In [1]:

```
##### Author : Amir Shokri
##### github link : https://github.com/amirshnll/Abalone
##### dataset link : http://archive.ics.uci.edu/ml/datasets/Abalone
##### email : amirsh.nll@gmail.com
```

In [35]:

```
import pandas as pd
from sklearn import tree
import matplotlib.pyplot as plt
import matplotlib.image as pltimg
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
```

In [52]:

```
#read file
df = pd.read_csv("D:\\abalone.txt", header=None)
for char in df:
    df = df.replace('M','1')
    df = df.replace('F','-1')
    df = df.replace('I','0')
df

#separate the feature columns from the target column.
features = [0,1,2,3,4,5,6,7]
X = df[features]
y = df[8]
print(X)
print(y)
```

```
       0      1      2      3       4       5       6       7
0      1  0.455  0.365  0.095  0.5140  0.2245  0.1010  0.1500
1      1  0.350  0.265  0.090  0.2255  0.0995  0.0485  0.0700
2     -1  0.530  0.420  0.135  0.6770  0.2565  0.1415  0.2100
3      1  0.440  0.365  0.125  0.5160  0.2155  0.1140  0.1550
4      0  0.330  0.255  0.080  0.2050  0.0895  0.0395  0.0550
...   ..    ...    ...    ...     ...     ...     ...     ...
4172  -1  0.565  0.450  0.165  0.8870  0.3700  0.2390  0.2490
4173   1  0.590  0.440  0.135  0.9660  0.4390  0.2145  0.2605
4174   1  0.600  0.475  0.205  1.1760  0.5255  0.2875  0.3080
4175  -1  0.625  0.485  0.150  1.0945  0.5310  0.2610  0.2960
4176   1  0.710  0.555  0.195  1.9485  0.9455  0.3765  0.4950

[4177 rows x 8 columns]
0       15
1        7
2        9
3       10
4        7
        ..
4172    11
4173    10
4174     9
4175    10
4176    12
Name: 8, Length: 4177, dtype: int64
```

In [53]:

```
#separate the Training data and Test data
X_train, X_test, y_train, y_test = train_test_split(X,y,random_state=1, test_s
ize=0.2)
# Feature scaling
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

In [54]:

```
#Finally for the Decision Tree
dtree =tree.DecisionTreeClassifier()
dtree.fit(X_train, y_train.values.ravel())
```

Out[54]:

```
DecisionTreeClassifier()
```

In [55]:

```
#In the prediction step, the model is used to predict the response for given d
ata.
predictions = dtree.predict(X_test)
print(predictions)
```

```
[12  9  8  8 11  5 10  9  9 12  9  6 11 12  5  9 11 10 10 11  9  8  6 16
 12  9 11  9 10 11 20 13 14  9  8  8 10 11 13 13 11 10 17 11  8  9 11 16
 13 10  9  8 10  6 13  9  9 17  8  6 13  9 11  8  8  7 11  8  8  6 10  8
  6  8  8  6 15 11  9 11  9  9 13 13 14  8  9 10 13  9 13  8  8 11  7 11
 12 14  6 10  7 12  6  9 13  9 16 12 10 10  7  9  9  7  6  9  8  6  9 14
 11 11  7 19 11 16 15 15  7 11  9  6 10  8 10  7  8 13 10 11  9  9  9 12
  8 14 12 10  9 10  9  7  9  9  8 14 10 10  8 11 11 11  9  7  9 10  9 12
  8 20  6 10 29  8  7 11 11 10  6  8 12  8  8  7  9 10 12  8 10 13  6 12
  6 12  6 11  8  5 10  8  4  8 15  9  8  7 11  7  9  6 15  7 13  7  8 12
 12 10 15  7  7  8 17 11 12 15  9  8 11  9 11  8 11 13  8  5 11 14 10 10
 16  9 10 10  7 10 10 11  9 11  9  9 11 11  6 15 10 13  7  9 10 15  7 10
 16 10  9 13  8 10  8 11 11  9 12 16  9 11  9 12 12  9 11  6  8  7  6  6
  6  7  6 12 10  8  9  9  8  8 11  9 10  6  8 11 12 10 15  7 10  9 13 13
  8  8  7 10 11  9  9  6 17  9 21 11  8 12  7 10 10 11  9  8 11  8  9  9
  9  4 10 16  6 10 17 16  8 11 11 13 13  9  9  5 10  9  4  9  8  9 14 10
  8  6  9 11  8 14  4  9 13  5  8 11 11 12  8 14 15  8  9  3 12  7  7 13
 23  7  8 10  6  9  9 17 13  7  8  9  8 21 10 15 13  7 11  9 14  5 10 13
 11 13 10 13  9  8 18  6 11  9  9  7 10  8 11  8 13  5  4 16  9  8 10  9
 10  5  9 10  8  8  6 13 10  8  8  9 11  8  9 15  7  8 12  8  4  8  9 11
  9  8 10 12 11  8  9 16  8 14 12 10 10  9 16  8  5 13  9 13 11  9 13  5
  6  7  9 10  7  8  7  9 13 11  7 11  8  6  5  6  8 11  8 12 17  5  8 11
 13 13 19  8 12 11 19  8 19 10  5  8  7 10 14  8  6 12  9 14  7  9 10  5
 17  9 11 11  9 11 10 11 13 10  6  5  8 11 10  9  5 11  9 10  8 10  8 10
  7 23 10  9  6 10  8  9  8  6  9  9  8 11 12 11 10  8 14  8 11 12  9 11
  9 13  6  8 11 10  8  5  9 12 10 10  3  9 11 15  9 10  8  7 11 14  4 12
 11  8  9  9  6 12 10 10  9 11  9 12 12  6 11  7  7  9  8 10 11 13  8 11
 10 10  7 10 15  9  9 12  9 11  9  6 13  5 15  9 10  6 11  7 11  9  8 11
  9  7 12  9 10 11  6 16  8  9  8  6 13  8  7  9  9 10  8  6  4  5  8 12
 10  6 14  9  8 10  7 14  5 10 10  6  9  9  8  9  9  9 14 13 11 14  8  8
  6 11  7  5 14  7  6 10 12  9 11 12 11  9 11 10 10 12 12 10 10 16  8 10
 12 19  9 11 12 11 13  9 10  7  9 14 10 10 10 13 13 10 11  8 14  8  8  9
  8 11  9 11  7  8  8 10 14  8 11 10  6  8 10  3 10  9  7 11 17  7 10 10
  7 10  9  6  9 10 19  7 11 10  7  9  8 10  9 10 10 10 11 16 14 13 10 18
 10 11  8 13 11  5  8 11 11 10  8 15  8  6  9 11 14 11 11  7  9  7 10 12
  8  9  9 10 15 13  6 11 10  7 12 10  8  8 13  8 10 13  9 10]
```

In [56]:

```
# Last thing: evaluation of algorithm performance in classifying
print(confusion_matrix(y_test,predictions))
print(classification_report(y_test,predictions))
```

```
[[ 0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  3  2  4  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  2  7  7  1  3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  1  9 16  8 10  2  3  3  0  0  0  0  0  1  0  0  0  0  0  0  0]
 [ 0  0  1  2 15 16 21 12  4  1  4  1  2  1  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  9 14 29 18 16  9  3  1  4  0  2  0  0  2  0  0  0  0  0]
 [ 0  0  0  0  3 12 20 36 20 19  5 11  1  4  0  1  0  0  0  0  0  0  0]
 [ 0  0  0  0  2  5 19 35 28 25 10  8  6  0  2  2  0  0  1  0  0  0  0]
 [ 0  0  0  0  0  2 10 16 18 22 10  6  3  1  0  2  0  0  0  1  0  0  0]
 [ 0  0  0  0  1  0  9 11  9 13  7  5  2  1  0  1  1  2  0  1  0  0  0]
 [ 0  0  0  0  0  0  2  5  9  4  5  5  3  2  2  1  0  0  0  0  0  1  0]
 [ 0  0  0  0  0  0  3  3  5  5  0  2  2  2  2  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  1  4  2  0  3  2  3  4  0  0  0  0  0  0  1  1]
 [ 0  0  0  0  0  0  0  0  0  4  1  2  0  1  1  0  0  0  1  0  0  0  0]
 [ 0  0  0  0  0  0  1  2  2  0  2  3  0  1  0  0  0  1  0  0  0  0  0]
 [ 0  0  0  0  0  1  1  1  2  2  0  0  1  0  0  1  0  1  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  1  2  0  0  1  0  1  1  0  1  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  1  0  0  2  0  1  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  1  0  0  0  1  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]]
              precision    recall  f1-score   support

           2       0.00      0.00      0.00         1
           3       0.00      0.00      0.00         2
           4       0.25      0.18      0.21        11
           5       0.30      0.35      0.33        20
           6       0.30      0.30      0.30        53
           7       0.27      0.20      0.23        80
           8       0.23      0.27      0.25       107
           9       0.25      0.27      0.26       132
          10       0.23      0.20      0.21       143
          11       0.20      0.24      0.22        91
          12       0.14      0.11      0.12        63
          13       0.10      0.13      0.11        39
          14       0.08      0.08      0.08        24
          15       0.17      0.14      0.15        21
          16       0.07      0.10      0.08        10
          17       0.00      0.00      0.00        12
          18       0.00      0.00      0.00        10
          19       0.00      0.00      0.00         7
          20       0.00      0.00      0.00         4
          21       0.00      0.00      0.00         3
          22       0.00      0.00      0.00         1
          23       0.00      0.00      0.00         2
          29       0.00      0.00      0.00         0

    accuracy                           0.21       836
   macro avg       0.11      0.11      0.11       836
weighted avg       0.21      0.21      0.21       836
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\_classification.p
y:1221: UndefinedMetricWarning: Precision and F-score are ill-defined and bei
ng set to 0.0 in labels with no predicted samples. Use `zero_division` parame
ter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\_classification.p
y:1221: UndefinedMetricWarning: Recall and F-score are ill-defined and being
set to 0.0 in labels with no true samples. Use `zero_division` parameter to c
ontrol this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [57]:

```
# mean accuracy on the given test data and labels.
dtree.score(X_test,y_test)
```

Out[57]:

```
0.20813397129186603
```

In [ ]: