Yuan Wang
440 Data Mining
Prof. Jiebo Luo
Homework 4

7.5

Step 1:

For each frequent item, construct its conditional pattern-base and conditional FP-Tree
Recursively create conditional FP-Tree until the resulting FP-Tree is empty

Step 2:

For each frequent itemset X, scan infrequent item sets Z, which contains X.
Find Y contains Z which are not in X
If Y is a frequent item set, then calculate
(P(X | Y)+P(Y | X))/2 = (sup(Z)/sup(Y)+sup(Z)/sup(X))/2 to determine whether X and Y are negatively correlated

7.9

The distance measure Pat Dist is a valid distance metric.
When $P_1 = P_2$, $Pat\_Dist\ (P_1\ ,\ P_2) = 0$
When $P_1 \neq P_2$, $Pat\_Dist\ (P_1\ ,\ P_2) > 0$
It also has
$Pat\_Dist\ (P_1\ ,\ P_2) = Pat\_Dist\ (P_2\ ,\ P_1)$,
$Pat\_Dist\ (P_1\ ,\ P_2) + Pat\_Dist\ (P_2\ ,\ P_3) \geq Pat\_Dist\ (P_1\ ,\ P_3)$

7.10

We can use clustering method based on δ-clusters. First, partitioning the dataset to N subsets, find the local representatives which δ-cover the most other patterns locally. Then find the global representative which δ-cover based on local combined.

8.3

Method (a) prune the rule, it may remove any precondition of the rule.
Method (b) prune the subtree, it may remove the whole subtree.
Method (a) is less restrictive.

8.5

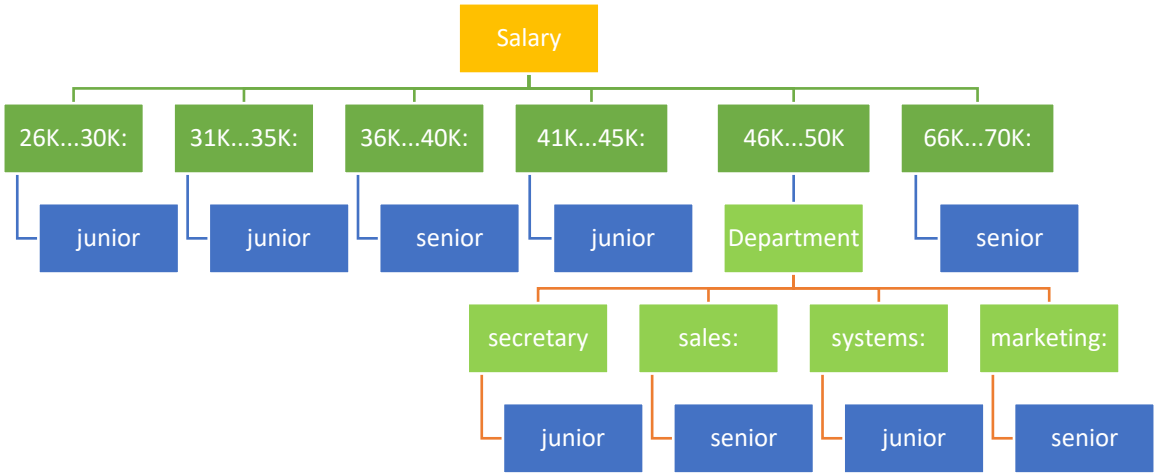
Use the Rainforest algorithm for this case. Assume there are C class labels. Scan the whole databased to build up AVC-list for each of the 50 attributes. The size of each AVC-list is 100*C. the total size of the AVC-set is 100*C*50, which fits 512MB memory. The other AVC-set takes less computation because there is less attributes available. We can compute the AVS-set for nodes at the same level of the tree in parallel to reduce the scan times.

8.7

(a)

The attribute selection measure should count each of tuples.
Determine the most common class among tuples

(b)



(c)

$P(X|senior) = 0;$

$P(X|junior) = \frac{31}{113} \times \frac{46}{113} \times \frac{20}{113} = 0.018$

Naïve Bayesian predicts "junior"

8.12

Tuple	Class	Prob	TP	FP	TN	FN	TPR	RPR
1	P	0.95	1	0	5	4	0.2	0
2	N	0.85	1	1	4	4	0.2	0.2

3	P	0.78	2	1	4	3	0.4	0.2
4	P	0.66	3	1	4	2	0.6	0.2
5	N	0.6	3	2	3	2	0.6	0.4
6	P	0.55	4	2	3	1	0.8	0.4
7	N	0.53	4	3	2	1	0.8	0.6
8	N	0.52	4	4	1	1	0.8	0.8
9	N	0.51	4	5	0	1	0.8	1
10	P	0.4	5	5	0	0	1	1

 # 8.12

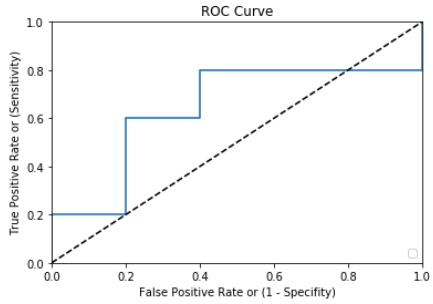
In [67]:
```
from sklearn.metrics import roc_curve

y = np.array([1, 0, 1, 1, 0, 1, 0, 0, 0, 1])
prob = np.array([0.95, 0.85, 0.78, 0.66, 0.6, 0.55, 0.53, 0.52, 0.51, 0.4])
FPR, TPR, thresholds = roc_curve(y, prob)


plt.plot(FPR, TPR)
plt.plot([0, 1], [0, 1], 'k--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.xlabel('False Positive Rate or (1 - Specifity)')
plt.ylabel('True Positive Rate or (Sensitivity)')
plt.title('ROC Curve')
plt.legend(loc="lower right")
```

```
No handles with labels found to put in legend.
```

Out[67]: <matplotlib.legend.Legend at 0x7faae2518c10>



8.14

Given that Alpha is 1%, we accept the null hypothesis because p_value = 2.3% > α =1%
I also use the scipy to test but it got slightly different value (though, it does not impact the conclusion), but I wonder why they are different.

In [19]:
```
import numpy as np
from scipy import stats
M1 = np.array([30.5, 32.2, 20.7, 20.6, 31.0, 41.0, 27.7, 26.0, 21.5, 26.0])
M2 = np.array([22.4, 14.5, 22.4, 19.6, 20.7, 20.4, 22.1, 19.4, 16.2, 35.0])
```

In [21]:
```
d = M1.mean() - M2.mean()
d
```

Out[21]:
```
6.449999999999999
```

In [25]:
```
M = M1 - M2
M
```

Out[25]:
```
array([ 8.1, 17.7, -1.7,  1. , 10.3, 20.6,  5.6,  6.6,  5.3, -9. ])
```

In [28]:
```
sd = np.sqrt(M.var())
sd
```

Out[28]:
```
8.25363556258695
```

In [29]:
```
n = len(M)
n
```

Out[29]:
```
10
```

In [39]:
```
t_value = d/(sd/np.sqrt(n))
t_value
```

Out[39]:
```
2.471237160087679
```

In [40]:
```
from scipy.stats import t
p_value = (1 - t.cdf(abs(t_value), 2*n-1)) * 2
p_value
```

Out[40]:
```
0.023096800526279493
```

```
Given that Alpha is 1%, we accept the null hypothesis because p_value = 2.3% > 1%
```

In [41]:
```
stats.ttest_rel(M1, M2)
# when I use Scipy code for two paired sample t test, however, it looks that the result are different
```

Out[41]:
```
Ttest_relResult(statistic=2.344421419296965, pvalue=0.043702633095373596)
```