Roll No: 205229133 ¶

Lab8. Animal Classification using Decision Trees

Objectives

In this lab, you will build ID3 and CART Decision Tree to classify whether an animal is a Mammal or Reptile.

Learning Outcomes

After completing this lab, you will be able to

- Create and import training dataset and test dataset
- Create ID3 Decision Tree using Entropy metric
- Create CART Decision Tree using Gini metric
- Visualize graph using graphviz

Import necessary Library

In [1]:

```
import csv
import pandas as pd
from sklearn import tree
from sklearn.tree import export_graphviz
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,classification_report
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report,auc,roc_c
import warnings
warnings.filterwarnings('ignore')
```

Step1. [Create Dataset]

Create the following dataset using Excel and save it as CSV file.

In [2]:

```
import pandas as pd
animal_data = pd.read_csv('animal.csv')
```

Step2. [Model building using ID3]

- Import your data set
- Create DT model using ‘entropy’ criterion
- Perform training and testing
- Print accuracy and classification report.
- Interpret your results

- Visualize your DT model using graphviz

In [3]:

```
animal_data
```

Out[3]:

	Toothed	Hair	Breathes	Legs	Species
0	True	True	True	True	Mammal
1	True	True	True	True	Mammal
2	True	False	True	False	Repite
3	False	True	True	True	Mammal
4	True	True	True	True	Mammal
5	True	True	True	True	Mammal
6	True	False	False	False	Repite
7	True	False	True	False	Repite
8	True	True	True	True	Mammal
9	False	False	True	True	Repite

In [4]:

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
animal_data["Label"] = label_encoder.fit_transform(animal_data["Species"])
animal_data
```

Out[4]:

	Toothed	Hair	Breathes	Legs	Species	Label
0	True	True	True	True	Mammal	0
1	True	True	True	True	Mammal	0
2	True	False	True	False	Repite	1
3	False	True	True	True	Mammal	0
4	True	True	True	True	Mammal	0
5	True	True	True	True	Mammal	0
6	True	False	False	False	Repite	1
7	True	False	True	False	Repite	1
8	True	True	True	True	Mammal	0
9	False	False	True	True	Repite	1

In [5]:

```
categories = list(label_encoder.inverse_transform([0,1]))
categories
```

Out[5]:

```
['Mammal', 'Repite']
```

In [6]:

```
X = animal_data.drop(['Label','Species'],axis=1)
```

In [7]:

```
y = animal_data.Label
```

In [8]:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,classification_report

X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.33,random_state=0)
clf = DecisionTreeClassifier(criterion='entropy',max_depth=4, random_state=42)
clf.fit(X_train,y_train)
```

Out[8]:

```
DecisionTreeClassifier(criterion='entropy', max_depth=4, random_state=42)
```

In [9]:

```
y_pred = clf.predict(X_test)
y_pred
```

Out[9]:

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

In [10]:

```
print("Accuracy of train :",clf.score(X_train,y_train))
print("Accuracy of test  :",clf.score(X_test,y_test))
print('\n')
print(classification_report(y_test,y_pred))
```

```
Accuracy of train : 1.0
Accuracy of test  : 0.75


              precision    recall  f1-score   support

           0       0.67      1.00      0.80         2
           1       1.00      0.50      0.67         2

    accuracy                           0.75         4
   macro avg       0.83      0.75      0.73         4
weighted avg       0.83      0.75      0.73         4
```

In [11]:

```
from sklearn import tree
from sklearn.tree import export_graphviz

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

Now open tree1.txt file which will be created in your working directory then Copy and paste the code to http://webgraphviz.com/ (http://webgraphviz.com/)

Another Way to visualize

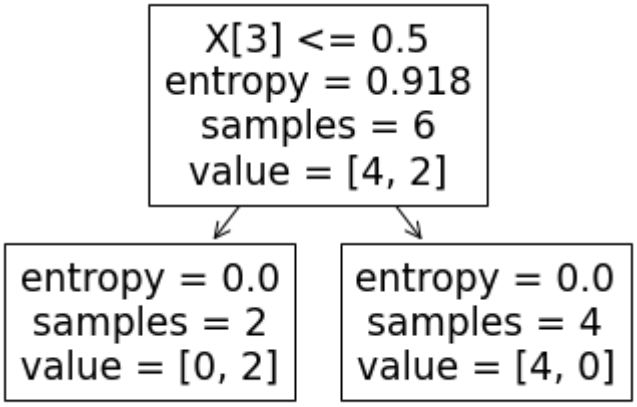
In [12]:

```
from sklearn import tree
%matplotlib inline

tree.plot_tree(clf)
```

Out[12]:

```
[Text(167.4, 163.07999999999998, 'X[3] <= 0.5\nentropy = 0.918\nsamples = 6
\nvalue = [4, 2]'),
 Text(83.7, 54.360000000000014, 'entropy = 0.0\nsamples = 2\nvalue = [0,
2]'),
 Text(251.10000000000002, 54.360000000000014, 'entropy = 0.0\nsamples = 4\nv
alue = [4, 0]')]
```



Step3. [Create a Test Set]

In [13]:

