# DecisionTree_HeartData

October 16, 2024

[1]:

```
# importing the necessary libraries
import numpy as np, pandas as pd
import matplotlib.pyplot as plt, seaborn as sns
```

[2]:

```
# reading the csv data and storing it in df
df = pd.read_csv('heart_v2.csv')
df.head()
```

[2]:

```
   age  sex   BP  cholestrol  heart disease
0   70    1  130         322              1
1   67    0  115         564              0
2   57    1  124         261              1
3   64    1  128         263              0
4   74    0  120         269              0
```

[3]:

```
df.isnull().sum()
```

[3]:

```
age              0
sex              0
BP               0
cholestrol       0
heart disease    0
dtype: int64
```

[4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 270 entries, 0 to 269
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   age            270 non-null    int64
 1   sex            270 non-null    int64
 2   BP             270 non-null    int64
 3   cholestrol     270 non-null    int64
 4   heart disease  270 non-null    int64
dtypes: int64(5)
memory usage: 10.7 KB
```

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

[5]:
```
              age         sex          BP  cholestrol  heart disease
count  270.000000  270.000000  270.000000  270.000000     270.000000
mean    54.433333    0.677778  131.344444  249.659259       0.444444
std      9.109067    0.468195   17.861608   51.686237       0.497827
min     29.000000    0.000000   94.000000  126.000000       0.000000
25%     48.000000    0.000000  120.000000  213.000000       0.000000
50%     55.000000    1.000000  130.000000  245.000000       0.000000
75%     61.000000    1.000000  140.000000  280.000000       1.000000
max     77.000000    1.000000  200.000000  564.000000       1.000000
```

[6]:
```
pip install six
```

```
Defaulting to user installation because normal site-packages is not writeable
Looking in links: /usr/share/pip-wheels
Requirement already satisfied: six in /opt/conda/envs/anaconda-
panel-2023.05-py310/lib/python3.11/site-packages (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

[7]:
```
pip install pydotplus
```

```
Defaulting to user installation because normal site-packages is not writeable
Looking in links: /usr/share/pip-wheels
Requirement already satisfied: pydotplus in ./.local/lib/python3.11/site-
packages (2.0.2)
Requirement already satisfied: pyparsing>=2.0.1 in /opt/conda/envs/anaconda-
panel-2023.05-py310/lib/python3.11/site-packages (from pydotplus) (3.0.9)
Note: you may need to restart the kernel to use updated packages.
```

[8]:
```
pip install graphviz
```

```
Defaulting to user installation because normal site-packages is not writeable
Looking in links: /usr/share/pip-wheels
Requirement already satisfied: graphviz in ./.local/lib/python3.11/site-packages
(0.20.3)
Note: you may need to restart the kernel to use updated packages.
```

[9]:
```
# define the X and y
X = df.drop('heart disease', axis = 1)
y = df['heart disease']
```

[10]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,␣
 ↪random_state = 100)
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

[10]:

```
((216, 4), (216,), (54, 4), (54,))
```

[11]:

```
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier(max_depth = 3)
dt.fit(X_train, y_train)
```

[11]:

```
DecisionTreeClassifier(max_depth=3)
```

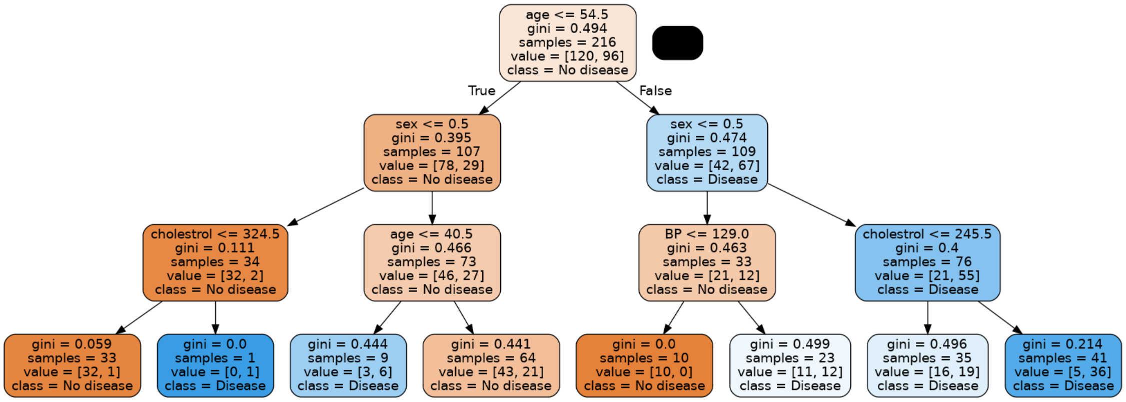
[12]:

```
from IPython.display import Image
from six import StringIO
from sklearn.tree import export_graphviz
import pydotplus, graphviz
```

[13]:

```
dot_data = StringIO()
export_graphviz(dt, out_file = dot_data, filled = True, rounded = True,␣
 ↪feature_names = X.columns, class_names = ['No disease', 'Disease'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

[13]:



[redacted]

[14]:

