SURUTHI S

225229141

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

In [2]:
```
df  = pd.read_csv("animals.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

STEP 2

In [4]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [15]:
```
feat = df.loc[:,'toothed':'legs']
label = df['species'].values
```

In [12]:
```
from sklearn.model_selection import train_test_split
```

In [28]:
```
xtrain,xtest,ytrain,ytest = train_test_split(feat,label,test_size=0.30,random
```

In [29]:
```
id3_tree = DecisionTreeClassifier(criterion='entropy')
```

In [30]:
```
id3_tree.fit(xtrain,ytrain)
```

Out[30]:
```
DecisionTreeClassifier(criterion='entropy')
```

In [31]:
```
ypred = id3_tree.predict(xtest)
```

In [32]:
```
from sklearn.metrics import classification_report

print(classification_report(ytest,ypred))
```

```
              precision    recall  f1-score   support

      Mammal       1.00      1.00      1.00         3

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```

accuracy = 1

```
https://scikit-learn.org/stable/modules/generated/sklearn.tree.export_graphviz.html
```

In [36]:
```
! pip install graphviz
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting graphviz
  Downloading graphviz-0.19.1-py3-none-any.whl (46 kB)
Installing collected packages: graphviz
Successfully installed graphviz-0.19.1
```

In [42]:
```
from sklearn.tree import export_graphviz
```

In [44]:
```
with open("tree1.dot",'w') as f:
    f= export_graphviz(id3_tree,out_file=f,
                               max_depth=4,
                               impurity= False,
                               feature_names = feat.columns.values,
                               class_names=['Reptile','Mammal'],
                               filled=True)
```

In [47]:
```
!type tree1.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 7\nvalue = [3, 4]\nclass = Mammal", fillcolor
="#cee6f8"] ;
1 [label="samples = 4\nvalue = [0, 4]\nclass = Mammal", fillcolor="#399de5"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 3\nvalue = [3, 0]\nclass = Reptile", fillcolor="#e58139"] ;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

In [49]:
```
from sklearn import tree
```

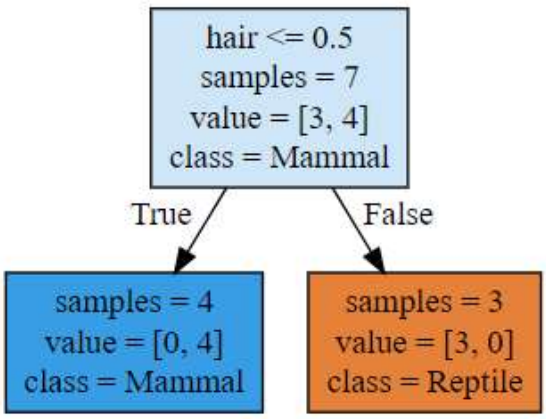
In [50]:
```
1 tree.plot_tree(id3_tree)
```

Out[50]:
```
[Text(248.0, 277.2, 'X[1] <= 0.5\nentropy = 0.985\nsamples = 7\nvalue = [3, 4]'),
 Text(124.0, 92.39999999999998, 'entropy = 0.0\nsamples = 4\nvalue = [0, 4]'),
 Text(372.0, 92.39999999999998, 'entropy = 0.0\nsamples = 3\nvalue = [3, 0]')]
```

Sample 1 | Sample 2 | Sample 3 | Sample 4 | Sample 5

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 7\nvalue = [3, 4]\nclass = Mammal", fillcolor="#ce
1 [label="samples = 4\nvalue = [0, 4]\nclass = Mammal", fillcolor="#399de5"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 3\nvalue = [3, 0]\nclass = Reptile", fillcolor="#e58139"] ;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

Generate Graph!



STEP 3

In [51]:
```
1 test =pd.read_csv("animals_test.csv")
```

In [52]:
```
1 test
```

Out[52]:

	toothed	hair	breathes	legs	species
0	False	False	True	False	Reptile
1	False	True	True	True	Mammal
2	True	False	True	True	Reptile

STEP 4

In [54]:
```
1 feat_test = test.loc[:,'toothed':'legs']
2 labels_test = test['species']
```

In [57]:
```
1 ypred_test = id3_tree.predict(feat_test)
```

In [58]:
```
1 print(classification_report(ypred_test,labels_test))
```

```
              precision    recall  f1-score   support

      Mammal       1.00      1.00      1.00         1
     Reptile       1.00      1.00      1.00         2

