Boston University
Department of Computer Science
CS 565 Data Mining

Midterm Exam
Date: Oct 29, 2007 Time: 4:00 p.m. - 5:30 p.m.

Problem 1 (Data Warehousing and OLAP) [30%]

1.1 **[15%]** Assume that we want to compute a 3-dimensional data cube where attribute A has 5,000 distinct values, attribute B has 1,000 distinct values, and attribute C has 10,000 distinct values. For this, we will use chunking, where each dimension is split into 10 equi-width partitions. Also we assume that the base cuboid ABC is available. If each cube cell stores one measure with 4 bytes, what is the *minimum memory requirement* to compute the data cube with a single scan of the base cuboid? (In other words, you should provide the minimum memory required in order to compute the data cube by reading the base cuboid only once.) Explain your result.

The best order is CAB with minimum memory requirements 26 Mbytes

1.2 **[15%]** Consider a data warehouse with three dimensions (no hierarchies) `city`,`item`, and `year`. The sizes of the cuboids are as shown in the table below:

cuboid	size
{city,item,year}	100
{city,item}	50
{city,year}	75
{item,year}	60
{city}	30
{item}	15
{year}	25
{}	1

Assume that the base cuboid {`city,item,year`} is materialized and consider the following *frequent* queries:

- Q1: Total sales per city and item
- Q2: Total sales per city and year
- Q3: Total sales per item
- Q4: Total sales per year

All queries have equal probability to be asked. Assume that the *linear cost model* applies. For example, the cost to compute the query {`city`} from cuboid {`city,item,year`} is equal to the cost of reading {`city,item,year`}, i.e., 100.

1.2.1 If you can choose only one cuboid from the table to materialize, which one would you choose and why?

You should choose {city, item} with benefit 100

1.2.2 Assume that (i) the base cuboid and (ii) the cuboid of question 1.2.1 are already materialized. If you can choose an extra cuboid to materialize, which one would you choose and why?

You should choose {year} because it gives benefit 75

Problem 2 (Association Rules) [30%]

Consider the following transactional database $\mathcal{D}$. Assume that $min\_support = 40\%$.

TID	items_bought
T100	{ I6, I1, I3}
T200	{ I1, I2, I4, I5, I3}
T300	{ I3, I2, I5}
T400	{ I6, I7}
T500	{ I1, I3, I2, I4, I5}
T600	{ I1, I3, I6}
T700	{ I1, I2, I5, I7}
T800	{ I2, I8, I5, I1}
T900	{ I4, I6}
T1000	{ I1, I2, I5 }

2.1 **[15%]** Apply the **FP-growth** algorithm to generate the frequent itemsets of $\mathcal{D}$. Show the FP-tree and the header table. For each frequent item, show how to generate the conditional pattern bases and conditional FP-trees, and the frequent itemsets generated by them.

Use the FP-growth to find the frquent itemsets

2.2 **[5%]** From the frequent itemsets you have mined, generate all association rules of confidence 100%.

From $I_1I_2I_5 : 5$ *we get* $I_1I_2 \rightarrow I_5$,$I_1I_5 \rightarrow I_2$
From $I_2I_5 : 6$ *we get* $I_2 \rightarrow I_5$,$I_5 \rightarrow I_2$
No more rules with 100% accuracy.

2.3 **[10%]** Consider a sampling approach to mine ALL frequent itemsets. First extract a sample S from the database D, where $S \in D$. Now mine all frequent sets in S at the given min_sup value. Let F_S denote all frequent itemsets in the sample. Now in a single pass over the original database D, we will count the actual support of each $X \in F_s$. After this, for each X we know its support in sample $sup_S(X)$ and its true support in database $sup_D(X)$.

If X is frequent in both S and D then X is a true positive (TP); if X is frequent in S but not in D then X is a false positive (FP); if X is not frequent in S but frequent in D then X is false negative (FN); and if X is not frequent in both S and D then X is true negative (TN).

It is easy to see that our sampling approach can easily find all sets that are TP and FP. As such we do not care about TN, since such sets are not frequent in both the sample and the database. The question is how we can ever find out if we missed any true frequent itemset, i.e., how we can find FN? Show how we can count the support in D for some additional sets derived from F_S, which can then help us determine if we have missed a true frequent set. (Hint: think about the negative border).

