Data Mining Assignment 4
Classification Basics
- 7.5*, 7.9, 7.10*
- 8.3, 8.5, 8.7, 8.12, 8.14
- Due 10/13

* changed from previous years

Aradhya Mathur

7.5 Section 7.2.4 presented various ways of defining negatively correlated patterns. Consider Definition 7.3: "Suppose that itemsets X and Y are both frequent, that is, $sup(X) \geq min\_sup$ and $sup(Y) \geq min\_sup$, where min_sup is the minimum support threshold. If $(P(X|Y) + P(Y|X))/2 < \epsilon$, where ϵ is a negative pattern threshold, then pattern $X \cup Y$ is a **negatively correlated pattern**." Design an efficient pattern growth algorithm for mining the set of negatively correlated patterns.

Answer)

Using FP Growth, we generate all frequent itemsets from the database. Both frequent and infrequent patterns have to be stored.

We have to find P(X|Y) and P(Y|X)

We know P(A|B) = P(A∩ *B*) / P(B)

Similarly, P(X|Y) = P(X ∩ Y) / P(Y) and P(Y|X) = P(X ∩ Y) / P(X)

Now we calculate P(X|Y) and P(Y|X)

P(X ∩ Y) = P(Z)

So, we write sup(X), sup(Y) and sup(Z) for P(X), P(Y) and P(Z) respectively.

P(X/Y) = P(X ∩ Y) / P(Y) = sup(Z) / sup(Y)

Similarly, P(Y/X) = P(X ∩ Y) / P(X) = sup(Z) / sup(X)

Now (P(X|Y)+P(Y|X)) / 2 can be written as

= (sup(Z)/sup(Y) + sup(Z)/sup(X))/2

If (sup(Z)/sup(Y) + sup(Z)/sup(X))/2 < ∈ , where ∈ is negative pattern threshold.

Using this we can calculate if X ∪ Y are negatively correlated pattern.

7.9 Section 7.5.1 defined a **pattern distance measure** between closed patterns P_1 and P_2 as

$$Pat\_Dist(P_1, P_2) = 1 - \frac{|T(P_1) \cap T(P_2)|}{|T(P_1) \cup T(P_2)|},$$

where $T(P_1)$ and $T(P_2)$ are the supporting transaction sets of P_1 and P_2, respectively. Is this a valid distance metric? Show the derivation to support your answer.

Answer)

P1 and P2 are two closed patterns

T(P1) and T(P2) are supporting transaction sets.

Pat_Dist (P1,P2) = 1- $\frac{|T(P1)\cap T(P2)|}{|T(P1)\cup T(P2)|}$

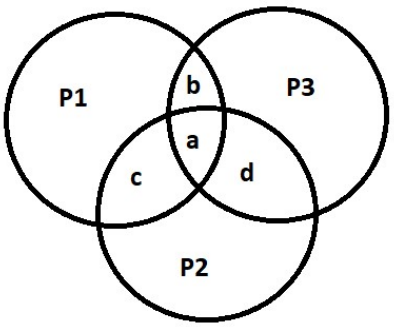
There are four properties to be satisfied:

(a) Dist(P1, P2) > 0 for all P1 ≠ P2
We know, $|T(P1) \cup T(P2)| > |T(P1) \cap T(P2)|$
$\frac{|T(P1)\cap T(P2)|}{|T(P1)\cup T(P2)|} < 1$
$1 - \frac{|T(P1)\cap T(P2)|}{|T(P1)\cup T(P2)|} > 0$, Hence satisfied

(b) Dist(P1, P2) = 0 for all P1 = P2
If P1 = P2 then ∩ *and* ∪ are same and hence numerator = denominator in $\frac{|T(P1)\cap T(P2)|}{|T(P1)\cup T(P2)|}$
This leads to Dist(P1,P2) = 1-1 = 0, Hence satisfied

(c) Dist(P1, P2) = Dist(P2, P1)'
As $|T(P1) \cap T(P2)| = |T(P2) \cap T(P1)|$ and $|T(P1) \cup T(P2)| = |T(P2) \cup T(P1)|$
$\frac{|T(P1)\cap T(P2)|}{|T(P1)\cup T(P2)|} = \frac{|T(P2)\cap T(P1)|}{|T(P2)\cup T(P1)|}$, Hence satisfied

(d) Dist(P1, P2) + Dist(P2, P3) ≥ Dist(P1, P3) for all P1, P2, P3
Basic triangle law which states that sum of any two sides should be greater than the 3rd side.
Using venn diagram here,



$|T(P1) \cap T(P2)| + |T(P1) \cap T(P3)| - |T(P1) \cap T(P2) \cap T(P3)| \leq |T(P1)|$

$c + b - a \leq T(P1)$ this can be easily seen and interpreted through venn diagram. As c+b-a will be part of whole P1, at most it can be equal if there are no distinct elements. This holds for P2 and P3 as well.

We can show this in the form,

$$1 - \frac{|T(P1) \cap T(P2)|}{|T(P1) \cup T(P2)|} + 1 - \frac{|T(P2) \cap T(P3)|}{|T(P2) \cup T(P3)|} \geq \quad 1 - \frac{|T(P1) \cap T(P3)|}{|T(P1) \cup T(P3)|}$$

$T(P1) \cup T(P2) = T(P1) + T(P2) - T(P1) \cap T(P2)$ (Basic Set Property)

$$1 - \frac{|T(P1) \cap T(P2)|}{|T(P1) + T(P2) - T(P1) \cap T(P2)|} + 1 - \frac{|T(P2) \cap T(P3)|}{|T(P2) + T(P3) - T(P2) \cap T(P3)|}$$

$$2 - \frac{|T(P1) \cap T(P2)|}{|T(P1) + T(P2) - T(P1) \cap T(P2)|} - \frac{|T(P2) \cap T(P3)|}{|T(P2) + T(P3) - T(P2) \cap T(P3)|} \geq \text{Dist(P1, P3)}$$

This proves that Dist(P1, P2) + Dist(P2, P3) $\geq$ Dist(P1, P3) , Hence satisfied

All four properties are satisfied

Therefore, Pat_Dist (P1,P2) – $1 - \frac{|T(P1) \cap T(P2)|}{|T(P1) \cup T(P2)|}$ is valid distance metric.

7.10 Association rule mining often generates a large number of rules, many of which may be similar, thus not containing much novel information. Design an efficient algorithm that **compresses** a large set of patterns into a small compact set. Discuss whether your mining method is robust under different pattern similarity definitions.

Answer) Pattern clustering can be used. When we mine association rules from non-transactional data, we might obtain large number of rules. Some of them might not be required or some might be redundant. Using clustering we can efficiently generate association mining rules. We can form general rules for clustering. For example, all the association rules (work_hours = 21) and (full_time = yes) till (work_hours = 39) and (full_time = yes) can be written as (20 < work_hours < 40) =) and (full_time = yes) as per new clustered rules. Clustered rules are useful in reducing huge number of association rules. These rules are easy to understand.

Also, we can use top-k most frequent patterns but it is better to use k most interesting patterns. In k most interesting, patterns are mutually independent, have less redundancy. There are patterns which have high significance along redundancy are called redundancy-aware top-k patterns but there is a trade-off between redundancy and significance.

None of these methods are robust in nature.

8.3 Given a decision tree, you have the option of (a) *converting* the decision tree to rules and then pruning the resulting rules, or (b) *pruning* the decision tree and then converting the pruned tree to rules. What advantage does (a) have over (b)?

Answer) Issue of overfitting is resolved by pruning. Instead of pruning a complete decision node, we can prune a single path as each path will have a different rule. In case root nodes are removed, pruning can occur without considering rebuilding of the tree. Also, rules are simpler to understand. Therefore (a) is better than (b)

8.5 Given a 5-GB data set with 50 attributes (each containing 100 distinct values) and 512 MB of main memory in your laptop, outline an efficient method that constructs decision trees in such large data sets. Justify your answer by rough calculation of your main memory usage.

Answer) This can be solved using RainForest. We will be creating AVC lists. AVC means Attribute-Value, Classlabel. There are total of 50 attributes and for each attribute there will be a AVC list of size 100xC as 100 distinct values are given. 50x100xC will be the total size as there are 50 attributes in total. 512 MB is more than enough to accommodate AVC sets for some value of C. In this we have to take care of units for example, 1 MB = 10^6 Bytes and 1 Byte = 8 bits. Therefore 512 MB is equal to 4096 x 10^6 bits. So, it is very huge for accommodating 50x100xC for any value of C. C will be in bits. And according to our memory maximum value of C can be 819,200 bits which is more than enough.

8.7 The following table consists of training data from an employee database. The data have been generalized. For example, "31 ... 35" for *age* represents the age range of 31 to 35. For a given row entry, *count* represents the number of data tuples having the values for *department, status, age*, and *salary* given in that row.

department	*status*	*age*	*salary*	*count*
sales	senior	31...35	46K...50K	30
sales	junior	26...30	26K...30K	40
sales	junior	31...35	31K...35K	40
systems	junior	21...25	46K...50K	20
systems	senior	31...35	66K...70K	5
systems	junior	26...30	46K...50K	3
systems	senior	41...45	66K...70K	3
marketing	senior	36...40	46K...50K	10
marketing	junior	31...35	41K...45K	4
secretary	senior	46...50	36K...40K	4
secretary	junior	26...30	26K...30K	6

Let *status* be the class label attribute.

(a) How would you modify the basic decision tree algorithm to take into consideration the *count* of each generalized data tuple (i.e., of each row entry)?

(b) Use your algorithm to construct a decision tree from the given data.

(c) Given a data tuple having the values "*systems*," "*26...30*," and "*46–50K*" for the attributes *department, age*, and *salary*, respectively, what would a naïve Bayesian classification of the *status* for the tuple be?

Answer)

a) While calculating information gain, count of each tuple has to be taken into account. Count is also to find most common class.

b) Method 1) Remove count column

Department	**Age**	**Salary**	**Status**
Sales	31..35	46-50	Senior
Sales	26..30	26-30	Junior
Sales	31..35	31-35	Junior
Systems	21..25	46-50	Junior
Systems	31..35	66-70	Senior
Systems	26..30	46-50	Junior
Systems	41..45	66-70	Senior
Marketing	36..40	46-50	Senior
Marketing	31..35	41-45	Junior
Secretary	46..50	36-40	Senior
Secretary	26..30	26-30	Junior

Senior = P = 5

Junior = N = 6

Entropy = $-\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

Using base 2 for log function

Entropy = $-\frac{5}{11}\log\left(\frac{5}{11}\right) - \frac{6}{11}\log\left(\frac{6}{11}\right)$

Entropy = 0.994

For Department

	Senior	Junior	I(Pi, Ni)
Sales	1	2	0.91833
Systems	2	2	1
Marketing	1	1	1
Secretary	1	1	1

I(Pi, Ni) is calculated using $= -\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

Entropy of Department = $\frac{\sum Pi+Ni}{Pi+N}\ I(Pi, Ni)$

= (3*0.91833 + 4*1 + 2*1 +2*1) / 11 = 0.9777

Gain = Entropy – Entropy of Department = 0.994-0.9777 = 0.0163

For Age

	Senior	Junior	I(Pi, Ni)
21..25	0	1	0
26..30	0	3	0
31..35	2	2	1
36..40	1	0	0
41..45	1	0	0
46..50	1	0	0

I(Pi, Ni) is calculated using $= - \frac{P}{P+N} \log \left(\frac{P}{P+N}\right) - \frac{N}{P+N} \log \left(\frac{N}{P+N}\right)$

Entropy of Age = $\frac{\sum Pi+Ni}{Pi+Ni} \; I(Pi, Ni)$
= 4*1 / 11 = 0.36363
Gain = Entropy – Entropy of Age = 0.994-0.3636 = 0.6304

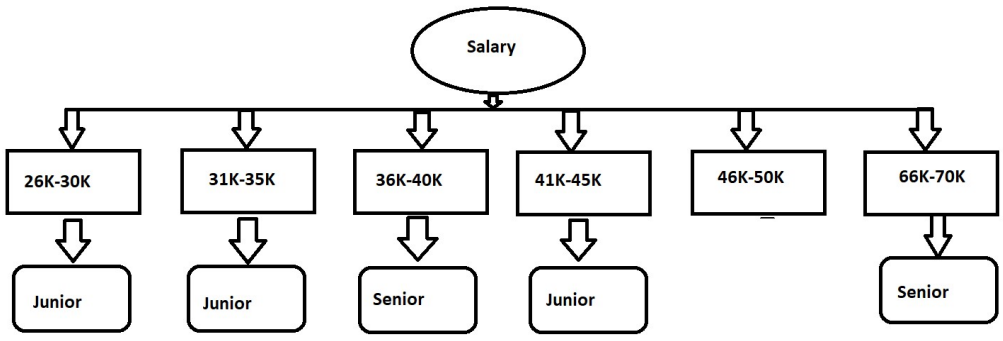
For Salary

	Senior	Junior	I(Pi, Ni)
26..30	0	2	0
31..35	0	1	0
36..40	1	0	0
41..45	0	1	0
46..50	2	2	1
66-70	2	0	0

I(Pi, Ni) is calculated using $= - \frac{P}{P+N} \log \left(\frac{P}{P+N}\right) - \frac{N}{P+N} \log \left(\frac{N}{P+N}\right)$

Entropy of Salary = $\frac{\sum Pi+}{Pi} \; I(Pi, Ni)$
= 4*1 / 11 = 0.36363
Gain = Entropy – Entropy of Salary = 0.994-0.3636 = 0.6304

Gain of both Salary and Age are same
Let's take Salary as root note
Step 1)



For Salary 46K-50K

Table looks like

Department	Age	Status
Sales	31..35	S
Systems	21..25	J
Systems	26..30	J
Marketing	36..40	S

Similarly repeating the steps

Senior = P = 2

Junior = N = 2

Entropy = $-\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

Using base 2 for log function

Entropy = $-\frac{2}{4}\log\left(\frac{2}{4}\right) - \frac{2}{4}\log\left(\frac{2}{4}\right)$

Entropy = 1

For Department

	Senior	Junior	I(Pi, Ni)
Sales	1	0	0
Systems	0	2	0
Marketing	1	0	0
Secretary	0	0	0

I(Pi, Ni) is calculated using $= -\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

Entropy of Department = $\frac{\sum Pi+}{Pi+} I(Pi, Ni)$

= 0

Gain = Entropy – Entropy of Department = 1-0 = 1

For Age

	Senior	Junior	I(Pi, Ni)
21..25	0	1	0
26..30	0	1	0
31..35	1	0	0
36..40	1	0	0

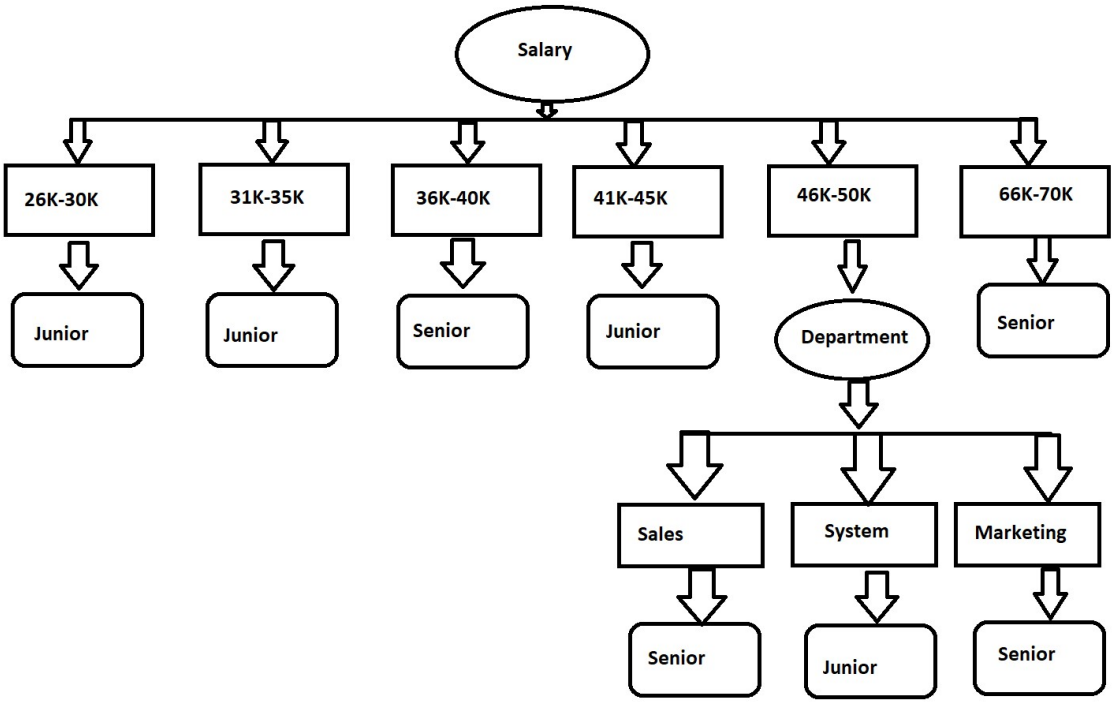
I(Pi, Ni) is calculated using $= -\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

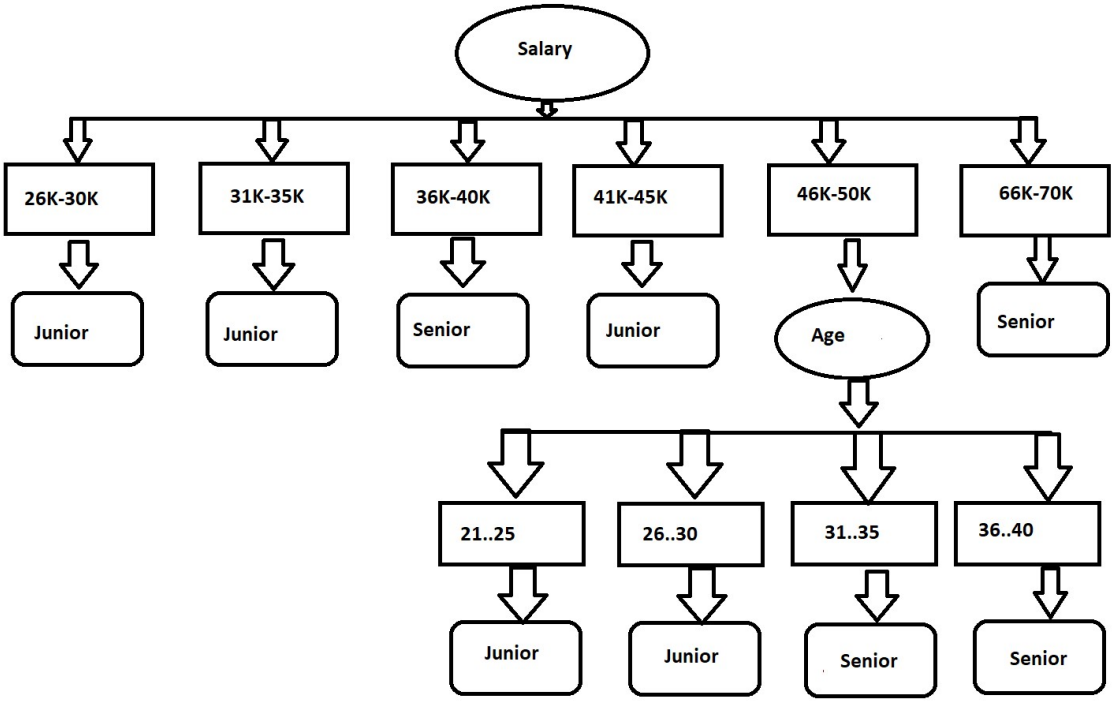
Entropy of Department = $\frac{\sum Pi+Ni}{Pi+N} I(Pi, Ni)$

= 0

Gain = Entropy – Entropy of Age = 1-0 = 1

Gain is same for both





Both decision tree will work efficiently

Method 2) Considering count column:

Department	Age	Salary	Count	Status
Sales	31..35	46-50	30	Senior
Sales	26..30	26-30	40	Junior
Sales	31..35	31-35	40	Junior
Systems	21..25	46-50	20	Junior
Systems	31..35	66-70	5	Senior
Systems	26..30	46-50	3	Junior
Systems	41..45	66-70	3	Senior
Marketing	36..40	46-50	10	Senior
Marketing	31..35	41-45	4	Junior
Secretary	46..50	36-40	4	Senior
Secretary	26..30	26-30	6	Junior

Senior = P = 52

Junior = N = 113

Entropy = $- \frac{P}{P+N} \log (\frac{P}{P+N}) - \frac{N}{P+N} \log (\frac{N}{P+N})$

Using base 2 for log function

Entropy = $- \frac{52}{165} \log (\frac{52}{165}) - \frac{113}{165} \log (\frac{113}{165})$

Entropy = 0.8990

For Department

	Senior	Junior	I(Pi, Ni)
Sales	30	80	0.8453
Systems	8	23	0.8237
Marketing	10	4	0.8439
Secretary	4	6	0.971

I(Pi, Ni) is calculated using = $- \frac{P}{P+N} \log (\frac{P}{P+N}) - \frac{N}{P+N} \log (\frac{N}{P+N})$

Entropy of Department = $\frac{\sum Pi+Ni}{Pi+N} \; I(Pi, Ni)$

= (110*0.8453 + 31*0.8237 + 14*0.8439 +10*0.971) / 165 = 0.8487

Gain = Entropy – Entropy of Department = 0.8990-0.8487 = 0.0503

For Age

	Senior	Junior	I(Pi, Ni)
21..25	0	20	0
26..30	0	49	0
31..35	35	44	0.9905
36..40	10	0	0
41..45	3	0	0
46..50	4	0	0

I(Pi, Ni) is calculated using $= -\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

Entropy of Age = $\frac{\sum Pi+Ni}{Pi+} I(Pi, Ni)$
= 79*0.9905 / 165 = 0.4742
Gain = Entropy – Entropy of Age = 0.8990-0.4742 = 0.4248

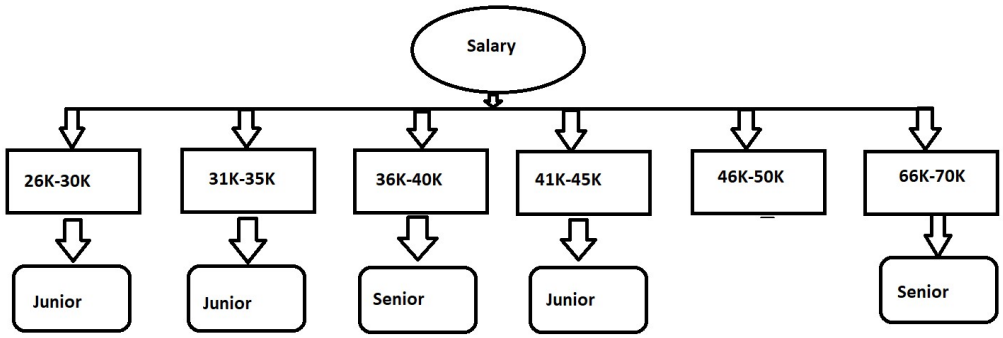
For Salary

	Senior	Junior	I(Pi, Ni)
26..30	0	46	0
31..35	0	40	0
36..40	4	0	0
41..45	0	4	0
46..50	40	23	0.9468
66-70	8	0	0

I(Pi, Ni) is calculated using $= -\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

Entropy of Salary = $\frac{\sum Pi+N}{Pi+Ni} I(Pi, Ni)$
= 63*0.9468 / 165 = 0.3615
Gain = Entropy – Entropy of Salary = 0.8990-0.3615 = 0.5375
Gain of both Salary is highest
We take Salary as root note

Step 1)



For Salary 46K-50K
Table looks like

Department	Age	Count	Status
Sales	31..35	30	S
Systems	21..25	20	J
Systems	26..30	3	J
Marketing	36..40	10	S

Similarly repeating the steps
Senior = P = 40
Junior = N = 23
Entropy = $-\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$
Using base 2 for log function
Entropy = 0.9468

For Department

	Senior	Junior	I(Pi, Ni)
Sales	30	0	0
Systems	0	23	0
Marketing	10	0	0
Secretary	0	0	0

I(Pi, Ni) is calculated using $= -\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

Entropy of Department = $\frac{\sum Pi+Ni}{Pi+Ni}\ I(Pi, Ni)$
= 0
Gain = Entropy – Entropy of Department = 0.9468-0 = 0.9468

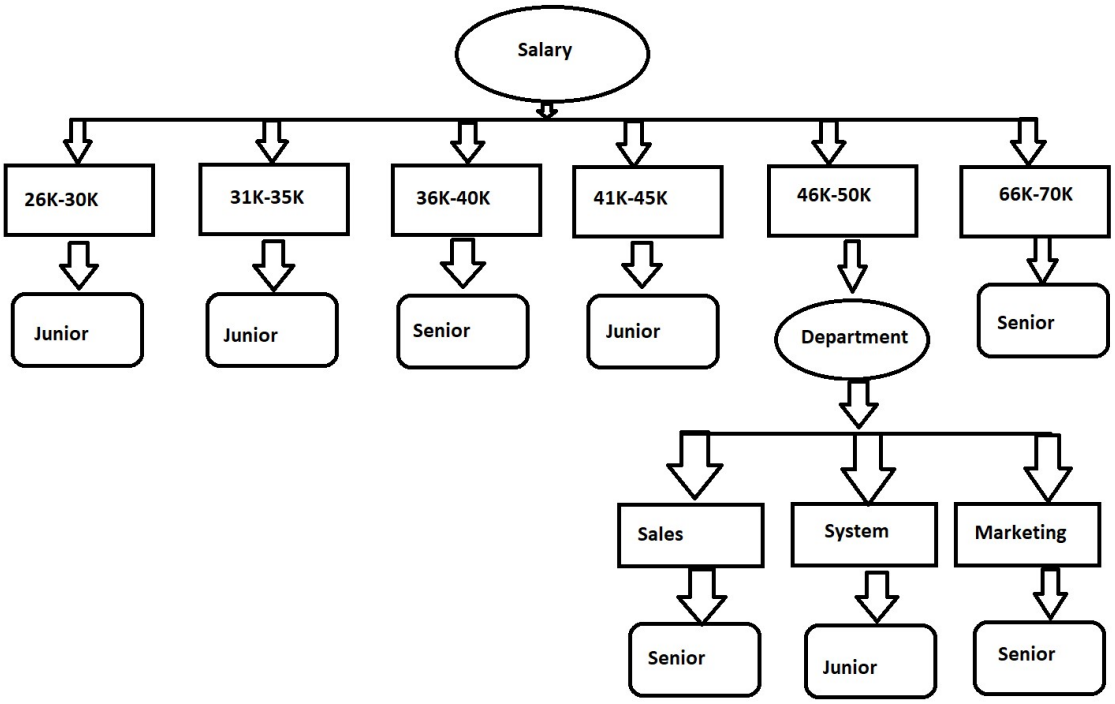
For Age

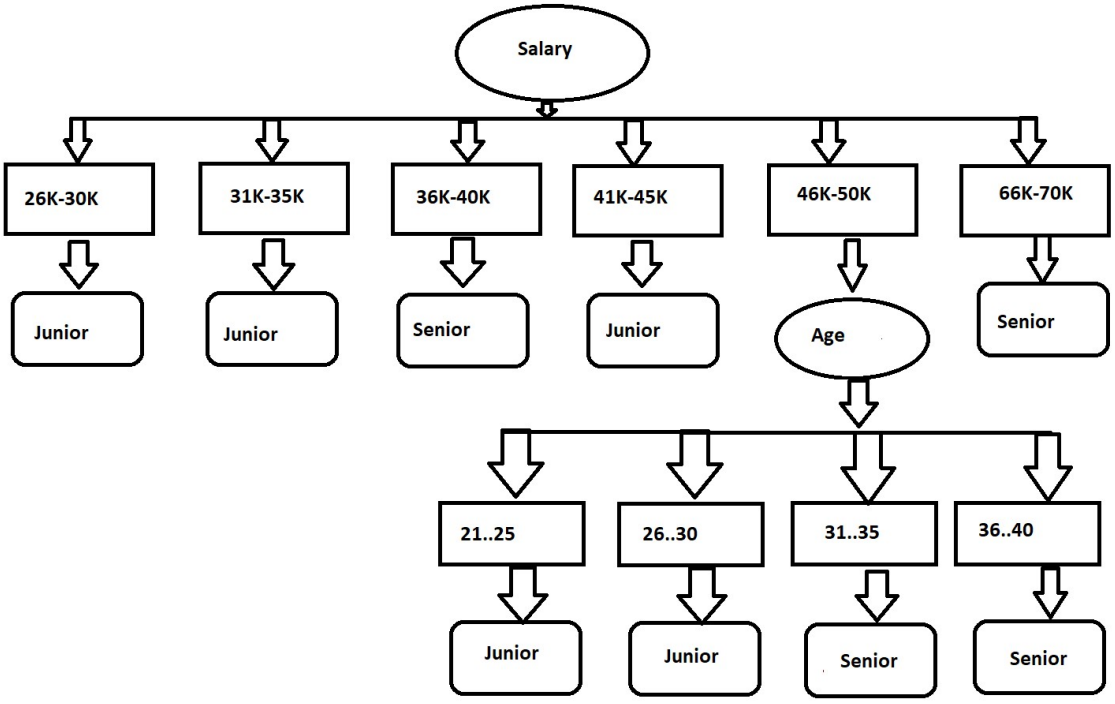
	Senior	Junior	I(Pi, Ni)
21..25	0	20	0
26..30	0	3	0
31..35	30	0	0
36..40	10	0	0

I(Pi, Ni) is calculated using $= -\frac{P}{P+N}\log\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log\left(\frac{N}{P+N}\right)$

Entropy of Department = $\frac{\sum Pi+Ni}{Pi+Ni}\ I(Pi, Ni)$
= 0
Gain = Entropy – Entropy of Age = 0.9468-0 = 0.9468

Gain is same for both





Both decision tree will work efficiently

Only difference between considering count column and not considering count column is that if we consider count column, we get salary as root node with gain significantly more than age while if we don't consider count, we get gain of salary and age same, in that case we can pick either as root node. Final decision tree is same for both methods.

c) Table looks like

department	*status*	*age*	*salary*	*count*
sales	senior	31...35	46K...50K	30
sales	junior	26...30	26K...30K	40
sales	junior	31...35	31K...35K	40
systems	junior	21...25	46K...50K	20
systems	senior	31...35	66K...70K	5
systems	junior	26...30	46K...50K	3
systems	senior	41...45	66K...70K	3
marketing	senior	36...40	46K...50K	10
marketing	junior	31...35	41K...45K	4
secretary	senior	46...50	36K...40K	4
secretary	junior	26...30	26K...30K	6

P(systems | junior) = 23/113
P(26..30 | junior) = 49/113
P(46-50k | junior) = 23/113
P(systems | senior) = 8/52
P(26..30 | senior) = 0 /52
P(46-50K | senior) = 40/52

P(X | junior) = P(systems | junior) x P(26..30 | junior) x P(46-50k | junior)
= 23/113 x 49/113 x 23/113
= 0.01796
P(X | senior) = P(systems | senior) x P(26..30 | senior) x P(46-50k | senior)
= 8/52 x 0/52 x 40/52
= 0.00
Tuple will be classified as junior as P(X | junior) > P(X | senior).

But, P(26..30 | senior) = 0 /52 so we have to apply laplacian correction. Increased count of 1 for each row and considering A is not systems and B is not 46-50

Department	**Age**	**Salary**	**Count**	**Status**
Sales	31..35	46-50	31	Senior
Sales	26..30	26-30	41	Junior
Sales	31..35	31-35	41	Junior
Systems	21..25	46-50	21	Junior
Systems	31..35	66-70	6	Senior
Systems	26..30	46-50	4	Junior
Systems	41..45	66-70	4	Senior
Marketing	36..40	46-50	11	Senior
Marketing	31..35	41-45	5	Junior
Secretary	46..50	36-40	5	Senior
Secretary	26..30	26-30	7	Junior
A	26..30	B	1	Senior

P(systems | junior) = 25/119
P(26..30 | junior) = 52/119
P(46-50k | junior) = 25/119
P(systems | senior) = 10/58
P(26..30 | senior) = 1 /58
P(46-50K | senior) = 42/58

P(X | junior) = P(systems | junior) x P(26..30 | junior) x P(46-50k | junior)
= 25/119 x 52/119 x 25/119
= 0.01928
P(X | senior) = P(systems | senior) x P(26..30 | senior) x P(46-50k | senior)
= 10/58 x 1/58 x 42/58
= 0.00215

Irrespective of Laplacian correction P(X | junior) > P(X | senior)
And hence, Tuple will be classified as junior.

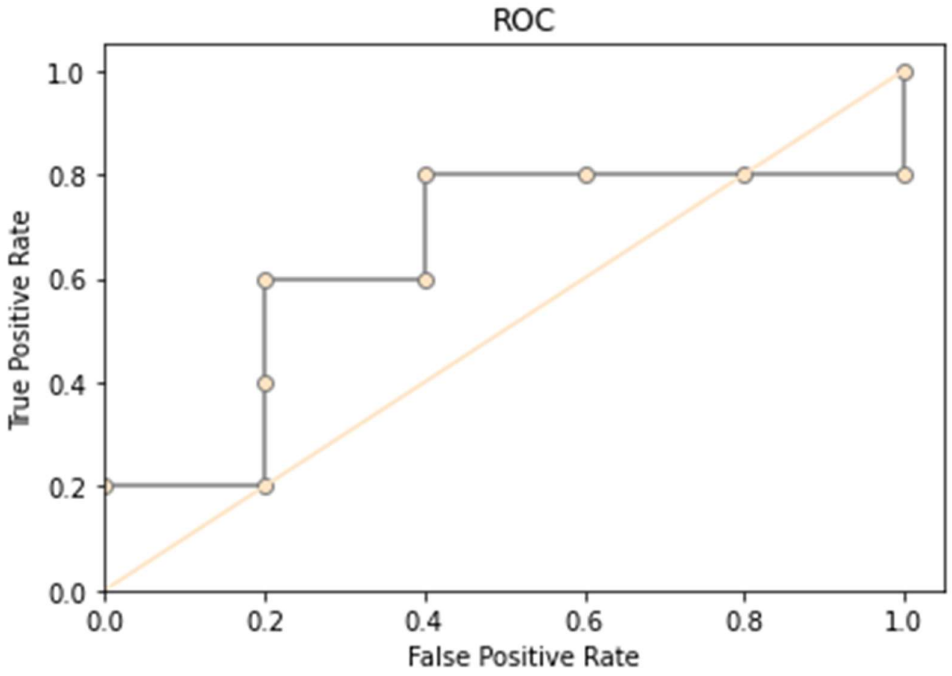
8.12 The data tuples of Figure 8.25 are sorted by decreasing probability value, as returned by a classifier. For each tuple, compute the values for the number of true positives (*TP*), false positives (*FP*), true negatives (*TN*), and false negatives (*FN*). Compute the true positive rate (*TPR*) and false positive rate (*FPR*). Plot the ROC curve for the data.

Tuple #	*Class*	*Probability*
1	*P*	0.95
2	*N*	0.85
3	*P*	0.78
4	*P*	0.66
5	*N*	0.60
6	*P*	0.55
7	*N*	0.53
8	*N*	0.52
9	*N*	0.51
10	*P*	0.40

Figure 8.25 Tuples sorted by decreasing score, where the score is the value returned by a probabilistic classifier.

Answer)

Tuple	Class	Probability	TP	FP	TN	FN	TPR	FPR
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



8.14 Suppose that we want to *select between two prediction models*, M_1 and M_2. We have performed 10 rounds of 10-fold cross-validation on each model, where the same data partitioning in round i is used for both M_1 and M_2. The error rates obtained for M_1 are 30.5, 32.2, 20.7, 20.6, 31.0, 41.0, 27.7, 26.0, 21.5, 26.0. The error rates for M_2 are 22.4, 14.5, 22.4, 19.6, 20.7, 20.4, 22.1, 19.4, 16.2, 35.0. Comment on whether one model is significantly better than the other considering a significance level of 1%.

Answer)

Significance level = 1% = 0.01

Used scipy.stats to perform ttest

```
import scipy.stats as 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])
```

```
stats.ttest_ind(M1, M2, equal_var=True)
```

```
Ttest_indResult(statistic=2.437567141225827, pvalue=0.02538670824824204)
```

Ttest = 2.437567141225827
Pvalue=0.02538670824824204)
Degree of freedom = 9
$\alpha = 0.01$ so $\alpha/2 = 0.005$ and therefore t value corresponding to it is 3.25.
Range is from -3.25 to 3.25 and our ttest value lies between it.
Here pvalue that we got is greater than significance level given.
Therefore, we cannot tell which model is better as we don't have enough evidence.