hd4xzrz5t

August 2, 2023

[1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

[2]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[3]:

```
df=pd.read_csv("/content/drive/MyDrive/mydatasets/C5_health care diabetes.csv")
df
```

[3]:

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
1              1       85             66             29        0  26.6
2              8      183             64              0        0  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           …      …            …            …      …   …
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age  Outcome
0                       0.627   50        1
1                       0.351   31        0
2                       0.672   32        1
3                       0.167   21        0
4                       2.288   33        1
..                        …  …      …
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
```

```
766                       0.349   47          1
767                       0.315   23          0

[768 rows x 9 columns]
```

```
[4]: df.head()
```

```
[4]:    Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
     0            6      148             72             35        0  33.6
     1            1       85             66             29        0  26.6
     2            8      183             64              0        0  23.3
     3            1       89             66             23       94  28.1
     4            0      137             40             35      168  43.1

        DiabetesPedigreeFunction  Age  Outcome
     0                     0.627   50        1
     1                     0.351   31        0
     2                     0.672   32        1
     3                     0.167   21        0
     4                     2.288   33        1
```

1 Data Cleaning and Data Preprocessing

```
[5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

```
[6]: df.describe()
```

```
[6]:        Pregnancies     Glucose  BloodPressure  SkinThickness     Insulin  \
     count   768.000000  768.000000     768.000000     768.000000  768.000000
```

```
mean      3.845052  120.894531      69.105469      20.536458   79.799479
std       3.369578   31.972618      19.355807      15.952218  115.244002
min       0.000000    0.000000       0.000000       0.000000    0.000000
25%       1.000000   99.000000      62.000000       0.000000    0.000000
50%       3.000000  117.000000      72.000000      23.000000   30.500000
75%       6.000000  140.250000      80.000000      32.000000  127.250000
max      17.000000  199.000000     122.000000      99.000000  846.000000

              BMI  DiabetesPedigreeFunction         Age     Outcome
count  768.000000                768.000000  768.000000  768.000000
mean    31.992578                  0.471876   33.240885    0.348958
std      7.884160                  0.331329   11.760232    0.476951
min      0.000000                  0.078000   21.000000    0.000000
25%     27.300000                  0.243750   24.000000    0.000000
50%     32.000000                  0.372500   29.000000    0.000000
75%     36.600000                  0.626250   41.000000    1.000000
max     67.100000                  2.420000   81.000000    1.000000
```

```
[7]: df.columns
```

```
[7]: Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
            'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
           dtype='object')
```

```
[8]: feature_matrix = df.iloc[:,0:8]
     target_vector = df.iloc[:,-1]
```

```
[9]: fs = StandardScaler().fit_transform(feature_matrix)
     logr = LogisticRegression()
     logr.fit(fs,target_vector)
```

```
[9]: LogisticRegression()
```

```
[10]: observation=[[1,2,3,4,5,6,7,8]]
      prediction = logr.predict(observation)
      print(prediction)
```

```
      [1]
```

```
[11]: logr.classes_
```

```
[11]: array([0, 1])
```

```
[12]: logr.predict_proba(observation)
```

```
[12]: array([[2.92369487e-04, 9.99707631e-01]])
```

Random Forest

```
[13]: x = df.iloc[:,0:8]
      y = df.iloc[:,-1]
```

```
[14]: from sklearn.model_selection import train_test_split
      x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

```
[15]: from sklearn.ensemble import RandomForestClassifier
      rfc = RandomForestClassifier()
      rfc.fit(x_train,y_train)
```

```
[15]: RandomForestClassifier()
```

```
[16]: parameters = {'max_depth':[1,2,3,4,5],'min_samples_leaf':[5,10,15,20,25],
                     'n_estimators': [10,20,30,40,50]
                     }
```

```
[17]: from sklearn.model_selection import GridSearchCV
      grid_search =␣
       ↪GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
      grid_search.fit(x_train,y_train)
```

```
[17]: GridSearchCV(cv=2, estimator=RandomForestClassifier(),
                   param_grid={'max_depth': [1, 2, 3, 4, 5],
                               'min_samples_leaf': [5, 10, 15, 20, 25],
                               'n_estimators': [10, 20, 30, 40, 50]},
                   scoring='accuracy')
```

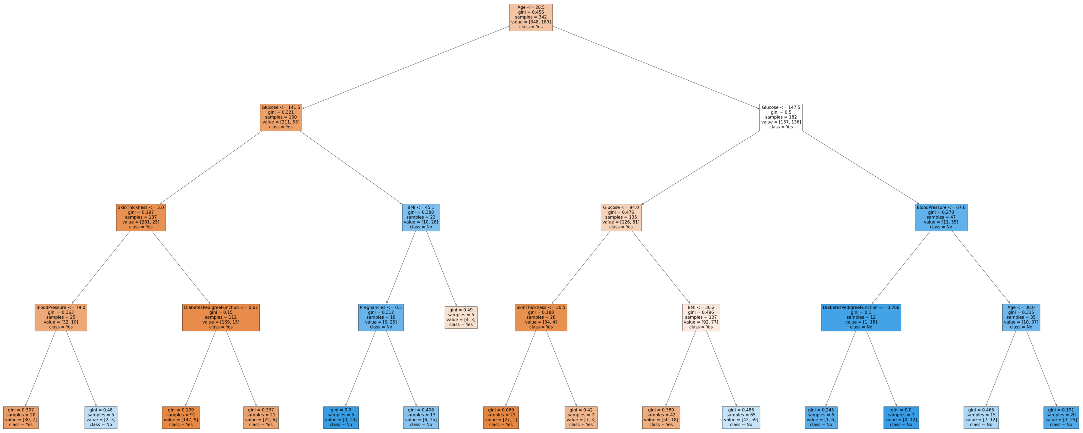
```
[18]: grid_search.best_score_
```

```
[18]: 0.7765424735060755
```

```
[19]: rfc_best = grid_search.best_estimator_
```

```
[20]: from sklearn.tree import plot_tree
      plt.figure(figsize=(89,40))
      plot_tree(rfc_best.estimators_[5], feature_names=x.columns, class_names=['Yes',␣
       ↪'No'], filled=True)
```

