[:, 0]
      predicted_probabilities = labels_and_scores[:, 1]
```

Affichons la matrice de confusion du modèle.

```
[99]: conf_matrix = confusion_matrix(true_labels, (predicted_probabilities > 0.5).
       ↪astype(int))
      print(conf_matrix)
```

```
[[69 15]
 [28 30]]
```

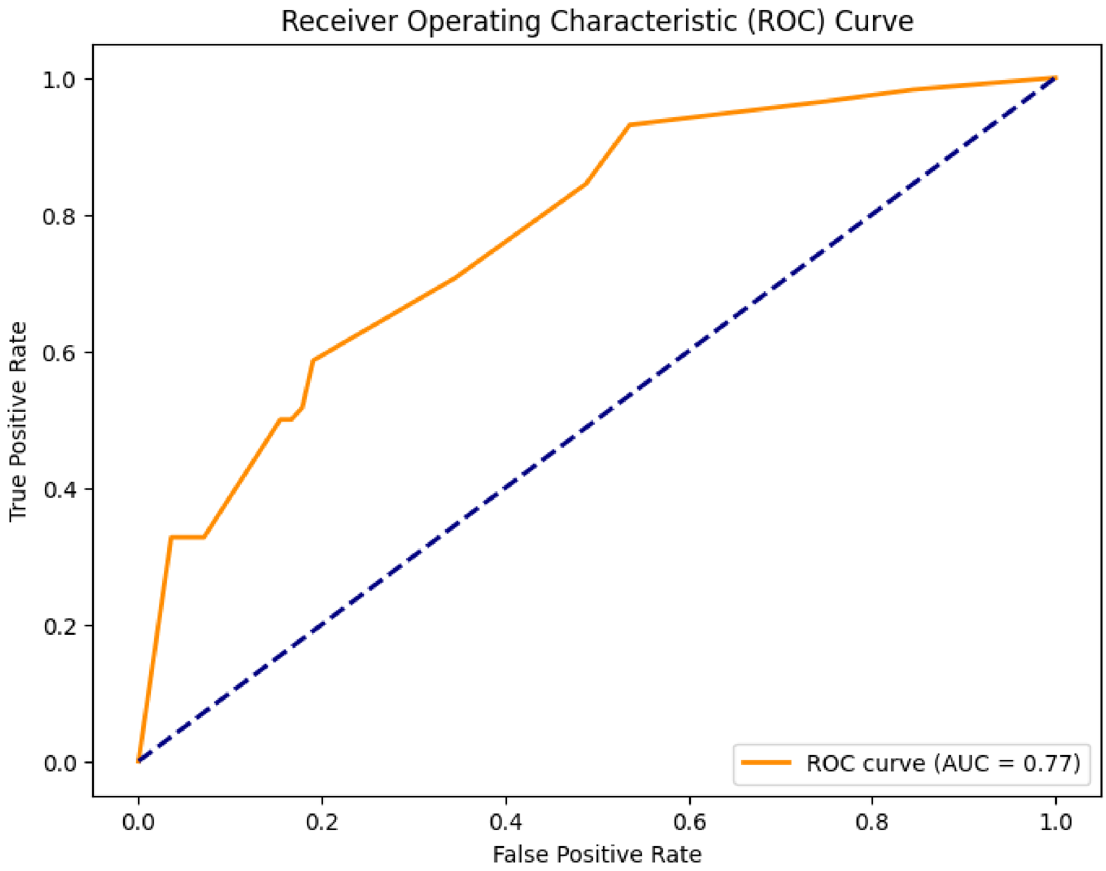
On observe que le programme n'est pas très efficace à détecter les vrais positifs, probablement dû au fait que les négatifs soient sur-représentés.

Affichons la courbe ROC:

[100]:

```
fpr, tpr, thresholds = roc_curve(true_labels, predicted_probabilities)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (AUC = {:.2f})'.
  ↪format(roc_auc))
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```



On observe que l'aire sous la courbe ROC, et donc la probabilité que le label soit plus élevé pour un malade que pour un non malade, est de 77%. Le marqueur est donc informatif mais semble peu efficace.

