NAME : RETHINAGIRI G

ROLL NO : 225229130

COURSE TITLE : PRACTICAL MACHINE LEARNING LAB

LAB.08 Animal Classification using Decision Trees

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

In [2]:
```
ani=pan.read_csv("C:\\Users\\1mscdsa30\\Downloads\\dataset_pml\\animals.csv")
```

In [3]:
```
ani
```

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]:
```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.metrics import*
```

In [5]:
```
X=ani.drop(['Species'],axis=1)
```

In [6]:
```
X
```

Out[6]:

	toothed	hair	breathes	legs
0	True	True	True	True
1	True	True	True	True
2	True	False	True	False
3	False	True	True	True
4	True	True	True	True
5	True	True	True	True
6	True	False	False	False
7	True	False	True	False
8	True	True	True	True
9	False	False	True	True

In [7]:
```
y=ani["Species"]
y
```

Out[7]:
```
0     Mammal
1     Mammal
2    Reptile
3     Mammal
4     Mammal
5     Mammal
6    Reptile
7    Reptile
8     Mammal
9    Reptile
Name: Species, dtype: object
```

In [8]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

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

In [10]:
```
dt.fit(X_train, y_train)


y_pred = dt.predict(X_test)
```

In [11]:
```
print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
Accuracy: 1.0
```

In [12]:
```
print("Classification Report : ",classification_report(y_test,y_pred))
```

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

     Mammal       1.00      1.00      1.00         1
    Reptile       1.00      1.00      1.00         1

avg / total       1.00      1.00      1.00         2
```

In [13]:
```
from sklearn import tree
```

In [14]:
```
with open("tree1.dot",'w') as f:
    f=tree.export_graphviz(dt,out_file=f,max_depth=4,
                           impurity=False,feature_names=X.columns.values,
                           class_names=['Reptile','Mammal'],
                           filled=True)
```

In [15]:
```
!type tree1.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 8\nvalue = [5, 3]\nclass = Reptile", fillcolor="#e5813966"] ;
1 [label="samples = 3\nvalue = [0, 3]\nclass = Mammal", fillcolor="#399de5ff"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 5\nvalue = [5, 0]\nclass = Reptile", fillcolor="#e58139ff"] ;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

Step-3

In [16]:
```
test=pan.read_csv("animals_test.csv")
```

In [17]:
```
test
```

Out[17]:

	toothed	hair	breathes	legs
0	False	False	True	False
1	False	True	True	True
2	True	False	True	True

Step-4

In [18]:
```
y_pred = dt.predict(test)
```

In [19]:
```
y_pred
```

Out[19]:
```
array(['Reptile', 'Mammal', 'Reptile'], dtype=object)
```

Step-5

In [20]:
```
dt1 = DecisionTreeClassifier(criterion='gini')
```

In [21]:
```
dt1.fit(X_train, y_train)


y_pred = dt1.predict(X_test)
```

In [22]:
```
print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
Accuracy: 1.0
```

In [23]:
```
print("Classification Report : ",classification_report(y_test,y_pred))
```

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

     Mammal       1.00      1.00      1.00         1
    Reptile       1.00      1.00      1.00         1

avg / total       1.00      1.00      1.00         2
```

In [24]:
```
with open("tree2.dot",'w') as f:
    f=tree.export_graphviz(dt1,out_file=f,max_depth=4,
                           impurity=False,feature_names=X.columns.values,
                           class_names=['Reptile','Mammal'],
                           filled=True)
```

In [25]:
```
!type tree2.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 8\nvalue = [5, 3]\nclass = Reptile", fillcolor="#e5813966"] ;
1 [label="samples = 3\nvalue = [0, 3]\nclass = Mammal", fillcolor="#399de5ff"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 5\nvalue = [5, 0]\nclass = Reptile", fillcolor="#e58139ff"] ;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

Step-6

In [26]:
```
zoo=pan.read_csv("Zoo.data")
```

In [27]:
```
zoo.head()
```

Out[27]:

	aardvark	1	0	0.1	1.1	0.2	0.3	1.2	1.3	1.4	1.5	0.4	0.5	4	0.6	0.7	1.6	1.7
0	antelope	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	1
1	bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	4
2	bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	1
3	boar	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	1	1
4	buffalo	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	1

In [28]:
```
zoo.size
```

Out[28]: 1800

In [29]:
```
X1=zoo.drop(["aardvark"],axis=1)
```

In [30]:
```
y1=zoo["aardvark"]
```

In [31]:
```
X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, test_size=0.2)
```

In [32]:
```
zoo1 = DecisionTreeClassifier(criterion='entropy')
```

In [33]:
```
zoo1.fit(X1_train, y1_train)


y1_pred = zoo1.predict(X1_test)
```

In [34]:
```
print("Accuracy:", accuracy_score(y1_test, y1_pred))
```

```
Accuracy: 0.0
```

In [35]:
```
print("Classification Report : ",classification_report(y1_test,y1_pred))
```

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

   antelope       0.00      0.00      0.00         0
       boar       0.00      0.00      0.00         0
    buffalo       0.00      0.00      0.00         1
       cavy       0.00      0.00      0.00         1
    chicken       0.00      0.00      0.00         1
       crab       0.00      0.00      0.00         1
   crayfish       0.00      0.00      0.00         1
       crow       0.00      0.00      0.00         0
       dove       0.00      0.00      0.00         1
       frog       0.00      0.00      0.00         0
       gull       0.00      0.00      0.00         0
    haddock       0.00      0.00      0.00         1
    hamster       0.00      0.00      0.00         1
       hare       0.00      0.00      0.00         0
       hawk       0.00      0.00      0.00         1
   housefly       0.00      0.00      0.00         0
       kiwi       0.00      0.00      0.00         0
   ladybird       0.00      0.00      0.00         1
    lobster       0.00      0.00      0.00         0
       moth       0.00      0.00      0.00         1
       newt       0.00      0.00      0.00         1
       oryx       0.00      0.00      0.00         1
   parakeet       0.00      0.00      0.00         0
    penguin       0.00      0.00      0.00         0
       puma       0.00      0.00      0.00         1
       rhea       0.00      0.00      0.00         1
   seahorse       0.00      0.00      0.00         0
    sealion       0.00      0.00      0.00         1
    skimmer       0.00      0.00      0.00         1
   slowworm       0.00      0.00      0.00         1
       slug       0.00      0.00      0.00         1
    tuatara       0.00      0.00      0.00         0
    vulture       0.00      0.00      0.00         1
       worm       0.00      0.00      0.00         0

avg / total       0.00      0.00      0.00        20
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-packages\sklearn\metrics\classif
ication.py:1135: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in label
s with no predicted samples.
  'precision', 'predicted', average, warn_for)
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-packages\sklearn\metrics\classif
ication.py:1137: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels w
ith no true samples.
  'recall', 'true', average, warn_for)
```

In [36]:
```
zoo["aardvark"].values
```

Out[36]:
```
array(['antelope', 'bass', 'bear', 'boar', 'buffalo', 'calf', 'carp',
       'catfish', 'cavy', 'cheetah', 'chicken', 'chub', 'clam', 'crab',
       'crayfish', 'crow', 'deer', 'dogfish', 'dolphin', 'dove', 'duck',
       'elephant', 'flamingo', 'flea', 'frog', 'frog', 'fruitbat',
       'giraffe', 'girl', 'gnat', 'goat', 'gorilla', 'gull', 'haddock',
       'hamster', 'hare', 'hawk', 'herring', 'honeybee', 'housefly',
       'kiwi', 'ladybird', 'lark', 'leopard', 'lion', 'lobster', 'lynx',
       'mink', 'mole', 'mongoose', 'moth', 'newt', 'octopus', 'opossum',
       'oryx', 'ostrich', 'parakeet', 'penguin', 'pheasant', 'pike',
       'piranha', 'pitviper', 'platypus', 'polecat', 'pony', 'porpoise',
       'puma', 'pussycat', 'raccoon', 'reindeer', 'rhea', 'scorpion',
       'seahorse', 'seal', 'sealion', 'seasnake', 'seawasp', 'skimmer',
       'skua', 'slowworm', 'slug', 'sole', 'sparrow', 'squirrel',
       'starfish', 'stingray', 'swan', 'termite', 'toad', 'tortoise',
       'tuatara', 'tuna', 'vampire', 'vole', 'vulture', 'wallaby', 'wasp',
       'wolf', 'worm', 'wren'], dtype=object)
```

In [38]:
```
with open("Zoo_Tree.dot",'w') as f:
    f=tree.export_graphviz(zoo1,out_file=f,max_depth=4,
                           impurity=False,feature_names=X1.columns.values,
                           class_names=zoo["aardvark"].values,
                           filled=True)
```

In [39]:
```
!type Zoo_Tree.dot
```

```
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 1, 0, 0, 0]\nclass = flamingo", fillcolor="#39e54700"]
;
56 -> 62 ;
63 [label="(...)", fillcolor="#C0C0C0"] ;
62 -> 63 ;
66 [label="(...)", fillcolor="#C0C0C0"] ;
62 -> 66 ;
69 [label="0.7 <= 0.5\nsamples = 9\nvalue = [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0\n0, 1, 0, 0, 0, 0, 1,
0, 0, 1, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0\n0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1\n0, 0, 0, 0, 0, 0, 0, 0, 0]\nclass = antelope", fillcolor="#e5813900"]
;
55 -> 69 ;
70 [label="0.1 <= 0.5\nsamples = 5\nvalue = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0\n0, 1, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1\n0, 0, 0, 0, 0, 0, 0, 0, 0]\nclass = antelope", fillcolor="#e5813900"]
;
69 -> 70 ;
71 [label="(...)", fillcolor="#C0C0C0"] ;
70 -> 71 ;
72 [label="(...)", fillcolor="#C0C0C0"] ;
```

In [40]:
```
zoo2 = DecisionTreeClassifier(criterion='gini')


zoo2.fit(X1_train, y1_train)

y2_pred = zoo2.predict(X1_test)
```

In [41]:
```
y2_pred
```

Out[41]:
```
array(['girl', 'crow', 'antelope', 'worm', 'parakeet', 'seal', 'gull',
       'calf', 'crow', 'lobster', 'starfish', 'boar', 'gnat', 'pitviper',
       'frog', 'kiwi', 'antelope', 'housefly', 'seahorse', 'parakeet'],
      dtype=object)
```

In [42]:
```
accuracy_score(y1_test, y2_pred)
```

Out[42]:
```
0.0
```

In [43]:
```
zoo['1.7'].unique
```

Out[43]:
```
<bound method Series.unique of 0     1
1     4
2     1
3     1
4     1
5     1
6     4
7     4
8     1
9     1
10    2
11    4
12    7
13    7
14    7
15    2
16    1
17    4
18    1
19    2
20    2
21    1
22    2
23    6
24    5
25    5
26    1
27    1
28    1
29    6
     ..
70    2
71    7
72    4
73    1
74    1
75    3
76    7
77    2
78    2
79    3
```

```
80    7
81    4
82    2
83    1
84    7
85    4
86    2
87    6
88    5
89    3
90    3
91    4
92    1
93    1
94    2
95    1
96    6
97    1
98    7
99    2
Name: 1.7, Length: 100, dtype: int64>
```

In [44]:

```
with open("Zoo_Tree1.dot",'w') as f:
    f=tree.export_graphviz(zoo2,out_file=f,max_depth=4,
                           impurity=False,feature_names=X1.columns.values,
                           class_names=zoo["aardvark"].values,
                           filled=True)
```

In [45]:
```
!type Zoo_Tree1.dot
```

```
4 [label="0.3 <= 0.5\nsamples = 4\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0\n0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1\n1, 0, 0, 0, 0, 0, 0, 0]\nclass = mink", fillcolor="#393ce500"] ;
3 -> 4 ;
5 [label="(...)", fillcolor="#C0C0C0"] ;
4 -> 5 ;
10 [label="(...)", fillcolor="#C0C0C0"] ;
4 -> 10 ;
11 [label="0.3 <= 0.5\nsamples = 14\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0\n1, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0\n0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1\n1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0,
0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 1]\nclass = chicken", fillcolor="#c8e53900"]
;
3 -> 11 ;
12 [label="(...)", fillcolor="#C0C0C0"] ;
11 -> 12 ;
23 [label="(...)", fillcolor="#C0C0C0"] ;
11 -> 23 ;
30 [label="1.3 <= 0.5\nsamples = 31\nvalue = [1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0\n0, 1, 0, 0, 0, 1, 1,
1, 0, 1, 1, 0, 1, 0\n0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0\n0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0\n0,
```

In [ ]: