In [ ]:

```
'''

Decision Tree is a Supervised learning technique that can be used for both classificati
but mostly it is preferred for solving Classification problems. It is a tree-structurec
where internal nodes represent the features of a dataset,
branches represent the decision rules and each leaf node represents the outcome

Attribute Selection Measures
While implementing a Decision tree,
the main issue arises that how to select the best attribute for the root node and for s
So, to solve such problems there is a technique which is called as Attribute selection
By this measurement, we can easily select the best attribute for the nodes of the tree.
There are two popular techniques for ASM, which are:

1.Information Gain
2.Gini Index
'''
```

In [23]:

```
from sklearn.datasets import load_iris
```

In [24]:

```
X,y = load_iris(return_X_y=True)
```

In [25]:

```
X # input features
y # target output
```

Out[25]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [26]:

```
from sklearn.tree import DecisionTreeClassifier
```

In [30]:

```
# clf = DecisionTreeClassifier(criterion='gini',max_depth=5)
```

In [82]:

```
clf = DecisionTreeClassifier(criterion='entropy',max_depth=5)
```

In [94]:

```python
from sklearn.model_selection import train_test_split
# for splitting training data and testing data from dataset
```

In [95]:

```python
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=0)
```

In [96]:

```python
clf.fit(x_train,y_train) # model training
```

Out[96]:

```
DecisionTreeClassifier(criterion='entropy', max_depth=5)
```

In [110]:

```python
y_test # acutual testing output
```

Out[110]:

```
array([2, 1, 0, 2, 0, 2, 0, 1, 1, 1, 2, 1, 1, 1, 1, 0, 1, 1, 0, 0, 2, 1,
       0, 0, 2, 0, 0, 1, 1, 0, 2, 1, 0, 2, 2, 1, 0, 1, 1, 1, 2, 0, 2, 0,
       0])
```

In [114]:

```python
y_pred = clf.predict(x_test)
y_pred # our model predicted output
```

Out[114]:

```
array([2, 1, 0, 2, 0, 2, 0, 1, 1, 1, 2, 1, 1, 1, 1, 0, 1, 1, 0, 0, 2, 1,
       0, 0, 2, 0, 0, 1, 1, 0, 2, 1, 0, 2, 2, 1, 0, 2, 1, 1, 2, 0, 2, 0,
       0])
```

In [99]:

```python
# Check model accuracy

acc = clf.score(x_test,y_test)

acc = acc*100
acc = "{:.2f}".format(acc)

print('Accuracy : ',acc,'%')
```

```
Accuracy :  97.78 %
```

In [100]:

```
# confusuion matrix

from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test,y_pred)
print('Confusion matrix \n',cm)
```

```
Confusion matrix
 [[16  0  0]
 [ 0 17  1]
 [ 0  0 11]]
```

In [101]:

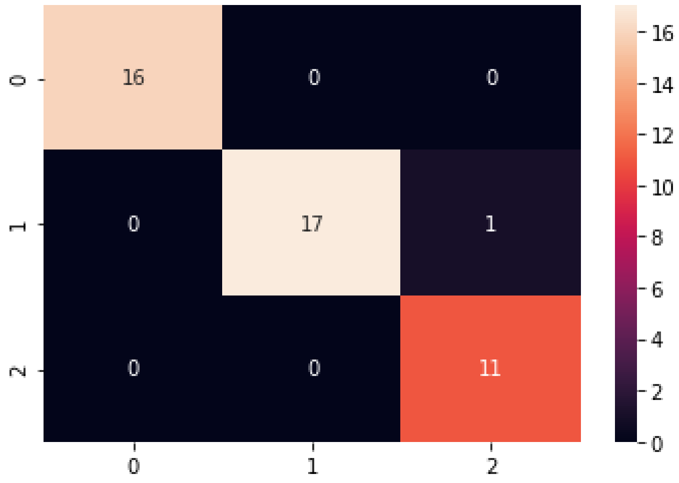
```
# seaborn heatmap

import seaborn as sns

sns.heatmap(cm,annot=True)
sns
```

Out[101]:

```
<module 'seaborn' from 'C:\\Users\\UmarKhan pathan\\anaconda3\\lib\\site-pac
kages\\seaborn\\__init__.py'>
```



In [102]:

```
# accuracy score

from sklearn.metrics import accuracy_score

score = accuracy_score(y_test,y_pred)
score = score*100

score = "{:.2f}".format(score)

print('Accuracy score : ',score,'%')
```

```
Accuracy score :  97.78 %
```

In [103]:

```
'''

Classification report
to display the model precision , recall , f1-score and support

'''
```

Out[103]:

```
'\n\nClassification report \nto display the model precision , recall , f1-sc
ore and support\n\n'
```

In [104]:

```
# importing  Classification report from sklearn.metrics

from sklearn.metrics import classification_report

cr = classification_report(y_test,y_pred)
print('Classification report: \n',cr)
```

```
Classification report: 
               precision    recall  f1-score   support

           0       1.00      1.00      1.00        16
           1       1.00      0.94      0.97        18
           2       0.92      1.00      0.96        11

    accuracy                           0.98        45
   macro avg       0.97      0.98      0.98        45
weighted avg       0.98      0.98      0.98        45
```

In [ ]:

```

```

In [ ]:

```

```