Let Y be a frequent itemset in the database, i.e., $Y \in F_D$, but assume Y is a false negative. The main observation is that the negative border of the lattice in the sample consists of all the minimal infrequent itemsets in the sample S. Thus any set Y that is infrequent in S must have a subset X, such that X in the negative border. Thus, we can count the support of the itemsets in the negative border of the sample lattice. If an itemset X in this set is not frequent, then we do not need to check any of its supersets. On the other hand, if X is frequent in D, then we need to expand it and consider its supersets using Apriori. Thus, we will never miss any frequent itemset Y.

Problem 3 (Classification and Clustering) [30%]

3.1 **[15%]** Consider the database of a car insurance company shown below:

Name	AgeGroup	CarType	CrashRisk
Ben	30-40	Family	Low
Paul	20-30	Sports	High
Bill	40-50	Sports	High
James	30-40	Family	Low
John	20-30	Family	High
Steven	30-40	Sports	High

Assume that $CrashRisk$ is the class attribute. Explain which of the remaining attributes are appropriate for classification. Show the complete decision tree that is produced on this dataset. Grow the tree from this root node, until the leaf nodes are "pure", i.e. contain only records from the same class. Explain what will be the class label for nodes with no training samples. Show the split test used at each node. For each leaf node, show the class and the records associated with it. Explain how you derived the split node using the *information gain* and *entropy* concepts.
Using the produced classifier, determine the class label of the following records {Pete, 20-30, Sports} and {Bob, 40-50, Family}.

We will use attributes AgeGroup and CarType for classification.
$I(2,4) = 0.92$
$E(age) = 2/6I(2,0) + 3/6I(2,1) + 1/6I(0,1) = 0.46$
$E(cartype) = 3/6I(2,1) + 3/6I(3,0) = 0.46$
Since $E(cartype) = E(age)$*, we can use any of the two to split first.*
We choose cartype arbitrarily.
If cartype = sports then class=high
If cartype = family then we use age to split futher:
If age=20-30 then high, if age=30-40 then low, if age=40-50 then low (majority voting)
$\{$*Pete, 20-30, Sports*$\}$ *goes to High*
$\{$*Bob, 40-50, Family*$\}$ *goes to Low*

3.2 **[5%]** Explain the concept "class conditional independence assumption" used by the Naive Bayesian Classifiers. Briefly describe the difference between Naive Bayes Classification and Bayesian Belief Networks.

The answer can be found in the chapter 6 in the book

3.3 **[10%]** Consider a two dimensional database $\mathcal{D}$ with the records : $R_1(2,2)$, $R_2(2,4)$, $R_3(4,2)$, $R_4(4,4)$, $R_5(3,6)$, $R_6(7,6)$, $R_7(9,6)$, $R_8(5,10)$, $R_9(8,10)$, $R_{10}(10,10)$. The distance function is the L_1 distance (Manhattan distance). Show the results of the k-means algorithm at each step, assuming that you start with two clusters ($k = 2$) with centers $C_1 = (6,6)$ and $C_2 = (9,7)$.

Use k-means. The first step assigns points 1,2,3,4,5,6, and 8 to $C1$ *and the other points to* $C2$*. The new centers are* $(3.85, 4.85)$ *and* $(9, 8.66)$*. In the next step, point 8 moves from* $C1$ *to* $C2$*. The new centers are* $(3.33, 4)$ *and* $(8, 9)$*. In the next step, point 6 moves from* $C1$ *to* $C2$*. After that move the algorithm stops. The final clusters are points* $(1, 2, 3, 4, 5)$ *and* $(6, 7, 8, 9, 10)$*.*

Problem 4 (Misc) [10%]

True or False:

1. Maximal frequent itemsets are sufficient to determine all frequent itemsets with their supports. *F*

2. High entropy means that the partitions in classification are "pure". *F*

3. A classification model must have 100% accuracy (overall) on the training dataset. *F*

4. The *snowflake* schema saves storage space compared to the *star* schema. *T*

5. All the strong association rules discovered by Apriori are interesting. *F*

— END OF PAPER —