1. 7.6 Prove that each entry in the following table correctly characterizes its corresponding rule constraint for frequent itemset mining.

	Rule Constraint	Antimonotonic	Monotonic	Succinct
(a)	$v \in S$	no	yes	yes
(b)	$S \subseteq V$	yes	no	yes
(c)	$min(S) \leq v$	no	yes	yes
(d)	$range(S) \leq v$	yes	no	no
(e)	$avg(S) \leq v$	convertible	convertible	no

Prof:

(a)	$v \in S$
Not Antimonotonic	If $v \notin S$, for all supersets of S (put as V, $S \subseteq V$) , we cannot say $v \notin V$ for sure.
Monotonic:	If $v \in S$, then for all supersets of S (put as V, $S \subseteq V$) , $v \in V$, hence is monotonic.
Succinct:	If $v \in S$, then we can directly generate all and only the supersets that have v inside.

(b)	$S \subseteq V$
Antinonotonic:	If $S \nsubseteq V$, then all the supersets of S $\nsubseteq V$.
Monotonic:	If $S \subseteq V$, then we cannot say all the supersets of S $\subseteq V$.
Succinct:	If $S \nsubseteq V$, then we can directly prune all the supersets of S, they$\nsubseteq V$. (? precise)

(c)	$min(S) \leq v$
Not Antimonotonic:	If $min(S) \not\leq v$, then $min(S) > v$, then in all the supersets of S, there may $\exists x \leq v$.
Monotonic:	If $min(S) \leq v$, then in all the supersets of S its minimum elememt $\leq min(S) \leq v$.
Succinct:	If $min(S) \leq v$, then we can directly generate all the supersets that contains min(S).

(d)	$range(S) \leq v$
Antimonotonic:	If $range(S) > v$, then all the supersets of S(put as V), $range(V) > v$.
Not monotonic:	If $range(S) \leq v$, then all the supersets of S, there may $\exists V,\ range(V) > v$.
Not succinct:	Not knowing min(S) and max(S), we cannot generate any sets that >v for sure.

(e)	$avg(S) \leq v$ (If only $avg(S) \leq v$, no other information can extract.)
Convertible: anti-	But if elements are ascending, $avg(S) > v,\ S \subseteq V,\ avg(V) > v$
Convertible: mono-	But if elements are descending, $avg(S) \leq v,\ S \subseteq V,\ avg(V) \leq v$
Not succinct:	Not knowing ascending or descending, $S \subseteq V$, $avg(V)$ may > or < or = v.

2. 7.7 Item price non-negative,

   (a) $BluDVD = v_{blu} \in S_a$; Same with 7.6(a), not antimonotonic, is monotonic, is succinct. How to mine: directly generate all the supersets contains v_{blue}, they are the only sets satisfy the property.

   (b) $sum(I) < 150, I \subseteq S_b$; From table 7.2 on textbook, it is antimonotonic, not monotonic, not succinct. The proof is easy that I will not do. How to mine: if $sum(I) \geq 150$, prune supersets of I.

   (c) $v_c = 0, sum(S_c) \geq 200$; Table 7.2, not antimonotonic, is monotonic, not succinct. How to mine: all supersets of $S_c$satisfy the property, thus should be included.

(d) $100 \leq avg(S_d) \leq 500$; If $avg(S_d) < 100$ ($avg(S_d) > 500$), if data descending(ascending), then all supersets of S_d's average is less than 100(greater than 500), antimonotonic(convertible). If $100 \leq avg(S_d) \leq 500$, no matter how the data look like, we cannot say all the supersets of S_d's average is between 100 and 500 for sure, thus not monotonic. Also, no matter how, we cannot directly generate any sets satisfy this property. How to mine: step1, sort data to two datasets: ascending and descending, for descending test the initial data, if it is less than 100, then whole datasets will not satisfy this property. For ascending, if the very first data is bigger than 500, then then whole dataset will not be satisfying.

The store manager is interested in rules of certain forms that guarantee itemsets which have Blu-ray DVD and its price is also larger than 150. Therefore, first find all the sets that have Blu-ray DVD inside. Then in order to find all sets that $sum(S) \geq 150$ in these Blu-ray DVD contained sets, do mining steps, whenever find a set Si $Sum(S_i) \geq 150$, all the supersets of it will be what we want.

3. 7.9 $Pat\_Dist(P_1,\ P_2) = 1 - \frac{|T(P_1) \cap T(P_2)|}{|T(P_1) \cup T(P_2)|}$, valid distance metric? Derivation.

   Yes.

4. 8.3 Given a decision tree, (a) converting the decision tree to rules and then pruning the resulting rules. (b) pruning the decision tree and then converting the pruned tree to rule. Advantages of (a) over (b)?

   (a) Converting the decision tree to rules and pruning the resulting rules means that we post-prune a fully-grown tree such as removing its branches and replacing them with leafs.

   (b) Pruning the decision tree and then converting the pruned tree to rule means that we pre-prunes the tree by halting its construction early(e.g. by deciding not to further split or partition the subset of training tuples at a given node).

   The advantage of post-pruning(a) over pre-pruning(b) is that post-pruning requires more computation than pre-pruning, yet generally leads to a more reliable tree. Pre-pruning requires split at first, however the goodness of a split may vary depending on how good the pre-specified threshold was put. High thresholds could result in oversimiplified trees, whereas low threshold could result in very little simplification. Post-pruning(a) doesn't have this issue though it requires a bit more computation. Overall, pre-pruning and post-pruning may interleaved for a combined approach, people rarely only use one approach over all others.

5. 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 roughly calculate your main memory usage.

   RainForest is a good scalable decision tree induction method using for large dataset which doesn't fit in memory. If using RainForest, the AVC-set for the root of the tree is about 100(samples) $\times$ C(classes)$\times$50(attributes), which should be fit into 512MB memory when C is not extremely large.

6. 8.7

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**
(sales,systems, marketing,secretary)	(senior, junior)	(21...50,6bins)	(26K...50K)(66K...70K)	165

(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)

The count of each tuple must be integrated into splitting rules. While determine the most common classes among the tuples, the count also need to be taken into consideration.

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

Not easy to plot, I will use arrows to describe:

salary-> (26~30K): junior

->(31~35K): junior

->(36~40K): senior

->(41~45K): junior

->(66~70K):senior

->(46~50K) ->department

department->systems: junior

->others(sales,marketing): senior

(c) "systems, 26...30, 46K...50K" naive Bayesian classification lead to what status?

$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$, naive means assume class-conditioner independence.

i. if junior, $P(junior) = \frac{113}{165}$,
$P(sales|junior) = \frac{80}{113}$, $\underline{P(systems|junior) = \frac{23}{113}}$, $P(marketing|junior) = \frac{4}{113}$, $P(secretary|junior) = \frac{6}{113}$
$P(21 \sim 25|junior) = \frac{20}{113}$, $P(26 \sim 30|junior) = \frac{49}{113}$, $P(31 \sim 35|junior) = \frac{44}{113}$
$P(26 \sim 30K|junior) = \frac{46}{113}$, $P(31 \sim 35K) = \frac{40}{113}$, $P(41 \sim 45K) = \frac{4}{113}$, $\underline{P(46 \sim 50K) = \frac{23}{113}}$

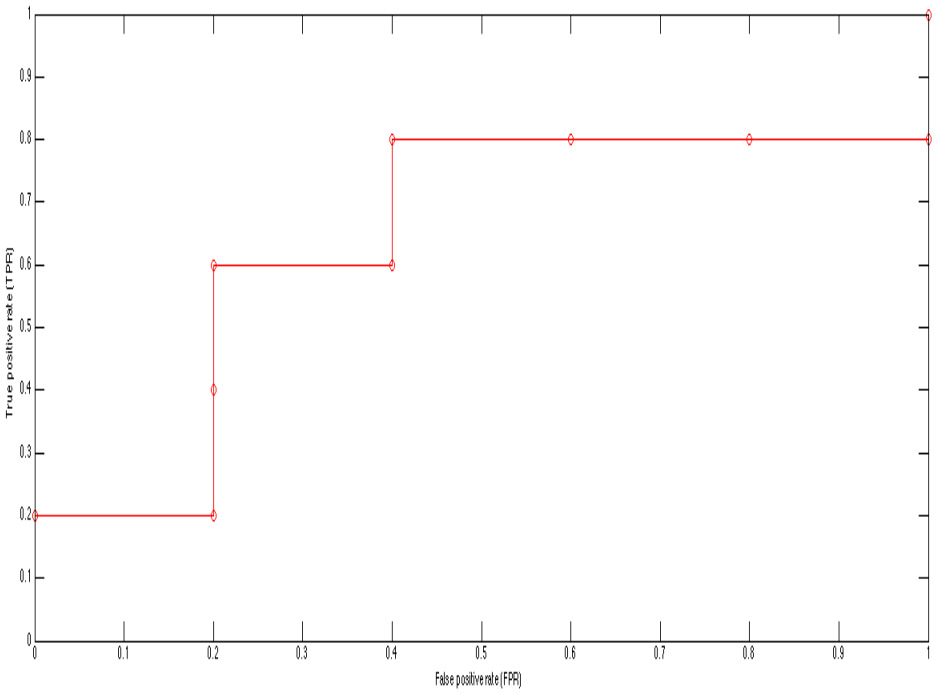
ii. if senior, $P(senior) = \frac{52}{165}$,
$P(sales|senior) = \frac{30}{52}$, $\underline{P(systems|senior) = \frac{8}{52}}$, $P(marketing|senior) = \frac{10}{52}$, $P(secretary|senior) = \frac{4}{52}$
$P(31 \sim 35|senior) = \frac{35}{52}$, $P(41 \sim 45|senior) = \frac{3}{52}$, $P(36 \sim 40|senior) = \frac{10}{52}$, $P(46 \sim 50|senior) = \frac{4}{52}$
$P(36 \sim 40K|senior) = \frac{4}{52}$, $\underline{P(46 \sim 50K|senior) = \frac{40}{52}}$, $P(66 \sim 70K|senior) = \frac{8}{52}$

iii. Taking the probabilities, we obtain that
$P(junior) \times P(systems, 26 \sim 30, 46 \sim 50K|junior) = \frac{113}{165} \times \frac{23}{113} \times \frac{49}{113} \times \frac{23}{113} = 0.0123$
$P(senior) \times P(systems, 26 \sim 30, 46 \sim 50K) = 0$
Therefore, the naive Bayesian classifier predicts JUNIOR for given tuple.

7. 8.12 Fig 8.25 are sorted by decreasing probability value, as returned by a classifer. Compute TP, FP, TN, FN, TPR, FPR. Plot ROC curve.

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	0.60	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	0.40	5	5	0	0	1	1

Figure 1: ROC curve



8. 8.14 Two prediction models, M1 and M2. 10 rounds of 10-fold cross-validation on each model, the same data partitioning in round i is used for both M1 and M2. The error rates for M1 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 M2 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%.

   t-statistic for pairwise comparison: $t = \frac{\bar{err}(M_1)-\bar{err}(M_2)}{\sqrt{var(M_1-M_2)/k}} = \frac{27.72-21.27}{\sqrt{75.6917*9/100}} = \frac{6.45}{2.61} = 2.4713$, we look up table value for z=0.005, t_z = 3.250, -3.250<2.4713<3.250, therefore we accept null hypothesis, no significant difference between M1 and M2.

   t-table n=10, search 9:

Figure 2: t-table

t Table

cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	0.50	0.25	0.20	0.15	0.10	0.05	0.025	0.01	0.005	0.001	0.0005
two-tails	1.00	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01	0.002	0.001
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	0.000	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	0.000	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	0.000	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.144	4.587
11	0.000	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	4.025	4.437
12	0.000	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	3.930	4.318
13	0.000	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	3.852	4.221
14	0.000	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	3.787	4.140
15	0.000	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	3.733	4.073
16	0.000	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	3.686	4.015
17	0.000	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.646	3.965
18	0.000	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.610	3.922
19	0.000	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.579	3.883
20	0.000	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.552	3.850
21	0.000	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.527	3.819
22	0.000	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.505	3.792
23	0.000	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.485	3.768
24	0.000	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.467	3.745
25	0.000	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.450	3.725
26	0.000	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.435	3.707
27	0.000	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.421	3.690
28	0.000	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.408	3.674
29	0.000	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.396	3.659
30	0.000	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.385	3.646
40	0.000	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	3.307	3.551
60	0.000	0.679	0.848	1.045	1.296	1.671	2.000	2.390	2.660	3.232	3.460
80	0.000	0.678	0.846	1.043	1.292	1.664	1.990	2.374	2.639	3.195	3.416
100	0.000	0.677	0.845	1.042	1.290	1.660	1.984	2.364	2.626	3.174	3.390
1000	0.000	0.675	0.842	1.037	1.282	1.646	1.962	2.330	2.581	3.098	3.300
z	0.000	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	3.090	3.291
	0%	50%	60%	70%	80%	90%	95%	98%	99%	99.8%	99.9%
	Confidence Level										