```
y_train_pred = dt.predict(X_train)
y_train_pred
```

[14]:

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

[15]:
```
y_test_pred = dt.predict(X_test)
y_test_pred
```

[15]:
```
array([0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0,
       1, 1, 1, 0, 0, 0, 1, 1, 0, 0])
```

[16]:
```
# evaluations
from sklearn.metrics import confusion_matrix, accuracy_score
print("Accuracy Score for Training Data", accuracy_score(y_train ,y_train_pred))
print("----------------------------------------------------")
print("Confusion Matrix\n", confusion_matrix(y_train ,y_train_pred))

print("----------------------------------------------------")
print("Accuracy Score for Test Data", accuracy_score(y_test ,y_test_pred))
print("----------------------------------------------------")
print("Confusion Matrix\n", confusion_matrix(y_test ,y_test_pred))
```

```
Accuracy Score for Training Data 0.7361111111111112
----------------------------------------------------
Confusion Matrix
 [[85 35]
 [22 74]]
----------------------------------------------------
Accuracy Score for Test Data 0.5
----------------------------------------------------
Confusion Matrix
 [[15 15]
 [12 12]]
```

[17]:
```
def get_dt_graph(dt_classifier):
    dot_data = StringIO()
    export_graphviz(dt_classifier, out_file = dot_data, filled = True, rounded␣
 ↪= True, feature_names = X.columns, class_names = ['No disease', 'Disease'])
    graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
    return graph
```

[28]:
```
def evaluate_model(dt_classifier):
    print("Train Accuracy Score", accuracy_score(y_train , dt_classifier.
 ↪predict(X_train)))
    print("----------------------------------------------------")
    print("Train data's Confusion Matrix\n", confusion_matrix(y_train,␣
 ↪dt_classifier.predict(X_train)))

    print("-"*50)
    print("Test Accuracy Score", accuracy_score(y_test ,dt_classifier.
 ↪predict(X_test)))
```

```
    print("----------------------------------------------------")
    print("Test data's Confusion Matrix\n", confusion_matrix(y_test␣
↪,dt_classifier.predict(X_test)))
```

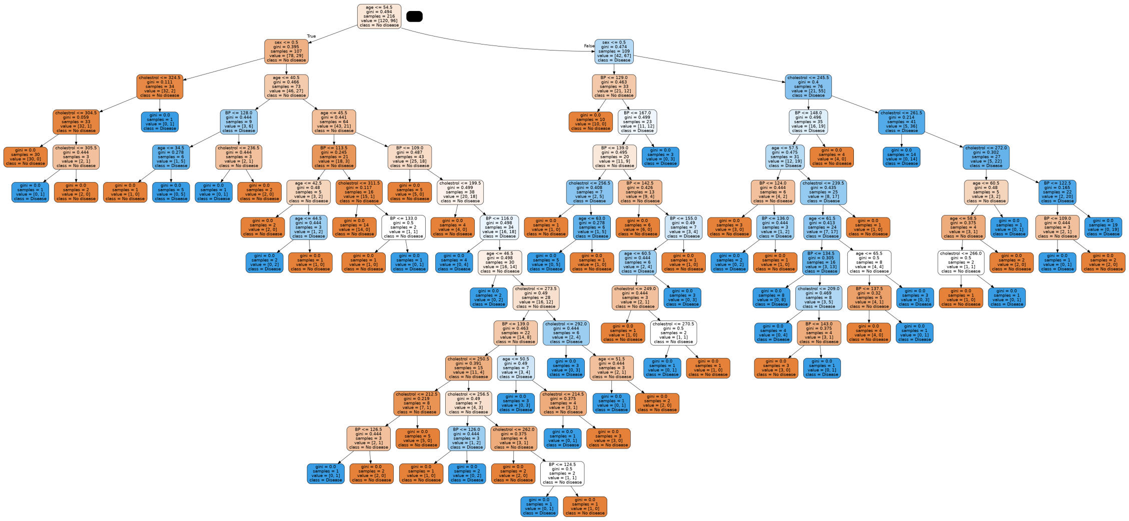
[32]:
```
# Random_state = 42 without setting any hyper parameters