```
[20]: [Text(0.49107142857142855, 0.9, 'Age <= 28.5\ngini = 0.456\nsamples = 342\nvalue
      = [348, 189]\nclass = Yes'),
       Text(0.26785714285714285, 0.7, 'Glucose <= 141.5\ngini = 0.321\nsamples =
      160\nvalue = [211, 53]\nclass = Yes'),
       Text(0.14285714285714285, 0.5, 'SkinThickness <= 5.0\ngini = 0.197\nsamples =
      137\nvalue = [201, 25]\nclass = Yes'),
       Text(0.07142857142857142, 0.3, 'BloodPressure <= 79.0\ngini = 0.363\nsamples =
      25\nvalue = [32, 10]\nclass = Yes'),
       Text(0.03571428571428571, 0.1, 'gini = 0.307\nsamples = 20\nvalue = [30,
      7]\nclass = Yes'),
       Text(0.10714285714285714, 0.1, 'gini = 0.48\nsamples = 5\nvalue = [2, 3]\nclass
      = No'),
       Text(0.21428571428571427, 0.3, 'DiabetesPedigreeFunction <= 0.67\ngini =
      0.15\nsamples = 112\nvalue = [169, 15]\nclass = Yes'),
       Text(0.17857142857142858, 0.1, 'gini = 0.109\nsamples = 91\nvalue = [147,
      9]\nclass = Yes'),
       Text(0.25, 0.1, 'gini = 0.337\nsamples = 21\nvalue = [22, 6]\nclass = Yes'),
       Text(0.39285714285714285, 0.5, 'BMI <= 45.1\ngini = 0.388\nsamples = 23\nvalue
      = [10, 28]\nclass = No'),
       Text(0.35714285714285715, 0.3, 'Pregnancies <= 0.5\ngini = 0.312\nsamples =
      18\nvalue = [6, 25]\nclass = No'),
       Text(0.32142857142857145, 0.1, 'gini = 0.0\nsamples = 5\nvalue = [0, 10]\nclass
      = No'),
       Text(0.39285714285714285, 0.1, 'gini = 0.408\nsamples = 13\nvalue = [6,
      15]\nclass = No'),
       Text(0.42857142857142855, 0.3, 'gini = 0.49\nsamples = 5\nvalue = [4, 3]\nclass
      = Yes'),
       Text(0.7142857142857143, 0.7, 'Glucose <= 147.5\ngini = 0.5\nsamples =
      182\nvalue = [137, 136]\nclass = Yes'),
       Text(0.5714285714285714, 0.5, 'Glucose <= 94.0\ngini = 0.476\nsamples =
      135\nvalue = [126, 81]\nclass = Yes'),
       Text(0.5, 0.3, 'SkinThickness <= 30.5\ngini = 0.188\nsamples = 28\nvalue = [34,
      4]\nclass = Yes'),
       Text(0.4642857142857143, 0.1, 'gini = 0.069\nsamples = 21\nvalue = [27,
      1]\nclass = Yes'),
       Text(0.5357142857142857, 0.1, 'gini = 0.42\nsamples = 7\nvalue = [7, 3]\nclass
      = Yes'),
       Text(0.6428571428571429, 0.3, 'BMI <= 30.2\ngini = 0.496\nsamples = 107\nvalue
      = [92, 77]\nclass = Yes'),
       Text(0.6071428571428571, 0.1, 'gini = 0.389\nsamples = 42\nvalue = [50,
      18]\nclass = Yes'),
       Text(0.6785714285714286, 0.1, 'gini = 0.486\nsamples = 65\nvalue = [42,
      59]\nclass = No'),
       Text(0.8571428571428571, 0.5, 'BloodPressure <= 67.0\ngini = 0.278\nsamples =
      47\nvalue = [11, 55]\nclass = No'),
       Text(0.7857142857142857, 0.3, 'DiabetesPedigreeFunction <= 0.268\ngini =
      0.1\nsamples = 12\nvalue = [1, 18]\nclass = No'),
       Text(0.75, 0.1, 'gini = 0.245\nsamples = 5\nvalue = [1, 6]\nclass = No'),
       Text(0.8214285714285714, 0.1, 'gini = 0.0\nsamples = 7\nvalue = [0, 12]\nclass
      = No'),
       Text(0.9285714285714286, 0.3, 'Age <= 38.0\ngini = 0.335\nsamples = 35\nvalue =
      [10, 37]\nclass = No'),
       Text(0.8928571428571429, 0.1, 'gini = 0.465\nsamples = 15\nvalue = [7,
      12]\nclass = No'),
       Text(0.9642857142857143, 0.1, 'gini = 0.191\nsamples = 20\nvalue = [3,
      25]\nclass = No')]
```


Age <= 28.5
gini = 0.456
samples = 342
value = [348, 189]
class = Yes
Glucose <= 141.5
gini = 0.321
samples = 160
value = [211, 53]
class = Yes
Glucose <= 147.5
gini = 0.5
samples = 182
value = [137, 136]
class = Yes
SkinThickness <= 5.0
gini = 0.197
samples = 137
value = [201, 25]
class = Yes
BMI <= 45.1
gini = 0.388
samples = 23
value = [10, 28]
class = No
Glucose <= 94.0
gini = 0.476
samples = 135
value = [126, 81]
class = Yes
BloodPressure <= 67.0
gini = 0.278
samples = 47
value = [11, 55]
class = No
BloodPressure <= 79.0
gini = 0.363
samples = 25
value = [32, 10]
class = Yes
DiabetesPedigreeFunction <= 0.67
gini = 0.15
samples = 112
value = [169, 15]
class = Yes
Pregnancies <= 0.5
gini = 0.312
samples = 18
value = [6, 25]
class = No
gini = 0.49
samples = 5
value = [4, 3]
class = Yes
SkinThickness <= 30.5
gini = 0.188
samples = 28
value = [34, 4]
class = Yes
BMI <= 30.2
gini = 0.496
samples = 107
value = [92, 77]
class = Yes
DiabetesPedigreeFunction <= 0.268
gini = 0.1
samples = 12
value = [1, 18]
class = No
Age <= 38.0
gini = 0.335
samples = 35
value = [10, 37]
class = No
gini = 0.307
samples = 20
value = [30, 7]
class = Yes
gini = 0.48
samples = 5
value = [2, 3]
class = No
gini = 0.109
samples = 91
value = [147, 9]
class = Yes
gini = 0.337
samples = 21
value = [22, 6]
class = Yes
gini = 0.0
samples = 5
value = [0, 10]
class = No
gini = 0.408
samples = 13
value = [6, 15]
class = No
gini = 0.069
samples = 21
value = [27, 1]
class = Yes
gini = 0.42
samples = 7
value = [7, 3]
class = Yes
gini = 0.389
samples = 42
value = [50, 18]
class = Yes
gini = 0.486
samples = 65
value = [42, 59]
class = No
gini = 0.245
samples = 5
value = [1, 6]
class = No
gini = 0.0
samples = 7
value = [0, 12]
class = No
gini = 0.465
samples = 15
value = [7, 12]
class = No
gini = 0.191
samples = 20
value = [3, 25]
class = No