Neural network

[101]:

```
predictions = best_nn.transform(dataset_test)
labels_and_scores = predictions.select("Outcome", "probability") \
  .rdd.map(lambda row: (float(row.Outcome), float(row.probability[1])))
labels_and_scores = np.array(labels_and_scores.collect())

true_labels = labels_and_scores[:, 0]
predicted_probabilities = labels_and_scores[:, 1]
```

Affichons la matrice de confusion du modèle.

[102]:

```
conf_matrix = confusion_matrix(true_labels, (predicted_probabilities > 0.5).
 ↪astype(int))
print(conf_matrix)
```

```
[[74 10]
 [25 33]]
```

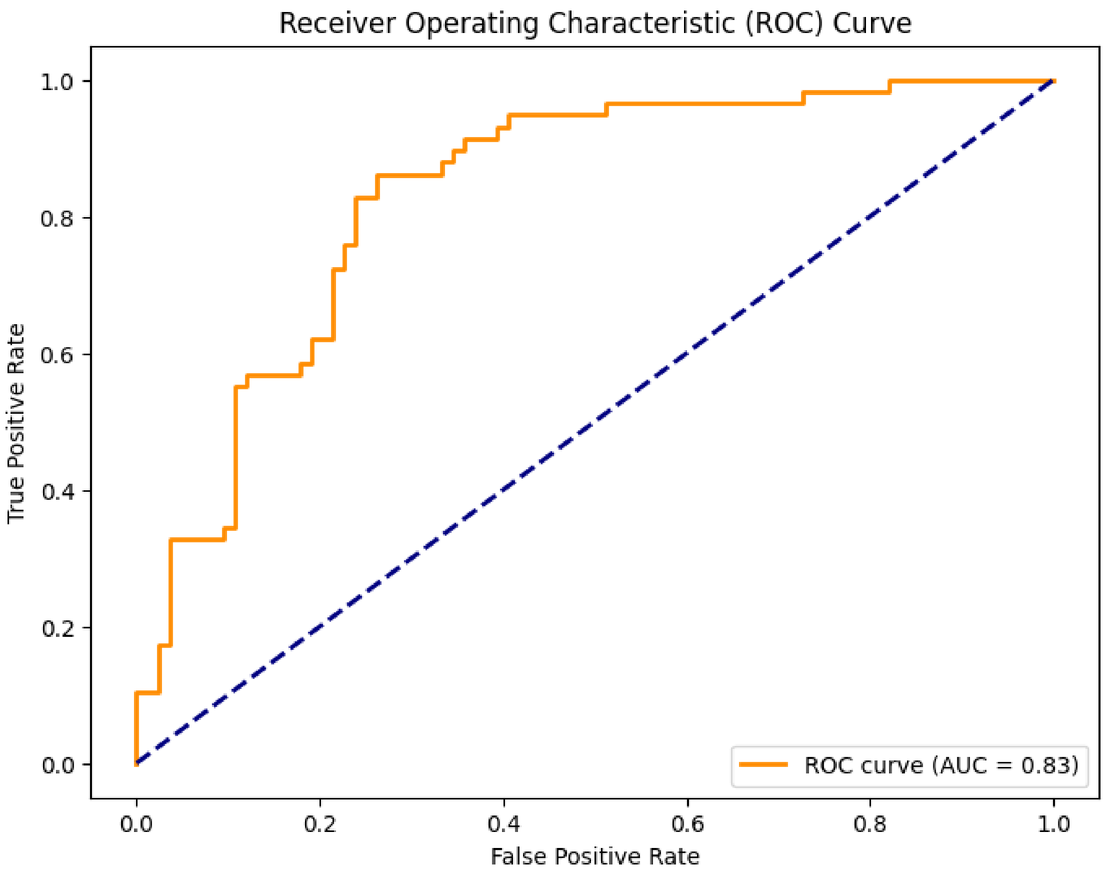
Le constat est similaire à celui obtenu pour les random forest: on observe que le programme n'est pas très efficace à détecter les vrais positifs, probablement dû au fait que les négatifs soient sur-représentés.

Affichons la courbe ROC:

[103]:

```
fpr, tpr, thresholds = roc_curve(true_labels, predicted_probabilities)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (AUC = {:.2f})'.
 ↪format(roc_auc))
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```



On observe que l'aire sous la courbe ROC, et donc la probabilité que le label soit plus élevé pour un malade que pour un non malade, est de 83%. Le marqueur est donc bien plus informatif que celui des random forest.