dt1 = DecisionTreeClassifier(random_state = 42)
dt1.fit(X_train, y_train)
```

[32]:
```
DecisionTreeClassifier(random_state=42)
```

[34]:
```
gph = get_dt_graph(dt1)
Image(gph.create_png())
```

[34]:



[36]:
```
evaluate_model(dt1)
```

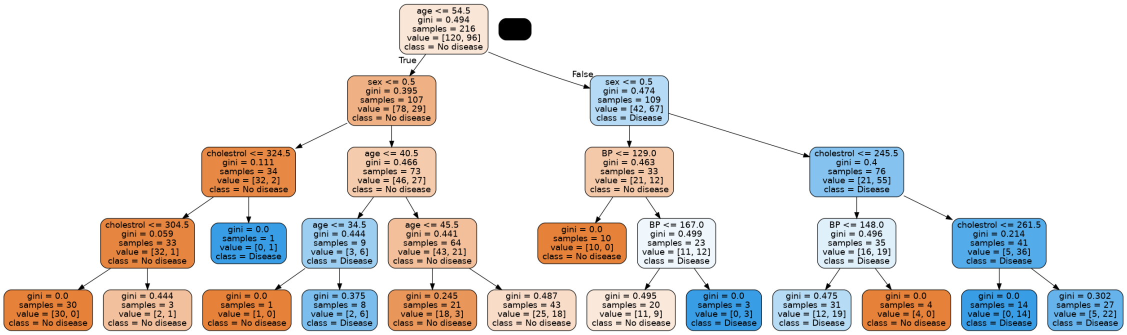
```
Train Accuracy Score 1.0
----------------------------------------------------
Train data's Confusion Matrix
 [[120   0]
 [  0  96]]
--------------------------------------------------
Test Accuracy Score 0.5370370370370371
----------------------------------------------------
Test data's Confusion Matrix
 [[19 11]
 [14 10]]
```

[38]:
```
dt2 = DecisionTreeClassifier(max_depth = 4)
dt2.fit(X_train, y_train)
```

[38]: DecisionTreeClassifier(max_depth=4)

[40]:
```
gph = get_dt_graph(dt2)
Image(gph.create_png())
```

[40]:



age <= 54.5 gini = 0.494 samples = 216 value = [120, 96] class = No disease
[redacted]
True
False
sex <= 0.5 gini = 0.395 samples = 107 value = [78, 29] class = No disease
sex <= 0.5 gini = 0.474 samples = 109 value = [42, 67] class = Disease
cholestrol <= 324.5 gini = 0.111 samples = 34 value = [32, 2] class = No disease
age <= 40.5 gini = 0.466 samples = 73 value = [46, 27] class = No disease
BP <= 129.0 gini = 0.463 samples = 33 value = [21, 12] class = No disease
cholestrol <= 245.5 gini = 0.4 samples = 76 value = [21, 55] class = Disease
cholestrol <= 304.5 gini = 0.059 samples = 33 value = [32, 1] class = No disease
gini = 0.0 samples = 1 value = [0, 1] class = Disease
age <= 34.5 gini = 0.444 samples = 9 value = [3, 6] class = Disease
age <= 45.5 gini = 0.441 samples = 64 value = [43, 21] class = No disease
gini = 0.0 samples = 10 value = [10, 0] class = No disease
BP <= 167.0 gini = 0.499 samples = 23 value = [11, 12] class = Disease
BP <= 148.0 gini = 0.496 samples = 35 value = [16, 19] class = Disease
cholestrol <= 261.5 gini = 0.214 samples = 41 value = [5, 36] class = Disease
gini = 0.0 samples = 30 value = [30, 0] class = No disease
gini = 0.444 samples = 3 value = [2, 1] class = No disease
gini = 0.0 samples = 1 value = [1, 0] class = No disease
gini = 0.375 samples = 8 value = [2, 6] class = Disease
gini = 0.245 samples = 21 value = [18, 3] class = No disease
gini = 0.487 samples = 43 value = [25, 18] class = No disease
gini = 0.495 samples = 20 value = [11, 9] class = No disease
gini = 0.0 samples = 3 value = [0, 3] class = Disease
gini = 0.475 samples = 31 value = [12, 19] class = Disease
gini = 0.0 samples = 4 value = [4, 0] class = No disease
gini = 0.0 samples = 14 value = [0, 14] class = Disease
gini = 0.302 samples = 27 value = [5, 22] class = Disease

[42]:
```
evaluate_model(dt2)
```

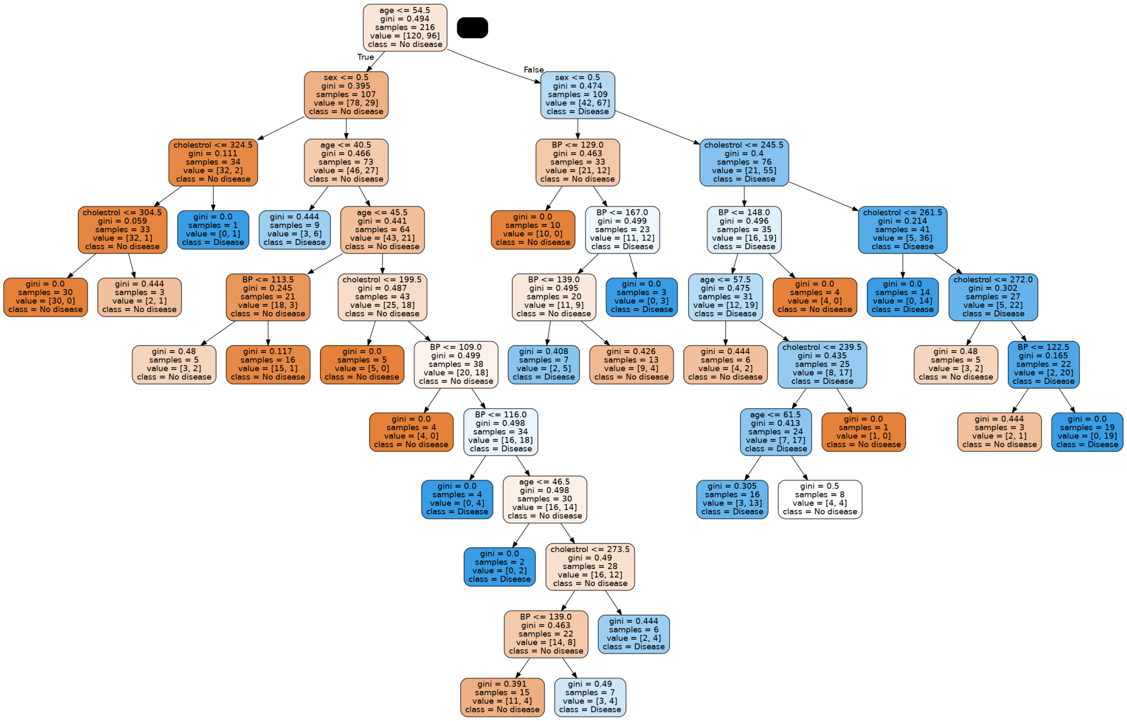
```
Train Accuracy Score 0.7685185185185185
--------------------------------------------------
Train data's Confusion Matrix
 [[101  19]
 [ 31  65]]
------------------------------------------------
Test Accuracy Score 0.6111111111111112
--------------------------------------------------
Test data's Confusion Matrix
 [[25  5]
 [16  8]]
```

[44]:
```
dt3 = DecisionTreeClassifier(min_samples_split = 20)
dt3.fit(X_train, y_train)
```

[44]: DecisionTreeClassifier(min_samples_split=20)

[46]:
```
gph = get_dt_graph(dt3)
Image(gph.create_png())
```

[46]:



[47]:
```
evaluate_model(dt3)
```

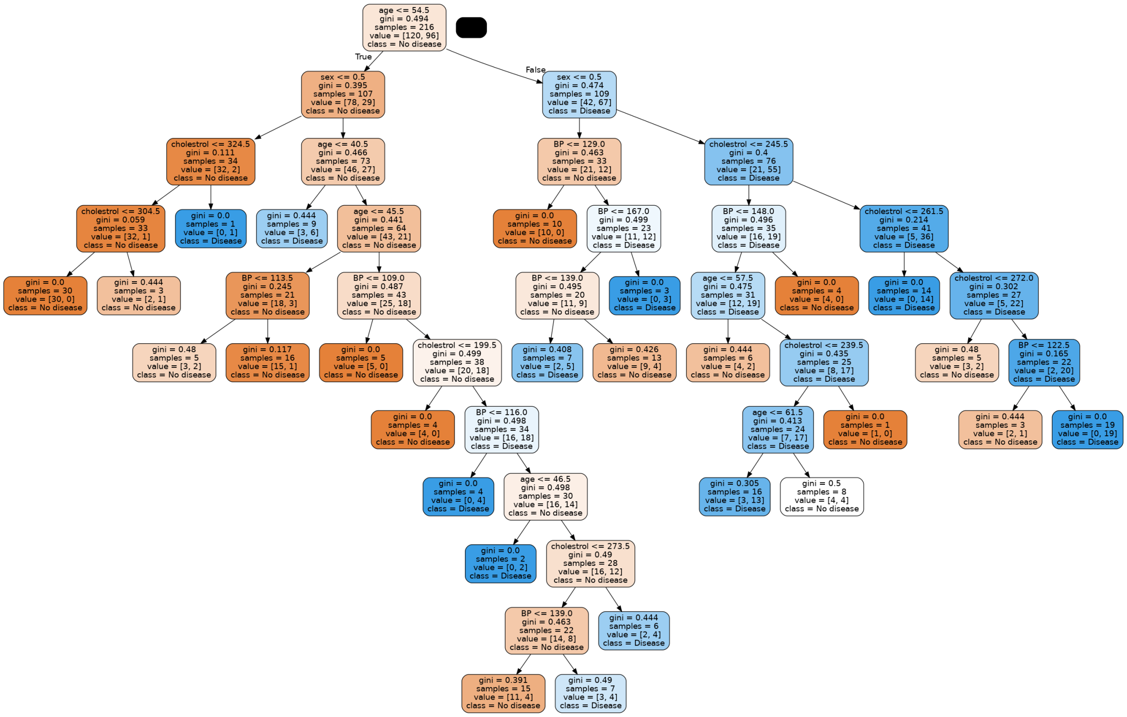
```
Train Accuracy Score 0.8425925925925926
--------------------------------------------------
Train data's Confusion Matrix
 [[107  13]
 [ 21  75]]
--------------------------------------------------
Test Accuracy Score 0.5740740740740741
--------------------------------------------------
Test data's Confusion Matrix
 [[24  6]
 [17  7]]
```

[50]:
```
dt4 = DecisionTreeClassifier(min_samples_split = 20, random_state = 75)
dt4.fit(X_train, y_train)
```

[50]:
```
DecisionTreeClassifier(min_samples_split=20, random_state=75)
```

[52]:
```
gph = get_dt_graph(dt4)
Image(gph.create_png())
```

[52]:


age <= 54.5 gini = 0.494 samples = 216 value = [120, 96] class = No disease
[redacted]
True
False
sex <= 0.5 gini = 0.395 samples = 107 value = [78, 29] class = No disease
sex <= 0.5 gini = 0.474 samples = 109 value = [42, 67] class = Disease
cholestrol <= 324.5 gini = 0.111 samples = 34 value = [32, 2] class = No disease
age <= 40.5 gini = 0.466 samples = 73 value = [46, 27] class = No disease
BP <= 129.0 gini = 0.463 samples = 33 value = [21, 12] class = No disease
cholestrol <= 245.5 gini = 0.4 samples = 76 value = [21, 55] class = Disease
cholestrol <= 304.5 gini = 0.059 samples = 33 value = [32, 1] class = No disease
gini = 0.0 samples = 1 value = [0, 1] class = Disease
gini = 0.444 samples = 9 value = [3, 6] class = Disease
age <= 45.5 gini = 0.441 samples = 64 value = [43, 21] class = No disease
gini = 0.0 samples = 10 value = [10, 0] class = No disease
BP <= 167.0 gini = 0.499 samples = 23 value = [11, 12] class = Disease
BP <= 148.0 gini = 0.496 samples = 35 value = [16, 19] class = Disease
cholestrol <= 261.5 gini = 0.214 samples = 41 value = [5, 36] class = Disease
gini = 0.0 samples = 30 value = [30, 0] class = No disease
gini = 0.444 samples = 3 value = [2, 1] class = No disease
BP <= 113.5 gini = 0.245 samples = 21 value = [18, 3] class = No disease
BP <= 109.0 gini = 0.487 samples = 43 value = [25, 18] class = No disease
BP <= 139.0 gini = 0.495 samples = 20 value = [11, 9] class = No disease
gini = 0.0 samples = 3 value = [0, 3] class = Disease
age <= 57.5 gini = 0.475 samples = 31 value = [12, 19] class = Disease
gini = 0.0 samples = 4 value = [4, 0] class = No disease
gini = 0.0 samples = 14 value = [0, 14] class = Disease
cholestrol <= 272.0 gini = 0.302 samples = 27 value = [5, 22] class = Disease
gini = 0.48 samples = 5 value = [3, 2] class = No disease
gini = 0.117 samples = 16 value = [15, 1] class = No disease
gini = 0.0 samples = 5 value = [5, 0] class = No disease
cholestrol <= 199.5 gini = 0.499 samples = 38 value = [20, 18] class = No disease
gini = 0.408 samples = 7 value = [2, 5] class = Disease
gini = 0.426 samples = 13 value = [9, 4] class = No disease
gini = 0.444 samples = 6 value = [4, 2] class = No disease
cholestrol <= 239.5 gini = 0.435 samples = 25 value = [8, 17] class = Disease
gini = 0.48 samples = 5 value = [3, 2] class = No disease
BP <= 122.5 gini = 0.165 samples = 22 value = [2, 20] class = Disease
gini = 0.0 samples = 4 value = [4, 0] class = No disease
BP <= 116.0 gini = 0.498 samples = 34 value = [16, 18] class = Disease
age <= 61.5 gini = 0.413 samples = 24 value = [7, 17] class = Disease
gini = 0.0 samples = 1 value = [1, 0] class = No disease
gini = 0.444 samples = 3 value = [2, 1] class = No disease
gini = 0.0 samples = 19 value = [0, 19] class = Disease
gini = 0.0 samples = 4 value = [0, 4] class = Disease
age <= 46.5 gini = 0.498 samples = 30 value = [16, 14] class = No disease
gini = 0.305 samples = 16 value = [3, 13] class = Disease
gini = 0.5 samples = 8 value = [4, 4] class = No disease
gini = 0.0 samples = 2 value = [0, 2] class = Disease
cholestrol <= 273.5 gini = 0.49 samples = 28 value = [16, 12] class = No disease
BP <= 139.0 gini = 0.463 samples = 22 value = [14, 8] class = No disease
gini = 0.444 samples = 6 value = [2, 4] class = Disease
gini = 0.391 samples = 15 value = [11, 4] class = No disease
gini = 0.49 samples = 7 value = [3, 4] class = Disease

[53]:
```
evaluate_model(dt4)
```

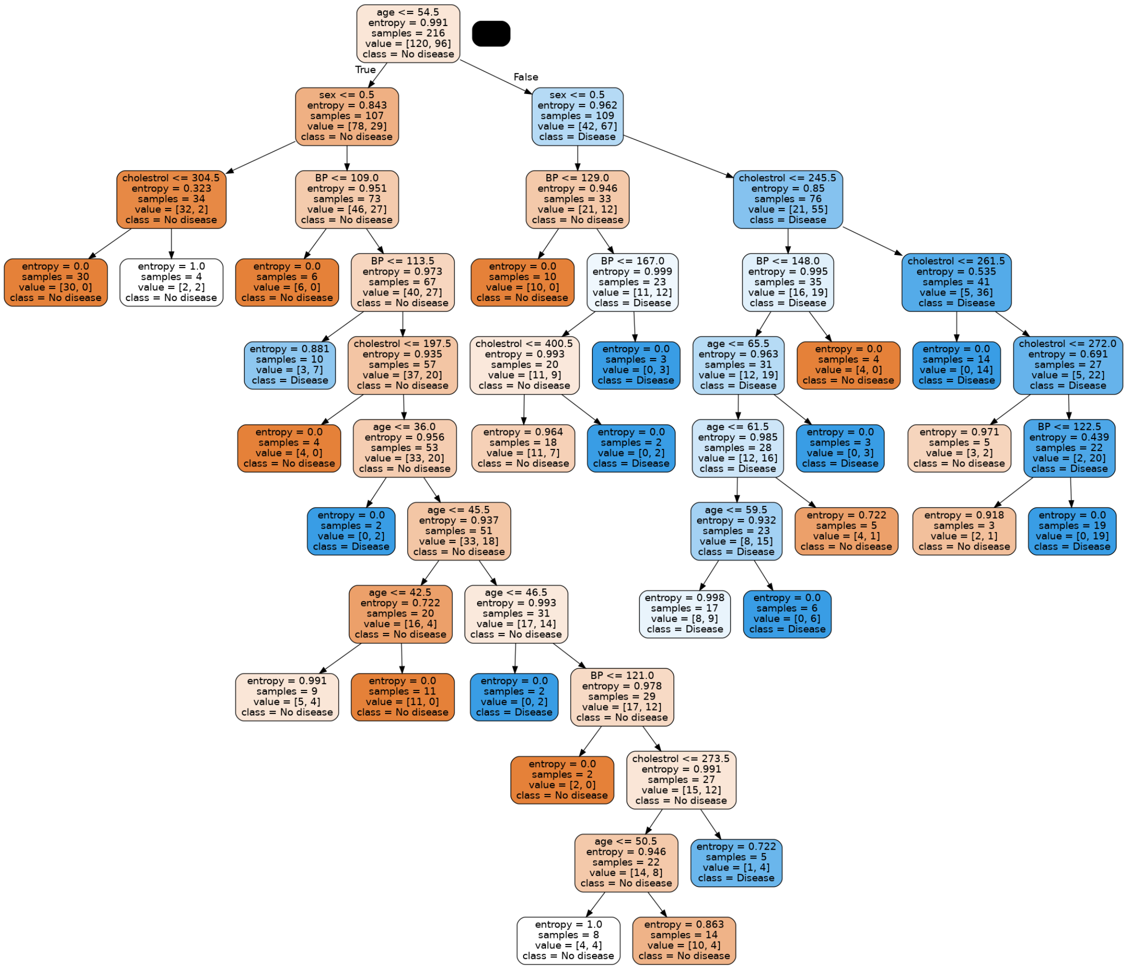
```
Train Accuracy Score 0.8425925925925926
----------------------------------------------------
Train data's Confusion Matrix
 [[107  13]
 [ 21  75]]
--------------------------------------------------
Test Accuracy Score 0.5740740740740741
----------------------------------------------------
Test data's Confusion Matrix
 [[24  6]
 [17  7]]
```

[56]:
```
dt5 = DecisionTreeClassifier(min_samples_split = 20, random_state = 42,␣
  ↪criterion = 'entropy')
dt5.fit(X_train, y_train)
```

[56]:
```
DecisionTreeClassifier(criterion='entropy', min_samples_split=20,
                       random_state=42)
```

[58]:
```
gph = get_dt_graph(dt5)
Image(gph.create_png())
```

[58]:


age <= 54.5 entropy = 0.991 samples = 216 value = [120, 96] class = No disease
[redacted]
True
False
sex <= 0.5 entropy = 0.843 samples = 107 value = [78, 29] class = No disease
sex <= 0.5 entropy = 0.962 samples = 109 value = [42, 67] class = Disease
cholestrol <= 304.5 entropy = 0.323 samples = 34 value = [32, 2] class = No disease
BP <= 109.0 entropy = 0.951 samples = 73 value = [46, 27] class = No disease
BP <= 129.0 entropy = 0.946 samples = 33 value = [21, 12] class = No disease
cholestrol <= 245.5 entropy = 0.85 samples = 76 value = [21, 55] class = Disease
entropy = 0.0 samples = 30 value = [30, 0] class = No disease
entropy = 1.0 samples = 4 value = [2, 2] class = No disease
entropy = 0.0 samples = 6 value = [6, 0] class = No disease
BP <= 113.5 entropy = 0.973 samples = 67 value = [40, 27] class = No disease
entropy = 0.0 samples = 10 value = [10, 0] class = No disease
BP <= 167.0 entropy = 0.999 samples = 23 value = [11, 12] class = Disease
BP <= 148.0 entropy = 0.995 samples = 35 value = [16, 19] class = Disease
cholestrol <= 261.5 entropy = 0.535 samples = 41 value = [5, 36] class = Disease
entropy = 0.881 samples = 10 value = [3, 7] class = Disease
cholestrol <= 197.5 entropy = 0.935 samples = 57 value = [37, 20] class = No disease
cholestrol <= 400.5 entropy = 0.993 samples = 20 value = [11, 9] class = No disease
entropy = 0.0 samples = 3 value = [0, 3] class = Disease
age <= 65.5 entropy = 0.963 samples = 31 value = [12, 19] class = Disease
entropy = 0.0 samples = 4 value = [4, 0] class = No disease
entropy = 0.0 samples = 14 value = [0, 14] class = Disease
cholestrol <= 272.0 entropy = 0.691 samples = 27 value = [5, 22] class = Disease
entropy = 0.0 samples = 4 value = [4, 0] class = No disease
age <= 36.0 entropy = 0.956 samples = 53 value = [33, 20] class = No disease
entropy = 0.964 samples = 18 value = [11, 7] class = No disease
entropy = 0.0 samples = 2 value = [0, 2] class = Disease
age <= 61.5 entropy = 0.985 samples = 28 value = [12, 16] class = Disease
entropy = 0.0 samples = 3 value = [0, 3] class = Disease
entropy = 0.971 samples = 5 value = [3, 2] class = No disease
BP <= 122.5 entropy = 0.439 samples = 22 value = [2, 20] class = Disease
entropy = 0.0 samples = 2 value = [0, 2] class = Disease
age <= 45.5 entropy = 0.937 samples = 51 value = [33, 18] class = No disease
age <= 59.5 entropy = 0.932 samples = 23 value = [8, 15] class = Disease
entropy = 0.722 samples = 5 value = [4, 1] class = No disease
entropy = 0.918 samples = 3 value = [2, 1] class = No disease
entropy = 0.0 samples = 19 value = [0, 19] class = Disease
age <= 42.5 entropy = 0.722 samples = 20 value = [16, 4] class = No disease
age <= 46.5 entropy = 0.993 samples = 31 value = [17, 14] class = No disease
entropy = 0.998 samples = 17 value = [8, 9] class = Disease
entropy = 0.0 samples = 6 value = [0, 6] class = Disease
entropy = 0.991 samples = 9 value = [5, 4] class = No disease
entropy = 0.0 samples = 11 value = [11, 0] class = No disease
entropy = 0.0 samples = 2 value = [0, 2] class = Disease
BP <= 121.0 entropy = 0.978 samples = 29 value = [17, 12] class = No disease
entropy = 0.0 samples = 2 value = [2, 0] class = No disease
cholestrol <= 273.5 entropy = 0.991 samples = 27 value = [15, 12] class = No disease
age <= 50.5 entropy = 0.946 samples = 22 value = [14, 8] class = No disease
entropy = 0.722 samples = 5 value = [1, 4] class = Disease
entropy = 1.0 samples = 8 value = [4, 4] class = No disease
entropy = 0.863 samples = 14 value = [10, 4] class = No disease

[59]:

```
evaluate_model(dt5)
```

```
Train Accuracy Score 0.8287037037037037
----------------------------------------------------
Train data's Confusion Matrix
 [[108  12]
 [ 25  71]]
--------------------------------------------------
Test Accuracy Score 0.6296296296296297
----------------------------------------------------
Test data's Confusion Matrix
 [[24  6]
 [14 10]]
```

[66]:

```
# hyper parameter tuning , creating the validation data set

from sklearn.model_selection import GridSearchCV
```

```
params = {'max_depth': [2,4,6,8,10,12,50,100],'min_samples_leaf':
 ↪[5,10,15,20,25,30,35,40],'criterion':['gini','entropy']}
```

```
[68]: grid_search = GridSearchCV(estimator = dt, param_grid = params, cv = 5, n_jobs␣
       ↪= -1, verbose = 1, scoring = 'accuracy')
      grid_search.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 128 candidates, totalling 640 fits
```

```
[68]: GridSearchCV(cv=5, estimator=DecisionTreeClassifier(random_state=42), n_jobs=-1,
                   param_grid={'criterion': ['gini', 'entropy'],
                               'max_depth': [2, 4, 6, 8, 10, 12, 50, 100],
                               'min_samples_leaf': [5, 10, 15, 20, 25, 30, 35, 40]},
                   scoring='accuracy', verbose=1)
```

```
[70]: score_df = pd.DataFrame(grid_search.cv_results_)
      score_df.head()
```

```
[70]:    mean_fit_time  std_fit_time  mean_score_time  std_score_time  \
      0       0.004720      0.000388         0.003100        0.000344
      1       0.003818      0.000517         0.002578        0.000268
      2       0.003519      0.000460         0.002110        0.000297
      3       0.003828      0.000730         0.002209        0.000101
      4       0.003579      0.000457         0.002313        0.000334

        param_criterion param_max_depth param_min_samples_leaf  \
      0            gini               2                      5
      1            gini               2                     10
      2            gini               2                     15
      3            gini               2                     20
      4            gini               2                     25

                                                    params  split0_test_score  \
      0  {'criterion': 'gini', 'max_depth': 2, 'min_sam…           0.659091
      1  {'criterion': 'gini', 'max_depth': 2, 'min_sam…           0.659091
      2  {'criterion': 'gini', 'max_depth': 2, 'min_sam…           0.659091
      3  {'criterion': 'gini', 'max_depth': 2, 'min_sam…           0.659091
      4  {'criterion': 'gini', 'max_depth': 2, 'min_sam…           0.772727

         split1_test_score  split2_test_score  split3_test_score  split4_test_score  \
      0           0.627907           0.744186           0.674419           0.651163
      1           0.627907           0.744186           0.674419           0.627907
      2           0.627907           0.744186           0.674419           0.627907
      3           0.627907           0.744186           0.674419           0.627907
      4           0.627907           0.744186           0.674419           0.627907

         mean_test_score  std_test_score  rank_test_score
```

```
0         0.671353        0.039394               33
1         0.666702        0.042735               50
2         0.666702        0.042735               50
3         0.666702        0.042735               50
4         0.689429        0.059552                1
```

[72]:

```
score_df.shape
```

[72]:

```
(128, 16)
```

[74]:

```
score_df.nlargest(5, "mean_test_score")
```

[74]:

