```
NAME: NIVETHA M

ROLL NO.:215229126
```

PML- LAB-8: Animal Classification using Decision Trees

Step1 : Create Datasets

In [1]:
```
import pandas as pd
```

In [2]:
```
df = pd.read_csv("animal.csv")
```

In [3]:
```
df
```

Out[3]:

	Toothed	hair	breathes	legs	species
0	True	True	True	True	Mammal
1	True	True	True	True	Mammal
2	True	False	True	False	Reptile
3	False	True	True	True	Mammal
4	True	True	True	True	Mammal
5	True	True	True	True	Mammal
6	True	False	False	False	Reptile
7	True	False	True	False	Reptile
8	True	True	True	True	Mammal
9	False	False	True	True	Reptile

In [4]:
```
X = df.drop(['species'],axis=1)
y = df['species']
```

In [5]:
```
from sklearn.model_selection import train_test_split
```

In [6]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.33)
```

Step2:Model Building using ID3

In [7]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [8]:
```
clf_entropy = DecisionTreeClassifier(criterion = "entropy")
```

In [9]:
```
clf_entropy.fit(X_train,y_train)
```

Out[9]:
```
DecisionTreeClassifier(criterion='entropy')
```

In [10]:
```
y_pred= clf_entropy.predict(X_test)
```

```
In [11]: from sklearn.metrics import accuracy_score,classification_report
```

```
In [12]: acc = accuracy_score(y_test,y_pred)
```

```
In [13]: acc
```

```
Out[13]: 1.0
```

```
In [14]: print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

      Mammal       1.00      1.00      1.00         2
     Reptile       1.00      1.00      1.00         2

    accuracy                           1.00         4
   macro avg       1.00      1.00      1.00         4
weighted avg       1.00      1.00      1.00         4
```

```
In [15]: from sklearn import tree
```

```
In [16]: with open("tree1.dot",'w')as f:
             f = tree.export_graphviz(clf_entropy,out_file=f,max_depth=4,
                                      impurity = False,feature_names = X.columns.values,
                                      class_names = ['Reptile','Mammal'],
                                      filled = True)
```

```
In [17]: !type tree1.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 6\nvalue = [4, 2]\nclass = Reptile", fillcolo
r="#f2c09c"] ;
1 [label="samples = 2\nvalue = [0, 2]\nclass = Mammal", fillcolor="#399de5"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 4\nvalue = [4, 0]\nclass = Reptile", fillcolor="#e58139"]
;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

Step3: Create a Test Set

```
In [20]: X_tests = pd.read_csv("animals_test.csv")
```

Step4: Perform prediction

```
In [21]: y_pred_1 = clf_entropy.predict(X_tests)
```

```
In [22]: y_pred_1
```

```
Out[22]: array(['Reptile', 'Mammal', 'Reptile'], dtype=object)
```

Step5: Build CART Decision Tree Model

In [23]:
```
cart_entropy = DecisionTreeClassifier(criterion = "gini")
```

In [24]:
```
cart_entropy.fit(X_train,y_train)
```

Out[24]:
```
DecisionTreeClassifier()
```

In [25]:
```
y_pred_cart = cart_entropy.predict(X_test)
```

In [26]:
```
acc_cart =  accuracy_score(y_test,y_pred_cart)
```

In [27]:
```
acc_cart
```

Out[27]:
```
1.0
```

In [28]:
```
print(classification_report(y_test,y_pred_cart))
```

```
              precision    recall  f1-score   support

      Mammal       1.00      1.00      1.00         2
     Reptile       1.00      1.00      1.00         2

    accuracy                           1.00         4
   macro avg       1.00      1.00      1.00         4
weighted avg       1.00      1.00      1.00         4
```

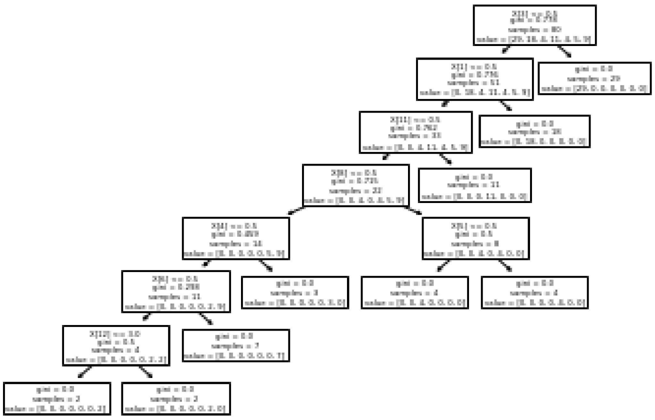
In [29]:
```
with open("tree2.dot",'w')as f:
    f = tree.export_graphviz(cart_entropy,out_file=f,max_depth=4,
                            impurity = False,feature_names = X.columns.values,
                            class_names = ['Reptile','Mammal'],
                            filled = True)
```

