how to build a Decision Tree, and compare DT1 and DT2

In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure

from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
import graphviz
from sklearn import preprocessing
```

In [2]:

```
df = pd.read_csv('wdbc.data',header = None)
```

In [3]:

```
df.head(5)
```

Out[3]:

	0	1	2	3	4	5	6	7	8	9	...	22
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	25.38
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	24.99
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	23.57
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	14.91
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	22.54

5 rows × 32 columns

In [4]:

```
df.dtypes
```

Out[4]:

```
0        int64
1       object
2      float64
3      float64
4      float64
5      float64
6      float64
7      float64
8      float64
9      float64
10     float64
11     float64
12     float64
13     float64
14     float64
15     float64
16     float64
17     float64
18     float64
19     float64
20     float64
21     float64
22     float64
23     float64
24     float64
25     float64
26     float64
27     float64
28     float64
29     float64
30     float64
31     float64
dtype: object
```

有关pandas中 DataFrame的处理

https://www.cnblogs.com/ffli/p/12202302.html (https://www.cnblogs.com/ffli/p/12202302.html)

This is the DT1: DT with Information Gain (IG)

In [5]:

```
X = df.iloc[:,2:32]
Y = df.iloc[:,1]
clf1 = DecisionTreeClassifier(criterion='entropy')
```

In [6]:

```
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score

acc_train_sum = 0.0
acc_test_sum = 0.0
pre_sum = 0.0
rec_sum = 0.0
for i in range(0,20):
    (X_train,X_test,Y_train,Y_test)=train_test_split(X, Y, test_size=0.3)
    clf1.fit(X_train, Y_train)
    y_pred = clf1.predict(X_test)

    acc_train_sum += clf1.score(X_train, Y_train)
    acc_test_sum += clf1.score(X_test, Y_test)
    pre_sum += precision_score(Y_test, y_pred, labels=['M','B'], pos_label='M', average='binary'
, sample_weight=None)
    rec_sum += recall_score(Y_test, y_pred, labels=['M','B'], pos_label='M', average='binary', s
ample_weight=None)
#    print(i, "time the score is", clf1.score(X_test, Y_test))
print("DT1:Average Accuray train is", acc_train_sum/20)
print("DT1:Average Accuray test is", acc_test_sum/20)
print("DT1:Average Precision is", pre_sum/20)
print("DT1:Average Recall is", rec_sum/20)
```

```
DT1:Average Accuray train is 1.0
DT1:Average Accuray test is 0.9254385964912281
DT1:Average Precision is 0.8980360547514327
DT1:Average Recall is 0.903790537215712
```

In [7]:

```
dot_data = tree.export_graphviz(clf1, out_file=None)
graph = graphviz.Source(dot_data)
graph.render("HW2-clf1")
!HW2-clf1.pdf
```

Next is the DT2: DT with IG & limited tree size, vary the number of levels and try to beat DT1

In [8]:

```
clf2 = DecisionTreeClassifier(criterion='entropy',min_samples_leaf=4, min_samples_split=10, max_
leaf_nodes=None)

acc_train_sum = 0.0
acc_test_sum = 0.0
pre_sum = 0.0
rec_sum = 0.0
for i in range(0,20):
    (X_train,X_test,Y_train,Y_test)=train_test_split(X, Y, test_size=0.3)
    clf2.fit(X_train, Y_train)
    y_pred = clf2.predict(X_test)

    acc_train_sum += clf2.score(X_train, Y_train)
    acc_test_sum += clf2.score(X_test, Y_test)
    pre_sum += precision_score(Y_test, y_pred, labels=['M','B'], pos_label='M', average='binary'
, sample_weight=None)
    rec_sum += recall_score(Y_test, y_pred, labels=['M','B'], pos_label='M', average='binary', s
ample_weight=None)
#    print(i, "time the score is", clf1.score(X_test, Y_test))
print("DT2:Average Accuray train is", acc_train_sum/20)
print("DT2:Average Accuray test is", acc_test_sum/20)
print("DT2:Average Precision is", pre_sum/20)
print("DT2:Average Recall is", rec_sum/20)
```

```
DT2:Average Accuray train is 0.983291457286432
DT2:Average Accuray test is 0.9368421052631577
DT2:Average Precision is 0.9257994815540134
DT2:Average Recall is 0.9020124833516256
```

In [9]:

```
dot_data = tree.export_graphviz(clf2, out_file=None)
graph = graphviz.Source(dot_data)
graph.render("HW2-clf2")
!HW2-clf2.pdf
```

In [10]:

```
#from sklearn.metrics import confusion_matrix
#print(confusion_matrix(Y_test, y_pred,labels = ['M','B']))
```