sxb7ennj2

April 29, 2024

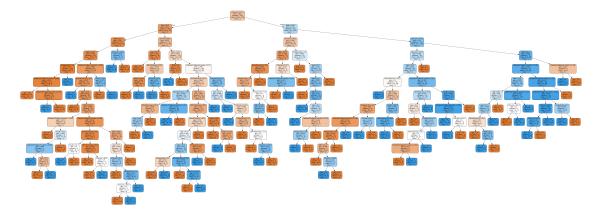
[]: #import pandas library

```
import pandas as pd
[]: #loading dataset
     df=pd.read_csv("/content/drive/My Drive/Colab Notebooks/diabetes_dataset.csv")
     df.head()
[]:
        Pregnancies Glucose BloodPressure ... DiabetesPedigreeFunction
                                                                             Age
     Outcome
                  6
                          148
                                          72 ...
                                                                     0.627
     0
                                                                              50
     1
                          85
                                                                     0.351
     1
                  1
                                          66 ...
                                                                              31
     0
     2
                                                                     0.672
                                                                              32
                  8
                          183
                                          64 ...
     1
     3
                  1
                          89
                                          66 ...
                                                                     0.167
                                                                              21
     0
     4
                  0
                          137
                                          40 ...
                                                                     2.288
                                                                              33
     [5 rows x 9 columns]
[]: #feature variables
     x=df.drop(['Outcome'], axis=1)
[]:
          Pregnancies Glucose BloodPressure ...
                                                    BMI DiabetesPedigreeFunction
     Age
     0
                    6
                            148
                                            72 ...
                                                    33.6
                                                                              0.627
     50
     1
                             85
                                            66 ...
                                                    26.6
                                                                              0.351
     31
     2
                    8
                            183
                                            64 ...
                                                    23.3
                                                                              0.672
     32
     3
                                                    28.1
                                                                              0.167
                    1
                             89
                                            66 ...
     21
                    0
                            137
                                            40 ... 43.1
                                                                              2.288
```

```
33
     . .
                                                                                     0.171
     763
                     10
                              101
                                                76 ... 32.9
     63
     764
                      2
                              122
                                                70 ...
                                                        36.8
                                                                                     0.340
     27
     765
                      5
                              121
                                                72 ...
                                                       26.2
                                                                                     0.245
     30
     766
                      1
                              126
                                                60
                                                        30.1
                                                                                     0.349
     47
     767
                      1
                               93
                                                70 ... 30.4
                                                                                     0.315
     23
     [768 rows x 8 columns]
[]: #target variable
     y=df.Outcome
     У
[]: 0
             1
     1
             0
     2
             1
     3
             0
     4
             1
     763
             0
     764
             0
     765
             0
     766
             1
     767
     Name: Outcome, Length: 768, dtype: int64
[]: from sklearn.tree import DecisionTreeClassifier # Import Decision Tree_
      \hookrightarrowClassifier
     {\tt from \ sklearn.model\_selection \ import \ train\_test\_split \ \# \ \mathit{Import \ train\_test\_split}_{\square}}
      \hookrightarrow function
     x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,__
       →random_state=1)
[]: # Create Decision Tree classifer object
     model = DecisionTreeClassifier()
     # Train Decision Tree Classifer
     model = model.fit(x_train,y_train)
     #Predict the response for test dataset
```

```
y_pred = model.predict(x_test)
[]: | #Evaluation using Accuracy score
     from sklearn import metrics #Import scikit-learn metrics module for accuracy,
      \hookrightarrow calculation
     print("Accuracy:",metrics.accuracy_score(y_test, y_pred)*100)
    Accuracy: 67.53246753246754
[]: #Evaluation using Confusion matrix
     from sklearn.metrics import confusion_matrix
     confusion_matrix(y_test,y_pred)
[]: array([[76, 23],
            [27, 28]])
[]: print("Accuracy:",((82+27)/154))
    Accuracy: 0.7077922077922078
[]: #Evaluation using Classification report
     from sklearn.metrics import classification_report
     print(classification_report(y_test,y_pred))
                  precision
                                recall f1-score
                                                   support
                        0.74
                                  0.77
                                            0.75
               0
                                                        99
               1
                        0.55
                                  0.51
                                            0.53
                                                        55
                                            0.68
                                                        154
        accuracy
                        0.64
                                  0.64
                                            0.64
                                                        154
       macro avg
    weighted avg
                        0.67
                                  0.68
                                            0.67
                                                        154
[]: #checking prediction value
     model.predict([[6,148,72,35,0,33.6,0.627,50]])
[]: array([1])
[]: #Import modules for Visualizing Decision trees
     from sklearn.tree import export_graphviz
     from sklearn.externals.six import StringIO
     from IPython.display import Image
     import pydotplus
[]: features=x.columns
     features
```

[]:



```
[]: # Create Decision Tree classifer object
model = DecisionTreeClassifier(criterion="entropy", max_depth=3)

# Train Decision Tree Classifer
model = model.fit(x_train,y_train)

#Predict the response for test dataset
y_pred = model.predict(x_test)

# Model Accuracy
print("Accuracy:",metrics.accuracy_score(y_test, y_pred)*100)
```

Accuracy: 79.87012987012987

The classification rate increased to 79.87%, which is better accuracy than the previous model.

```
[]: #Better Decision Tree Visualisation
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dot_data = StringIO()
```