In [30]:
```
!type tree2.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 6\nvalue = [4, 2]\nclass = Reptile", fillcolo
r="#f2c09c"] ;
1 [label="samples = 2\nvalue = [0, 2]\nclass = Mammal", fillcolor="#399de5"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 4\nvalue = [4, 0]\nclass = Reptile", fillcolor="#e58139"]
;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

In [44]:
```
tree.plot_tree(cart_entropy)
```

Out[44]:
```
[Text(273.92727272727274, 203.85, 'X[3] <= 0.5\ngini = 0.778\nsamples = 80\nva
lue = [29, 18, 4, 11, 4, 5, 9]'),
 Text(243.4909090909091, 176.67000000000002, 'X[1] <= 0.5\ngini = 0.776\nsampl
es = 51\nvalue = [0, 18, 4, 11, 4, 5, 9]'),
 Text(213.05454545454546, 149.49, 'X[11] <= 0.5\ngini = 0.762\nsamples = 33\nv
alue = [0, 0, 4, 11, 4, 5, 9]'),
 Text(182.61818181818182, 122.31, 'X[8] <= 0.5\ngini = 0.715\nsamples = 22\nva
lue = [0, 0, 4, 0, 4, 5, 9]'),
 Text(121.74545454545455, 95.13, 'X[4] <= 0.5\ngini = 0.459\nsamples = 14\nval
ue = [0, 0, 0, 0, 0, 5, 9]'),
 Text(91.30909090909091, 67.94999999999999, 'X[6] <= 0.5\ngini = 0.298\nsample
s = 11\nvalue = [0, 0, 0, 0, 0, 2, 9]'),
 Text(60.872727272727275, 40.77000000000001, 'X[12] <= 3.0\ngini = 0.5\nsample
s = 4\nvalue = [0, 0, 0, 0, 0, 2, 2]'),
 Text(30.436363636363637, 13.590000000000003, 'gini = 0.0\nsamples = 2\nvalue
= [0, 0, 0, 0, 0, 0, 2]'),
 Text(91.30909090909091, 13.590000000000003, 'gini = 0.0\nsamples = 2\nvalue =
[0, 0, 0, 0, 0, 2, 0]'),
 Text(121.74545454545455, 40.77000000000001, 'gini = 0.0\nsamples = 7\nvalue =
[0, 0, 0, 0, 0, 0, 7]'),
 Text(152.1818181818182, 67.94999999999999, 'gini = 0.0\nsamples = 3\nvalue =
[0, 0, 0, 0, 0, 3, 0]'),
 Text(243.4909090909091, 95.13, 'X[5] <= 0.5\ngini = 0.5\nsamples = 8\nvalue =
[0, 0, 4, 0, 4, 0, 0]'),
 Text(213.05454545454546, 67.94999999999999, 'gini = 0.0\nsamples = 4\nvalue =
[0, 0, 4, 0, 0, 0, 0]'),
 Text(273.92727272727274, 67.94999999999999, 'gini = 0.0\nsamples = 4\nvalue =
[0, 0, 0, 0, 4, 0, 0]'),
 Text(243.4909090909091, 122.31, 'gini = 0.0\nsamples = 11\nvalue = [0, 0, 0,
11, 0, 0, 0]'),
 Text(273.92727272727274, 149.49, 'gini = 0.0\nsamples = 18\nvalue = [0, 18,
0, 0, 0, 0, 0]'),
 Text(304.3636363636364, 176.67000000000002, 'gini = 0.0\nsamples = 29\nvalue
= [29, 0, 0, 0, 0, 0, 0]')]
```



Step6: Build DT with Zoo Dataset

In [31]:
```
zoo = pd.read_csv('zoo.data',names=['animals',1,2,3,4,5,6,7,8,9,10,11,12,13,14,
```

In [32]:
```
zoo.head()
```

Out[32]:

	animals	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	target
0	aardvark	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	1
1	antelope	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	1
2	bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	4
3	bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	1
4	boar	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	1	1

In [33]:
```
X1=zoo.drop(['animals','target'],axis=1)
y1=zoo[['target']]
```

In [34]:
```
X_trainz, X_testz, y_trainz, y_testz = train_test_split(X1,y1,test_size=0.2,ranc
```

Creating a model

In [35]:
```
clf_entropy.fit(X_trainz,y_trainz)
```

Out[35]:
```
DecisionTreeClassifier(criterion='entropy')
```

In [36]:
```
y_pred_z = clf_entropy.predict(X_testz)
```

CART Model

In [37]:
```
cart_entropy.fit(X_trainz,y_trainz)
```

Out[37]:
```
DecisionTreeClassifier()
```

In [38]:
```
y_pred_c = cart_entropy.predict(X_testz)
```

Accuracy score

In [39]:
```
acc_z = accuracy_score(y_pred_z,y_testz)
acc_z
```

Out[39]:
```
0.9523809523809523
```

In [40]:
```
acc_c = accuracy_score(y_pred_c,y_testz)
acc_c
```

Out[40]:
```
0.9523809523809523
```

Classification Report

In [41]:

```
import warnings
warnings.filterwarnings("ignore")
```

In [42]:

```
print("Classification_report for Entropy Equation :\n",classification_report(y_t
```

```
Classification_report for Entropy Equation :
               precision    recall  f1-score   support

           1       1.00      1.00      1.00        12
           2       1.00      1.00      1.00         2
           3       0.00      0.00      0.00         1
           4       0.67      1.00      0.80         2
           6       1.00      1.00      1.00         3
           7       1.00      1.00      1.00         1

    accuracy                           0.95        21
   macro avg       0.78      0.83      0.80        21
weighted avg       0.92      0.95      0.93        21
```

In [43]:

```
print("Classification_report for CART :\n",classification_report(y_testz,y_pred_
```

```
Classification_report for CART :
               precision    recall  f1-score   support

           1       1.00      1.00      1.00        12
           2       1.00      1.00      1.00         2
           3       0.00      0.00      0.00         1
           4       1.00      1.00      1.00         2
           5       0.00      0.00      0.00         0
           6       1.00      1.00      1.00         3
           7       1.00      1.00      1.00         1

    accuracy                           0.95        21
   macro avg       0.71      0.71      0.71        21
weighted avg       0.95      0.95      0.95        21
```