```
import csv
fields = ['Name','Toothed', 'Hair', 'Breathes', 'Legs', 'Species']

# data rows of csv file
rows = [ ['Turtile','FALSE','FALSE','TRUE','FALSE','Reptile'],
         ['Blue Whales','FALSE','TRUE','TRUE','TRUE','Mammal'],
         ['Crocodile','TRUE','FALSE','TRUE','TRUE','Reptile'] ]

# name of csv file
filename = "testing.csv"


with open(filename, 'w') as file:
    # creating a csv dict writer object
    writer = csv.writer(file)

    # writing headers (field names)
    writer.writerow(fields)

    # writing data rows
    writer.writerows(rows)
```

In [14]:

```
test_set = pd.read_csv('testing.csv')
test_set
```

Out[14]:

	Name	Toothed	Hair	Breathes	Legs	Species
0	Turtile	False	False	True	False	Reptile
1	Blue Whales	False	True	True	True	Mammal
2	Crocodile	True	False	True	True	Reptile

In [15]:

```
test_set["Label"] = label_encoder.fit_transform(test_set["Species"])
test_set
```

Out[15]:

	Name	Toothed	Hair	Breathes	Legs	Species	Label
0	Turtile	False	False	True	False	Reptile	1
1	Blue Whales	False	True	True	True	Mammal	0
2	Crocodile	True	False	True	True	Reptile	1

Step4. [Perform prediction]

- Use your ID3 DT model that you created before and predict labels for this test set. Check your predictions. Correct?

In [16]:

```
stp4 = test_set.drop(['Name','Species','Label'],axis=1)
y_prd = clf.predict(stp4)
y_prd
```

Out[16]:

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

In [17]:

```
accuracy_score(test_set.Label,y_prd)
```

Out[17]:

```
0.6666666666666666
```

Step5. [Build CART Decision Tree Model]

- Now, you are going to build a new CART decision tree using criterion='gini'.
- Train you model with full training data (No, train test split, this time)
- Predict samples for the test file
- Visualize your CART DT using graphviz

In [18]:

```
clf_1 = DecisionTreeClassifier(criterion='gini',max_depth=4, random_state=42)
clf_1.fit(X,y)
```

Out[18]:

```
DecisionTreeClassifier(max_depth=4, random_state=42)
```

In [19]:

```
clf_1.predict(stp4)
```

Out[19]:

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

In [20]:

```
with open("tree2.txt", 'w') as f:
    f = tree.export_graphviz(clf_1,out_file=f,max_depth = 4,impurity = False,feature_names
```

Now open tree2.txt file which will be created in your working directory then Copy and paste the code to [http://webgraphviz.com/ (http://webgraphviz.com/)](http://webgraphviz.com/)

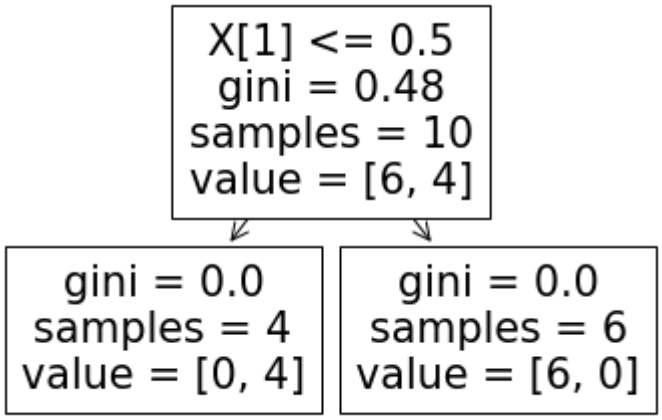
Another Way to visualize

In [21]:

```
from sklearn import tree
tree.plot_tree(clf_1)
```

Out[21]:

```
[Text(167.4, 163.07999999999998, 'X[1] <= 0.5\ngini = 0.48\nsamples = 10\nva
lue = [6, 4]'),
 Text(83.7, 54.360000000000014, 'gini = 0.0\nsamples = 4\nvalue = [0, 4]'),
 Text(251.10000000000002, 54.360000000000014, 'gini = 0.0\nsamples = 6\nvalu
e = [6, 0]')]
```



Step6. [Buid DT with Zoo dataset]

- Download full animal dataset at https://archive.ics.uci.edu/ml/datasets/Zoo (https://archive.ics.uci.edu/ml/datasets/Zoo)
- Import, build model using ID3 and CART, train and test accuracy. Print classification report. Visualize your trees.

In [22]:

```
animal_2 = pd.read_csv("zoo.csv")
```

In [23]:

```
animal_2.head()
```

Out[23]:

	name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breath
0	aardvark	1	0	0	1	0	0	1	1	1	
1	antelope	1	0	0	1	0	0	0	1	1	
2	bass	0	0	1	0	0	1	1	1	1	
3	bear	1	0	0	1	0	0	1	1	1	
4	boar	1	0	0	1	0	0	1	1	1	

In [24]:

```
animal_2.shape
```

Out[24]:

```
(101, 18)
```

In [25]:

```
animal_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101 entries, 0 to 100
Data columns (total 18 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   name      101 non-null    object
 1   hair      101 non-null    int64 
 2   feathers  101 non-null    int64 
 3   eggs      101 non-null    int64 
 4   milk      101 non-null    int64 
 5   airborne  101 non-null    int64 
 6   aquatic   101 non-null    int64 
 7   predator  101 non-null    int64 
 8   toothed   101 non-null    int64 
 9   backbone  101 non-null    int64 
 10  breathes  101 non-null    int64 
 11  venomous  101 non-null    int64 
 12  fins      101 non-null    int64 
 13  legs      101 non-null    int64 
 14  tail      101 non-null    int64 
 15  domestic  101 non-null    int64 
 16  catsize   101 non-null    int64 
 17  type      101 non-null    int64 
dtypes: int64(17), object(1)
memory usage: 14.3+ KB
```

In [26]:

```
X1 = animal_2.drop(['name','type'],axis=1)
y1 = animal_2.type
```

In [27]:

```
X__train,X__test,y__train,y__test = train_test_split(X1,y1,test_size=0.33,random_state=0)
```

In [28]:

```
clf_2 = DecisionTreeClassifier(criterion='entropy',max_depth=3, random_state=52)
clf_2.fit(X__train,y__train)
```

Out[28]:

```
DecisionTreeClassifier(criterion='entropy', max_depth=3, random_state=52)
```

In [29]:

```
clf_2.predict(X__test)
```

Out[29]:

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

In [30]:

```
clf_3 = DecisionTreeClassifier(criterion='gini',max_depth=4, random_state=42)
clf_3.fit(X__train,y__train)
```

Out[30]:

```
DecisionTreeClassifier(max_depth=4, random_state=42)
```

In [31]:

```
y_pred=clf_3.predict(X__test)
y_pred
```

Out[31]:

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

In [32]:

```
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
ss = scale.fit_transform(X__train)
ss1 = scale.transform(X__test)

print("model accuracy :",accuracy_score(y__test,y_pred))
print("Train accuracy :",clf_3.score(ss,y__train))
print("Test accuracy  :",clf_3.score(ss1,y__test))
```

```
model accuracy : 0.8235294117647058
Train accuracy : 0.9253731343283582
Test accuracy  : 0.8235294117647058
```

In [33]:

```
animal_2.type.value_counts(dropna=False)
```

Out[33]:

```
1    41
2    20
4    13
7    10
6     8
3     5
5     4
Name: type, dtype: int64
```

In [34]:

```
with open("tree3.txt", 'w') as f:
    f = tree.export_graphviz(clf_3,out_file=f,max_depth = 16,impurity = False,feature_names
```

In [35]:

```
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         6
           3       0.00      0.00      0.00         4
           4       1.00      1.00      1.00         7
           5       0.00      0.00      0.00         1
           6       1.00      0.50      0.67         2
           7       0.25      1.00      0.40         2

    accuracy                           0.82        34
   macro avg       0.61      0.64      0.58        34
weighted avg       0.81      0.82      0.80        34
```

Now open tree3.txt file which will be created in your working directory then Copy and paste the code to [http://webgraphviz.com/ (http://webgraphviz.com/)](http://webgraphviz.com/)

In [36]:

```
from sklearn import tree
tree.plot_tree(clf_3)
```

Out[36]:

```
[Text(239.14285714285714, 195.696, 'X[3] <= 0.5\ngini = 0.736\nsamples = 67
\nvalue = [29, 14, 1, 6, 3, 6, 8]'),
 Text(191.31428571428572, 152.208, 'X[1] <= 0.5\ngini = 0.763\nsamples = 38
\nvalue = [0, 14, 1, 6, 3, 6, 8]'),
 Text(143.4857142857143, 108.72, 'X[11] <= 0.5\ngini = 0.747\nsamples = 24\n
value = [0, 0, 1, 6, 3, 6, 8]'),
 Text(95.65714285714286, 65.232, 'X[4] <= 0.5\ngini = 0.66\nsamples = 18\nva
lue = [0, 0, 1, 0, 3, 6, 8]'),
 Text(47.82857142857143, 21.744, 'gini = 0.556\nsamples = 13\nvalue = [0, 0,
1, 0, 3, 1, 8]'),
 Text(143.4857142857143, 21.744, 'gini = 0.0\nsamples = 5\nvalue = [0, 0, 0,
0, 0, 5, 0]'),
 Text(191.31428571428572, 65.232, 'gini = 0.0\nsamples = 6\nvalue = [0, 0,
0, 6, 0, 0, 0]'),
 Text(239.14285714285714, 108.72, 'gini = 0.0\nsamples = 14\nvalue = [0, 14,
0, 0, 0, 0, 0]'),
 Text(286.9714285714286, 152.208, 'gini = 0.0\nsamples = 29\nvalue = [29, 0,
0, 0, 0, 0, 0]')]
```