In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
from sklearn.datasets import make_classification
```

In [4]:
```
X,y=make_classification(n_samples=1000,n_features=5,n_clusters_per_class=1,n_clas
```

In [5]:
```
X[0:5]
```

Out[5]:
```
array([[ 1.54701705,  0.84770596, -0.41725021, -0.62356778, -0.19388577],
       [ 0.80633556,  0.40985594, -0.45641095, -0.3052022 ,  0.50935923],
       [ 0.94390268,  0.70041038,  1.11385452, -0.49394417,  1.42305455],
       [ 1.92091517,  0.95815739, -1.2235022 , -0.71578154,  0.66588981],
       [ 1.45270369,  0.69035375, -1.18119669, -0.52009219, -0.22745417]])
```

In [6]:
```
y[0:5]
```

Out[6]:
```
array([0, 0, 1, 0, 0])
```

In [7]:
```
X.shape,y.shape
```

Out[7]:
```
((1000, 5), (1000,))
```

In [8]:
```
#get train test split
```

In [9]:
```
from sklearn.model_selection import train_test_split
```

In [10]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=252
```

In [11]:
```
X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

Out[11]:
```
((700, 5), (300, 5), (700,), (300,))
```

In [12]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [13]:
```
model=DecisionTreeClassifier()
```

In [14]:
```
model.fit(X_train,y_train)
```

Out[14]:
```
DecisionTreeClassifier()
```

In [15]:
```
#get model predictions
```

In [16]:
```
y_pred=model.predict(X_test)
```

In [17]:
```
y_pred.shape
```

Out[17]:
```
(300,)
```

In [18]:
```
y_pred
```

Out[18]:
```
array([1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1,
       0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1,
       0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1,
       1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0,
       1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1])
```

In [19]:
```
from sklearn.metrics import confusion_matrix,classification_report,accuracy_score
```

In [20]:
```
confusion_matrix(y_pred,y_test)
```

Out[20]:
```
array([[155,   3],
       [  2, 140]], dtype=int64)
```

In [21]:
```
accuracy_score(y_pred,y_test)
```

Out[21]:
```
0.9833333333333333
```

In [22]:
```
print(classification_report(y_pred,y_test))
```

```
              precision    recall  f1-score   support

           0       0.99      0.98      0.98       158
           1       0.98      0.99      0.98       142

    accuracy                           0.98       300
   macro avg       0.98      0.98      0.98       300
weighted avg       0.98      0.98      0.98       300
```

In [23]:
```
r2_score(y_pred,y_test)
```

Out[23]:
```
0.933143162774113
```

In [24]:
```
from sklearn.model_selection import GridSearchCV
```

In [25]:
```
parameters={'criterion':['gini','entropy'],'max_depth':[2,3,4,5,6,7,8,9,10,11,12,
gridsearch=GridSearchCV(DecisionTreeClassifier(),parameters)
gridsearch.fit(X_train,y_train)
```

Out[25]:
```
GridSearchCV(estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 15,
                                       20, 30, 40, 50, 70, 90, 120, 150]})
```

In [26]:
```
gridsearch.best_params_
```

Out[26]:
```
{'criterion': 'entropy', 'max_depth': 3}
```

In [27]:
```
gridsearch.best_score_
```

Out[27]:
```
0.9885714285714287
```

In [28]:
```
gridsearch.best_estimator_
```

Out[28]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=3)
```

In [29]:
```
gridsearch.best_index_
```

Out[29]:
```
21
```

In [30]:
```
y_pred_grid=gridsearch.predict(X_test)
```

In [31]:
```
confusion_matrix(y_test,y_pred_grid)
```

Out[31]:
```
array([[156,   1],
       [  1, 142]], dtype=int64)
```

In [32]:
```
print(classification_report(y_test,y_pred_grid))
```

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       157
           1       0.99      0.99      0.99       143

    accuracy                           0.99       300
   macro avg       0.99      0.99      0.99       300
weighted avg       0.99      0.99      0.99       300
```

In [33]:
```
#https://www.youtube.com/watch?v=w4frwjt8uCo
```

In [ ]:

In [ ]:

In [ ]:

## P4.b **2. Random Forest Classification with Artificial Generated Dataset**

In [ ]:

In [34]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [35]:
```
from sklearn.datasets import make_classification
```

In [36]:
```
X,y=make_classification(n_samples=1000,n_features=5,n_clusters_per_class=1,n_clas
```

In [37]:
```
X[0:5]
```

Out[37]:
```
array([[ 1.54701705,  0.84770596, -0.41725021, -0.62356778, -0.19388577],
       [ 0.80633556,  0.40985594, -0.45641095, -0.3052022 ,  0.50935923],
       [ 0.94390268,  0.70041038,  1.11385452, -0.49394417,  1.42305455],
       [ 1.92091517,  0.95815739, -1.2235022 , -0.71578154,  0.66588981],
       [ 1.45270369,  0.69035375, -1.18119669, -0.52009219, -0.22745417]])
```

In [38]:
```
y[0:5]
```

Out[38]:
```
array([0, 0, 1, 0, 0])
```

In [39]:
```
X.shape,y.shape
```

Out[39]:
```
((1000, 5), (1000,))
```

In [40]:
```
from sklearn.model_selection import train_test_split
```

In [41]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,train_size=0.7,random_state=25
```

In [42]:
```
X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

Out[42]:
```
((700, 5), (300, 5), (700,), (300,))
```

In [43]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [44]:
```
model=RandomForestClassifier()
```

In [45]:
```
model.fit(X_train,y_train)
```

Out[45]:
```
RandomForestClassifier()
```

In [46]:
```
y_pred=model.predict(X_test)
```

In [47]:
```
y_pred.shape
```

Out[47]:
```
(300,)
```

In [48]:
```
y_pred
```

Out[48]:
```
array([1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1,
       0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1,
       1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0,
       1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1])
```

In [49]:

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
```

In [50]:

```
accuracy_score(y_test,y_pred)
```

Out[50]:

```
0.99
```

In [51]:

```
confusion_matrix(y_test,y_pred)
```

Out[51]:

```
array([[156,   1],
       [  2, 141]], dtype=int64)
```

In [52]:

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       157
           1       0.99      0.99      0.99       143

    accuracy                           0.99       300
   macro avg       0.99      0.99      0.99       300
weighted avg       0.99      0.99      0.99       300
```

In [56]:

```
from sklearn.model_selection import GridSearchCV
parameters={'n_estimators':[10,20,30,100,200,500],'max_features':['auto','sqrt'],
gridsearch=GridSearchCV(RandomForestClassifier(),parameters)
gridsearch.fit(X_train,y_train)
```

Out[56]:

```
GridSearchCV(estimator=RandomForestClassifier(),
             param_grid={'bootstrap': [True, False],
                         'max_features': ['auto', 'sqrt'],
                         'min_samples_split': [4, 8],
                         'n_estimators': [10, 20, 30, 100, 200, 500]})
```

In [54]:

```
gridsearch.best_params_
```

Out[54]:

```
{'bootstrap': True,
 'max_features': 'sqrt',
 'min_samples_split': 4,
 'n_estimators': 10}
```

In [55]:

```
gridsearch.best_estimator_
```

Out[55]:

```
RandomForestClassifier(max_features='sqrt', min_samples_split=4,
                       n_estimators=10)
```

In [57]:

```
gridsearch.best_index_
```

Out[57]:

```
25
```

In [58]:

```
y_pred_grid=gridsearch.predict(X_test)
```

In [64]:

```
confusion_matrix(y_test,y_pred_grid)
```

Out[64]:

```
array([[156,   1],
       [  2, 141]])
```

In [65]:

```
print(classification_report(y_test,y_pred_grid))
```

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       157
           1       0.99      0.99      0.99       143

    accuracy                           0.99       300
   macro avg       0.99      0.99      0.99       300
weighted avg       0.99      0.99      0.99       300
```

In [66]:

```
print(classification_report(y_pred,y_pred_grid))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       158
           1       1.00      1.00      1.00       142

    accuracy                           1.00       300
   macro avg       1.00      1.00      1.00       300
weighted avg       1.00      1.00      1.00       300
```