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```

ACCURACY =1

STEP 5

In [59]:
```
1 gini_tree = DecisionTreeClassifier(criterion='gini')
```

In [60]:
```
1 gini_tree.fit(feat,label)
```

Out[60]:
```
DecisionTreeClassifier()
```

In [61]:
```
1 ypred_gini = gini_tree.predict(feat_test)
```

In [62]:
```
print(classification_report(ypred_gini,labels_test))
```

```
              precision    recall  f1-score   support

      Mammal       1.00      1.00      1.00         1
     Reptile       1.00      1.00      1.00         2

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```

ACCURACY =1

In [64]:
```
with open("tree2.dot",'w') as f:
    f= export_graphviz(gini_tree,out_file=f,
                               max_depth=4,
                               impurity= False,
                               feature_names = feat.columns.values,
                               class_names=['Reptile','Mammal'],
                               filled=True)
```

In [65]:
```
!type tree1.dot
```

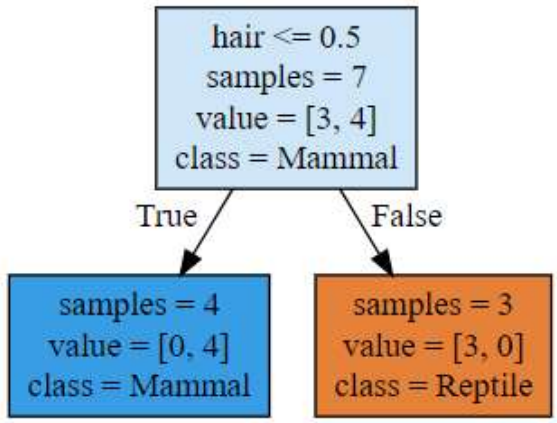
```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 7\nvalue = [3, 4]\nclass = Mammal", fillcolor
="#cee6f8"] ;
1 [label="samples = 4\nvalue = [0, 4]\nclass = Mammal", fillcolor="#399de5"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 3\nvalue = [3, 0]\nclass = Reptile", fillcolor="#e58139"] ;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

(Your Graphviz data is private and never harvested)

Sample 1 | Sample 2 | Sample 3 | Sample 4 | Sample 5

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 7\nvalue = [3, 4]\nclass = Mammal", fillcolor="#ce
1 [label="samples = 4\nvalue = [0, 4]\nclass = Mammal", fillcolor="#399de5"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 3\nvalue = [3, 0]\nclass = Reptile", fillcolor="#e58139"] ;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

Generate Graph!



STEP 6

In [91]:
```
names = ['animal name','hair','feathers','eggs','milk','airborne','aquatic','
        'fins','legs','tail','domestic','catsize','type']
df = pd.read_csv("zoo.csv",header=None,names=names)
df
```

Out[91]:

	animal name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes
0	aardvark	1	0	0	1	0	0	1	1	1	1
1	antelope	1	0	0	1	0	0	0	1	1	1
2	bass	0	0	1	0	0	1	1	1	1	0
3	bear	1	0	0	1	0	0	1	1	1	1
4	boar	1	0	0	1	0	0	1	1	1	1
...	...	...	...	...	...	...	...	...	...	...	..
96	wallaby	1	0	0	1	0	0	0	1	1	1
97	wasp	1	0	1	0	1	0	0	0	0	1
98	wolf	1	0	0	1	0	0	1	1	1	1
99	worm	0	0	1	0	0	0	0	0	0	1
100	wren	0	1	1	0	1	0	0	0	1	1

101 rows × 18 columns

In [98]:
```
zoo_feat = df.drop(['animal name',"type"],axis=1)
zoo_label = df.type
```

In [103]:
```
zoo_xtrain,zoo_xtest,zoo_ytrain,zoo_ytest = train_test_split(zoo_feat,zoo_lab
```

ID 3

In [111]:
```
from sklearn.metrics import accuracy_score
```

In [118]:
```
def dtree(criterion):
    tree = DecisionTreeClassifier(criterion = criterion)
    tree.fit(zoo_xtrain,zoo_ytrain)
    train_acc = tree.predict(zoo_xtrain)
    test_acc = tree.predict(zoo_xtest)
    print(f"Train Accuracy for {criterion} is ",accuracy_score(zoo_ytrain,tra
    print(f"Test Accuracy for {criterion} is ",accuracy_score(zoo_ytest,test_

```

In [119]:
```
1 dtree(criterion='entropy')
```

```
Train Accuracy for entropy is  1.0
Test Accuracy for entropy is  0.9523809523809523
```

In [120]:
```
1 dtree(criterion='gini')
```

```
Train Accuracy for gini is  1.0
Test Accuracy for gini is  0.9523809523809523
```

In [ ]:
```
1
```

REFRENCES

ID3 algorithm, stands for Iterative Dichotomiser 3, is a classification algorithm that follows a greedy approach of building a decision tree by selecting a best attribute that yields maximum Information Gain (IG) or minimum Entropy (H).

ID3 stands for Iterative Dichotomiser 3 and is named such because the algorithm iteratively (repeatedly) dichotomizes(divides) features into two or more groups at each step.

change criterion from "gini" to "entropy" for ID3

https://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart (https://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart)
https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTre (https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTre