```
225229124
```

Step1

In [1]:

```
import pandas as pd
```

In [2]:

```
df = pd.read_csv('species.csv')
df
```

Out[2]:

	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

Step2

In [3]:

```
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier as dtc
clf_entropy = dtc(criterion = "entropy")
```

In [4]:

```
X = df.iloc[:,:-1]
y = df.iloc[:,-1]
```

In [5]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size = 0.3, random_state
```

In [6]:

```
clf_entropy.fit(X_train,y_train)
y_pred = clf_entropy.predict(X_test)
y_pred
```

Out[6]:

```
array(['Mammal', 'Mammal', 'Mammal'], dtype=object)
```

In [7]:

```
from sklearn.metrics import accuracy_score,classification_report
accuracy_score(y_test,y_pred)
```

Out[7]:

```
1.0
```

In [8]:

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

```
             precision    recall  f1-score   support

     Mammal       1.00      1.00      1.00         3

avg / total       1.00      1.00      1.00         3
```

In [9]:

```
with open('tree1.dot','w') as f:
    f = tree.export_graphviz(clf_entropy,out_file=f,max_depth=4,impurity=False,feature_n
```

In [10]:

```
!type tree1.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 7\nvalue = [3, 4]\nclass = Mammal", fillc
olor="#399de540"] ;
1 [label="samples = 4\nvalue = [0, 4]\nclass = Mammal", fillcolor="#399de5
ff"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 3\nvalue = [3, 0]\nclass = Reptile", fillcolor="#e5813
9ff"] ;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

Step3

In [11]:

```
df1=pd.read_csv('animals_test.csv')
df1
```

Out[11]:

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

Step4

In [12]:

```
y_pred1 = clf_entropy.predict(df1)
y_pred1
```

Out[12]:

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

In [13]:

```
accuracy_score(y_test,y_pred1)
```

Out[13]:

```
0.3333333333333333
```

In [14]:

```
print(classification_report(y_test,y_pred1))
```

```
             precision    recall  f1-score   support

     Mammal       1.00      0.33      0.50         3
    Reptile       0.00      0.00      0.00         0

avg / total       1.00      0.33      0.50         3


C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\sit
e-packages\sklearn\metrics\classification.py:1137: UndefinedMetricWarning:
Recall and F-score are ill-defined and being set to 0.0 in labels with no
true samples.
  'recall', 'true', average, warn_for)
```

Step5

In [15]:

```
clf_gini = tree.DecisionTreeClassifier(criterion = "gini")
clf_gini.fit(X,y)
```

Out[15]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=Non
e,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=Non
e,
            splitter='best')
```

In [16]:

```
clf_gini.predict(df1)
```

Out[16]:

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

In [17]:

```
with open('tree2.dot','w') as f:
    f = tree.export_graphviz(clf_gini,out_file=f,max_depth=4,impurity=False,feature_name
```

In [18]:

```
!type tree2.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="hair <= 0.5\nsamples = 10\nvalue = [6, 4]\nclass = Reptile", fil
lcolor="#e5813955"] ;
1 [label="samples = 4\nvalue = [0, 4]\nclass = Mammal", fillcolor="#399de5
ff"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="samples = 6\nvalue = [6, 0]\nclass = Reptile", fillcolor="#e5813
9ff"] ;
0 -> 2 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

Step6

In [19]:

```
zoo = pd.read_csv('zoo.data')
zoo.head()
```

Out[19]:

	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 [20]:

```
X = zoo.iloc[:,1:-1]
y = zoo['1.7']
```

In [21]:

```
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size = 0.3, random_state
```

In [22]:

```
clf_entropy.fit(X_train,y_train)
y_pred = clf_entropy.predict(X_test)
y_pred
```

Out[22]:

```
array([1, 1, 2, 7, 1, 6, 2, 7, 2, 1, 1, 1, 1, 4, 5, 1, 7, 2, 7, 1, 2, 5,
       1, 2, 1, 2, 2, 6, 1, 4], dtype=int64)
```

In [23]:

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

Out[23]:

```
0.9666666666666667
```

In [24]:

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

```
             precision    recall  f1-score   support

          1       1.00      1.00      1.00        12
          2       1.00      1.00      1.00         8
          3       0.00      0.00      0.00         1
          4       1.00      1.00      1.00         2
          5       0.50      1.00      0.67         1
          6       1.00      1.00      1.00         2
          7       1.00      1.00      1.00         4

avg / total       0.95      0.97      0.96        30


C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\sit
e-packages\sklearn\metrics\classification.py:1135: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples.
  'precision', 'predicted', average, warn_for)
```

In [25]:

```
with open('tree3.dot','w') as f:
    f = tree.export_graphviz(clf_entropy,out_file=f,max_depth=4,impurity=False,feature_n
```

In [26]:

```
!type tree3.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="1.1 <= 0.5\nsamples = 70\nvalue = [28, 12, 4, 11, 3, 6, 6]\nclas
s = 1", fillcolor="#e5813946"] ;
1 [label="1.3 <= 0.5\nsamples = 42\nvalue = [0, 12, 4, 11, 3, 6, 6]\nclass
= 2", fillcolor="#b7e53908"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="1.4 <= 0.5\nsamples = 25\nvalue = [0, 12, 1, 0, 0, 6, 6]\nclass
= 2", fillcolor="#b7e53951"] ;
1 -> 2 ;
3 [label="0.2 <= 0.5\nsamples = 12\nvalue = [0, 0, 0, 0, 0, 6, 6]\nclass =
6", fillcolor="#b139e500"] ;
2 -> 3 ;
4 [label="1.2 <= 0.5\nsamples = 8\nvalue = [0, 0, 0, 0, 0, 2, 6]\nclass =
7", fillcolor="#e53986aa"] ;
3 -> 4 ;
5 [label="(...)", fillcolor="#C0C0C0"] ;
4 -> 5 ;
8 [label="(...)", fillcolor="#C0C0C0"] ;
4 -> 8 ;
9 [label="samples = 4\nvalue = [0, 0, 0, 0, 0, 4, 0]\nclass = 6", fillcolo
r="#b139e5ff"] ;
3 -> 9 ;
10 [label="0 <= 0.5\nsamples = 13\nvalue = [0, 12, 1, 0, 0, 0, 0]\nclass =
2", fillcolor="#b7e539ea"] ;
2 -> 10 ;
11 [label="samples = 1\nvalue = [0, 0, 1, 0, 0, 0, 0]\nclass = 3", fillcol
or="#39e54dff"] ;
10 -> 11 ;
12 [label="samples = 12\nvalue = [0, 12, 0, 0, 0, 0, 0]\nclass = 2", fillc
olor="#b7e539ff"] ;
10 -> 12 ;
13 [label="0.5 <= 0.5\nsamples = 17\nvalue = [0, 0, 3, 11, 3, 0, 0]\nclass
= 4", fillcolor="#39e5e292"] ;
1 -> 13 ;
14 [label="4 <= 2.0\nsamples = 6\nvalue = [0, 0, 3, 0, 3, 0, 0]\nclass =
3", fillcolor="#39e54d00"] ;
13 -> 14 ;
15 [label="samples = 3\nvalue = [0, 0, 3, 0, 0, 0, 0]\nclass = 3", fillcol
or="#39e54dff"] ;
14 -> 15 ;
16 [label="samples = 3\nvalue = [0, 0, 0, 0, 3, 0, 0]\nclass = 5", fillcol
or="#3956e5ff"] ;
14 -> 16 ;
17 [label="samples = 11\nvalue = [0, 0, 0, 11, 0, 0, 0]\nclass = 4", fillc
olor="#39e5e2ff"] ;
13 -> 17 ;
18 [label="samples = 28\nvalue = [28, 0, 0, 0, 0, 0, 0]\nclass = 1", fillc
olor="#e58139ff"] ;
0 -> 18 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```