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

In [2]:
```
from sklearn.datasets import load_breast_cancer
iris = load_breast_cancer()
```

In [3]:
```
X = pd.DataFrame(iris.data, columns=['Xvar'+str(i+1)+"S" for i in range(iris.data.shape[1])])
X
```

Out[3]:

	Xvar1S	Xvar2S	Xvar3S	Xvar4S	Xvar5S	Xvar6S	Xvar7S	Xvar8S	Xvar9S	Xvar10S	...	Xvar21S	Xvar22S	Xvar23S	Xva
0	17.990	10.38	122.80	1001.0	0.11840	0.27760	0.300100	0.147100	0.2419	0.07871	...	25.380	17.33	184.60	201
1	20.570	17.77	132.90	1326.0	0.08474	0.07864	0.086900	0.070170	0.1812	0.05667	...	24.990	23.41	158.80	195
2	19.690	21.25	130.00	1203.0	0.10960	0.15990	0.197400	0.127900	0.2069	0.05999	...	23.570	25.53	152.50	170
3	11.420	20.38	77.58	386.1	0.14250	0.28390	0.241400	0.105200	0.2597	0.09744	...	14.910	26.50	98.87	567.
4	20.290	14.34	135.10	1297.0	0.10030	0.13280	0.198000	0.104300	0.1809	0.05883	...	22.540	16.67	152.20	157
5	12.450	15.70	82.57	477.1	0.12780	0.17000	0.157800	0.080890	0.2087	0.07613	...	15.470	23.75	103.40	741.
6	18.250	19.98	119.60	1040.0	0.09463	0.10900	0.112700	0.074000	0.1794	0.05742	...	22.880	27.66	153.20	160
7	13.710	20.83	90.20	577.9	0.11890	0.16450	0.093660	0.059850	0.2196	0.07451	...	17.060	28.14	110.60	897.
8	13.000	21.82	87.50	519.8	0.12730	0.19320	0.185900	0.093530	0.2350	0.07389	...	15.490	30.73	106.20	739.
9	12.460	24.04	83.97	475.9	0.11860	0.23960	0.227300	0.085430	0.2030	0.08243	...	15.090	40.68	97.65	711.
10	16.020	23.24	102.70	797.8	0.08206	0.06669	0.032990	0.033230	0.1528	0.05697	...	19.190	33.88	123.80	115
11	15.780	17.89	103.60	781.0	0.09710	0.12920	0.099540	0.066060	0.1842	0.06082	...	20.420	27.28	136.50	129
12	19.170	24.80	132.40	1123.0	0.09740	0.24580	0.206500	0.111800	0.2397	0.07800	...	20.960	29.94	151.70	133
13	15.850	23.95	103.70	782.7	0.08401	0.10020	0.099380	0.053640	0.1847	0.05338	...	16.840	27.66	112.00	876.
14	13.730	22.61	93.60	578.3	0.11310	0.22930	0.212800	0.080250	0.2069	0.07682	...	15.030	32.01	108.80	697.
15	14.540	27.54	96.73	658.8	0.11390	0.15950	0.163900	0.073640	0.2303	0.07077	...	17.460	37.13	124.10	943.
16	14.680	20.13	94.74	684.5	0.09867	0.07200	0.073950	0.052590	0.1586	0.05922	...	19.070	30.88	123.40	113
17	16.130	20.68	108.10	798.8	0.11700	0.20220	0.172200	0.102800	0.2164	0.07356	...	20.960	31.48	136.80	131
18	19.810	22.15	130.00	1260.0	0.09831	0.10270	0.147900	0.094980	0.1582	0.05395	...	27.320	30.88	186.80	239
19	13.540	14.36	87.46	566.3	0.09779	0.08129	0.066640	0.047810	0.1885	0.05766	...	15.110	19.26	99.70	711.
20	13.080	15.71	85.63	520.0	0.10750	0.12700	0.045680	0.031100	0.1967	0.06811	...	14.500	20.49	96.09	630.
21	9.504	12.44	60.34	273.9	0.10240	0.06492	0.029560	0.020760	0.1815	0.06905	...	10.230	15.66	65.13	314.
22	15.340	14.26	102.50	704.4	0.10730	0.21350	0.207700	0.097560	0.2521	0.07032	...	18.070	19.08	125.10	980.
23	21.160	23.04	137.20	1404.0	0.09428	0.10220	0.109700	0.086320	0.1769	0.05278	...	29.170	35.59	188.00	261
24	16.650	21.38	110.00	904.6	0.11210	0.14570	0.152500	0.091700	0.1995	0.06330	...	26.460	31.56	177.00	221
25	17.140	16.40	116.00	912.7	0.11860	0.22760	0.222900	0.140100	0.3040	0.07413	...	22.250	21.40	152.40	146
26	14.580	21.53	97.41	644.8	0.10540	0.18680	0.142500	0.087830	0.2252	0.06924	...	17.620	33.21	122.40	896.
27	18.610	20.25	122.10	1094.0	0.09440	0.10660	0.149000	0.077310	0.1697	0.05699	...	21.310	27.26	139.90	140
28	15.300	25.27	102.40	732.4	0.10820	0.16970	0.168300	0.087510	0.1926	0.06540	...	20.270	36.71	149.30	126
29	17.570	15.05	115.00	955.1	0.09847	0.11570	0.098750	0.079530	0.1739	0.06149	...	20.010	19.52	134.90	122
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
539	7.691	25.44	48.34	170.4	0.08668	0.11990	0.092520	0.013640	0.2037	0.07751	...	8.678	31.89	54.49	223.
540	11.540	14.44	74.65	402.9	0.09984	0.11200	0.067370	0.025940	0.1818	0.06782	...	12.260	19.68	78.78	457.
541	14.470	24.99	95.81	656.4	0.08837	0.12300	0.100900	0.038900	0.1872	0.06341	...	16.220	31.73	113.50	808.
542	14.740	25.42	94.70	668.6	0.08275	0.07214	0.041050	0.030270	0.1840	0.05680	...	16.510	32.29	107.40	826.
543	13.210	28.06	84.88	538.4	0.08671	0.06877	0.029870	0.032750	0.1628	0.05781	...	14.370	37.17	92.48	629.
544	13.870	20.70	89.77	584.8	0.09578	0.10180	0.036880	0.023690	0.1620	0.06688	...	15.050	24.75	99.17	688.
545	13.620	23.23	87.19	573.2	0.09246	0.06747	0.029740	0.024430	0.1664	0.05801	...	15.350	29.09	97.58	729.
546	10.320	16.35	65.31	324.9	0.09434	0.04994	0.010120	0.005495	0.1885	0.06201	...	11.250	21.77	71.12	384.
547	10.260	16.58	65.85	320.8	0.08877	0.08066	0.043580	0.024380	0.1669	0.06714	...	10.830	22.04	71.08	357.

548	9.683	19.34	61.05	285.7	0.08491	0.05030	0.023370	0.009615	0.1580	0.06235	...	10.930	25.59	69.10	364.
549	10.820	24.21	68.89	361.6	0.08192	0.06602	0.015480	0.008160	0.1976	0.06328	...	13.030	31.45	83.90	505.
550	10.860	21.48	68.51	360.5	0.07431	0.04227	0.000000	0.000000	0.1661	0.05948	...	11.660	24.77	74.08	412.
551	11.130	22.44	71.49	378.4	0.09566	0.08194	0.048240	0.022570	0.2030	0.06552	...	12.020	28.26	77.80	436.
552	12.770	29.43	81.35	507.9	0.08276	0.04234	0.019970	0.014990	0.1539	0.05637	...	13.870	36.00	88.10	594.
553	9.333	21.94	59.01	264.0	0.09240	0.05605	0.039960	0.012820	0.1692	0.06576	...	9.845	25.05	62.86	295.
554	12.880	28.92	82.50	514.3	0.08123	0.05824	0.061950	0.023430	0.1566	0.05708	...	13.890	35.74	88.84	595.
555	10.290	27.61	65.67	321.4	0.09030	0.07658	0.059990	0.027380	0.1593	0.06127	...	10.840	34.91	69.57	357.
556	10.160	19.59	64.73	311.7	0.10030	0.07504	0.005025	0.011160	0.1791	0.06331	...	10.650	22.88	67.88	347.
557	9.423	27.88	59.26	271.3	0.08123	0.04971	0.000000	0.000000	0.1742	0.06059	...	10.490	34.24	66.50	330.
558	14.590	22.68	96.39	657.1	0.08473	0.13300	0.102900	0.037360	0.1454	0.06147	...	15.480	27.27	105.90	733.
559	11.510	23.93	74.52	403.5	0.09261	0.10210	0.111200	0.041050	0.1388	0.06570	...	12.480	37.16	82.28	474.
560	14.050	27.15	91.38	600.4	0.09929	0.11260	0.044620	0.043040	0.1537	0.06171	...	15.300	33.17	100.20	706.
561	11.200	29.37	70.67	386.0	0.07449	0.03558	0.000000	0.000000	0.1060	0.05502	...	11.920	38.30	75.19	439.
562	15.220	30.62	103.40	716.9	0.10480	0.20870	0.255000	0.094290	0.2128	0.07152	...	17.520	42.79	128.70	915.
563	20.920	25.09	143.00	1347.0	0.10990	0.22360	0.317400	0.147400	0.2149	0.06879	...	24.290	29.41	179.10	181
564	21.560	22.39	142.00	1479.0	0.11100	0.11590	0.243900	0.138900	0.1726	0.05623	...	25.450	26.40	166.10	202
565	20.130	28.25	131.20	1261.0	0.09780	0.10340	0.144000	0.097910	0.1752	0.05533	...	23.690	38.25	155.00	173
566	16.600	28.08	108.30	858.1	0.08455	0.10230	0.092510	0.053020	0.1590	0.05648	...	18.980	34.12	126.70	112
567	20.600	29.33	140.10	1265.0	0.11780	0.27700	0.351400	0.152000	0.2397	0.07016	...	25.740	39.42	184.60	182
568	7.760	24.54	47.92	181.0	0.05263	0.04362	0.000000	0.000000	0.1587	0.05884	...	9.456	30.37	59.16	268.

569 rows × 30 columns

In [4]:

```
Y = pd.DataFrame(iris.target, columns=['Y'])
Y.T
```

Out[4]:

	0	1	2	3	4	5	6	7	8	9	...	559	560	561	562	563	564	565	566	567	568
Y	0	0	0	0	0	0	0	0	0	0	...	1	1	1	0	0	0	0	0	0	1

1 rows × 569 columns

In [5]:

```
from sklearn.cross_validation import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3)
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

```
/data/soft/py3/lib/python3.6/site-packages/sklearn/cross_validation.py:41: DeprecationWarning: This mod
ule was deprecated in version 0.18 in favor of the model_selection module into which all the refactored
classes and functions are moved. Also note that the interface of the new CV iterators are different fro
m that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
```

Out[5]: `((398, 30), (171, 30), (398, 1), (171, 1))`

In [6]:

```
from sklearn import tree
from sklearn import metrics
import numpy as np
```

In [7]:

```
model = tree.DecisionTreeClassifier(min_samples_leaf=50)
model.fit(X_train, Y_train)
with open("013.Test_DecisionTreeClassifier_var.result.dot", "w") as f:
    tree.export_graphviz(model, f)
```

In [8]:
```
! cat "013.Test_DecisionTreeClassifier_var.result.dot"
```

```
digraph Tree {
node [shape=box] ;
0 [label="X[27] <= 0.142\ngini = 0.472\nsamples = 398\nvalue = [152, 246]"] ;
1 [label="X[20] <= 15.68\ngini = 0.15\nsamples = 258\nvalue = [21, 237]"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="X[15] <= 0.012\ngini = 0.028\nsamples = 208\nvalue = [3, 205]"] ;
1 -> 2 ;
3 [label="gini = 0.105\nsamples = 54\nvalue = [3, 51]"] ;
2 -> 3 ;
4 [label="gini = 0.0\nsamples = 154\nvalue = [0, 154]"] ;
2 -> 4 ;
5 [label="gini = 0.461\nsamples = 50\nvalue = [18, 32]"] ;
1 -> 5 ;
6 [label="X[0] <= 16.1\ngini = 0.12\nsamples = 140\nvalue = [131, 9]"] ;
0 -> 6 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
7 [label="gini = 0.295\nsamples = 50\nvalue = [41, 9]"] ;
6 -> 7 ;
8 [label="gini = 0.0\nsamples = 90\nvalue = [90, 0]"] ;
6 -> 8 ;
}
```

In [9]:
```python
col = X_train.columns
for i in range(len(model.tree_.threshold)):

    if model.tree_.feature[i] == -2:
        print(model.tree_.feature[i])
    else:
        print(col[model.tree_.feature[i]],model.tree_.feature[i],model.tree_.threshold[i], model.tree_.i
mpurity[i])
```

```
Xvar28S 27 0.1423499882221222 0.47210929016944014
Xvar21S 20 15.680000305175781 0.1495402920497566
Xvar16S 15 0.012025000527501106 0.02843010355029585
-2
-2
-2
Xvar1S 0 16.099998474121094 0.12030612244897965
-2
-2
```

In [10]:
```python
rule = []
arule = {}
r = []
with open("013.Test_DecisionTreeClassifier_var.result.dot", "r") as f:
    for i in f:
        i = i.strip('\n')
        if "digraph Tree {" in i or "node [shape=box] ;" in i or "}" in i:
            pass
        elif "->" in i:
            if "[" in i:
                i = i[:i.find("[")]
            i = i.replace(" ", "").replace(";", "")
            [i0, i1] = i.split("->")
            i0, i1 = int(i0), int(i1)
            i2 = 1 if i1 == (i0 + 1) else 0
            rule.append([i0, i1, i2])
        else:
            ii = i[i.find('[label="')+len('[label="'):i.find("\\n")]
            if "gini" in ii:
                ii = ""
            arule[int(i.split(" ")[0])] = ii

        if 'label="gini =' in i:
            t0 = int(i.split(' ')[0])
            t = i[i.find("nvalue ="):].replace('"] ;','').replace('nvalue = [',"")
            [t1, t2] = t.replace('\n', "").split(",")
            t1, t2 = int(t1), int(t2)
            r.append([t0, t1, t2, t2/(t1+t2)*100])

print(rule, arule)

vrule = arule
for i, j, k in rule:
    t = arule[i]
    if k == 1:
        pass
    else:
        t = t.replace("<=", ">")

    if arule[j] != "":
        vrule[j] = t + " and " + arule[j]
    else:
        vrule[j] = t

for i, j in vrule.items():
    n = 0
    for k in col:
        j = j.replace("X["+str(n)+"]", k)
        n += 1
    print(i, j)
r = pd.DataFrame(r, columns=['No', 'G', 'B', 'B/(G+B)'])
r.sort_values('B/(G+B)', ascending=False)
```

```
[[0, 1, 1], [1, 2, 1], [2, 3, 1], [2, 4, 0], [1, 5, 0], [0, 6, 0], [6, 7, 1], [6, 8, 0]] {0: 'X[27] <=
0.142', 1: 'X[20] <= 15.68', 2: 'X[15] <= 0.012', 3: '', 4: '', 5: '', 6: 'X[0] <= 16.1', 7: '', 8: ''}
0 Xvar28S <= 0.142
1 Xvar28S <= 0.142 and Xvar21S <= 15.68
2 Xvar28S <= 0.142 and Xvar21S <= 15.68 and Xvar16S <= 0.012
3 Xvar28S <= 0.142 and Xvar21S <= 15.68 and Xvar16S <= 0.012
4 Xvar28S > 0.142 and Xvar21S > 15.68 and Xvar16S > 0.012
5 Xvar28S > 0.142 and Xvar21S > 15.68
6 Xvar28S > 0.142 and Xvar1S <= 16.1
7 Xvar28S > 0.142 and Xvar1S <= 16.1
8 Xvar28S > 0.142 and Xvar1S > 16.1
```

Out[10]:

	No	G	B	B/(G+B)
1	4	0	154	100.000000
0	3	3	51	94.444444
2	5	18	32	64.000000
3	7	41	9	18.000000
4	8	90	0	0.000000

In [11]:
```python
Q = "Xvar24S > 884.55 and Xvar28S > 0.111 and Xvar19S > 0.016"
```

In [12]:
```
d = pd.merge(X, Y, left_index=True, right_index=True)
d = d.query(Q)
pd.value_counts(d['Y'], normalize=True)
```

Out[12]:
```
0    0.99
1    0.01
Name: Y, dtype: float64
```

In [13]:
```
d = pd.merge(X_train, Y_train, left_index=True, right_index=True)
d = d.query(Q)
pd.value_counts(d['Y'], normalize=True)
```

Out[13]:
```
0    0.986486
1    0.013514
Name: Y, dtype: float64
```

In [14]:
```
d = pd.merge(X_test, Y_test, left_index=True, right_index=True)
d = d.query(Q)
pd.value_counts(d['Y'], normalize=True)
```

Out[14]:
```
0    1.0
Name: Y, dtype: float64
```