```
    mean_fit_time  std_fit_time  mean_score_time  std_score_time  \
4        0.003579      0.000457         0.002313        0.000334
12       0.003464      0.000664         0.002302        0.000146
20       0.004924      0.001540         0.002535        0.000197
28       0.004347      0.002339         0.002274        0.000101
36       0.003034      0.000081         0.002308        0.000249

   param_criterion param_max_depth param_min_samples_leaf  \
4             gini               2                     25
12            gini               4                     25
20            gini               6                     25
28            gini               8                     25
36            gini              10                     25

                                               params  split0_test_score  \
4   {'criterion': 'gini', 'max_depth': 2, 'min_sam…           0.772727
12  {'criterion': 'gini', 'max_depth': 4, 'min_sam…           0.772727
20  {'criterion': 'gini', 'max_depth': 6, 'min_sam…           0.772727
28  {'criterion': 'gini', 'max_depth': 8, 'min_sam…           0.772727
36  {'criterion': 'gini', 'max_depth': 10, 'min_sa…           0.772727

    split1_test_score  split2_test_score  split3_test_score  \
4            0.627907           0.744186           0.674419
12           0.627907           0.744186           0.674419
20           0.627907           0.744186           0.674419
28           0.627907           0.744186           0.674419
36           0.627907           0.744186           0.674419

    split4_test_score  mean_test_score  std_test_score  rank_test_score
4            0.627907         0.689429        0.059552                1
12           0.627907         0.689429        0.059552                1
20           0.627907         0.689429        0.059552                1
28           0.627907         0.689429        0.059552                1
36           0.627907         0.689429        0.059552                1
```

[76]:
```
grid_search.best_estimator_
```

[76]:
```
DecisionTreeClassifier(max_depth=2, min_samples_leaf=25, random_state=42)
```

[78]:
```
dt_best = grid_search.best_estimator_
```

[80]:
```
evaluate_model(dt_best)
```

```
Train Accuracy Score 0.7129629629629629
----------------------------------------------------
Train data's Confusion Matrix
 [[99 21]
 [41 55]]
--------------------------------------------------
Test Accuracy Score 0.5740740740740741
----------------------------------------------------
Test data's Confusion Matrix
 [[24  6]
 [17  7]]
```

[82]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, dt_best.predict(X_test)))
```

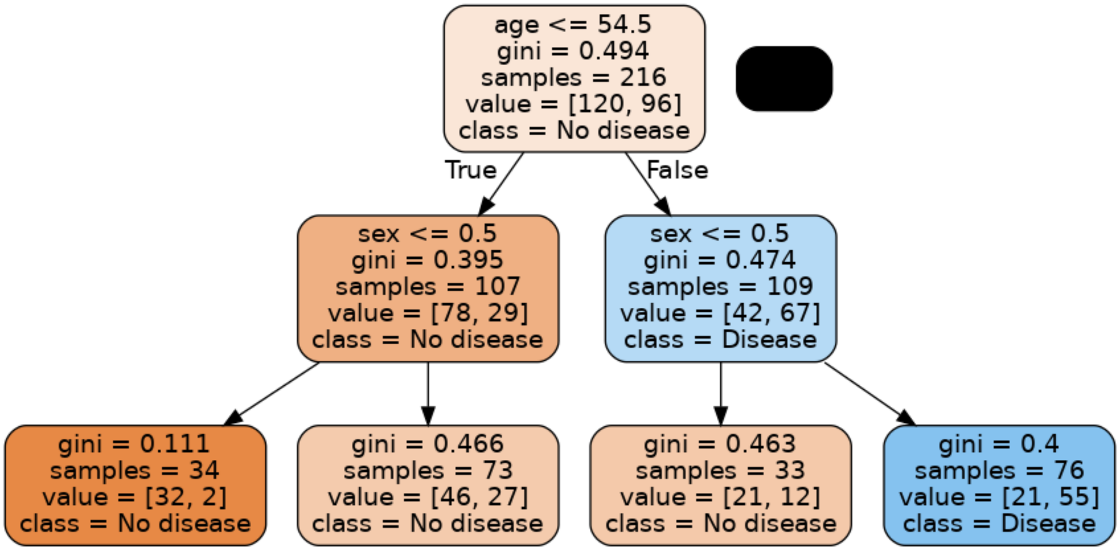
```
              precision    recall  f1-score   support

           0       0.59      0.80      0.68        30
           1       0.54      0.29      0.38        24

    accuracy                           0.57        54
   macro avg       0.56      0.55      0.53        54
weighted avg       0.56      0.57      0.54        54
```

[84]:
```
gph = get_dt_graph(dt_best)
Image(gph.create_png())
```

[84]:


age <= 54.5
gini = 0.494
samples = 216
value = [120, 96]
class = No disease
True
False
sex <= 0.5
gini = 0.395
samples = 107
value = [78, 29]
class = No disease
sex <= 0.5
gini = 0.474
samples = 109
value = [42, 67]
class = Disease
gini = 0.111
samples = 34
value = [32, 2]
class = No disease
gini = 0.466
samples = 73
value = [46, 27]
class = No disease
gini = 0.463
samples = 33
value = [21, 12]
class = No disease
gini = 0.4
samples = 76
value = [21, 55]
class = Disease

```
[ ]:
```