CSC 391/691 – Spring 2014
Test 2 – Take Home
Due – 4pm, Tuesday, 5/6/14

Name __Shuowen Wei_____

THIS WORK MUST BE DONE BY YOU AND YOU ALONE!
No Exceptions.

If you use printed or Internet resources, other than the text, you must indicate the source appropriately. I will try to clarify any questions that you don't understand, but otherwise do your own work.

I. (12 pts) Let's think about the simplest type of clustering, clustering in one dimension on the real number line. The values (data points) we will cluster are 1, 4, 9, 16, 25, 36, 49, 64, 81, and 100, i.e., the squares of 1 through 10. Let's consider using a k-means algorithm, with k=2, or two clusters.

The algorithm tells us to randomly choose two values from the data set as centroids. You can verify that no matter which two points we choose, some prefix of the sequence of squares will go into the cluster of the smaller centroid and the remaining suffix will go into the other cluster. As a result, there are only nine possible clusterings that can be achieved, ranging from {1}{4,9,...,100} through {1,4,...,81}{100}.

Now let's consider the (re)clustering phase where the centroids of the two clusters are (re)calculated and all points are potentially (re)assigned to the nearer of the two new centroids. For which of the nine possible initial clusterings mentioned above, if any, are no points moved from one cluster to the other during the (re)clustering phase?

Answer: Assume such clustering exists, set them to be: $\{1, 4, \dots, n^2\}$ and $\{(n+1)^2, \dots, 100\}$, where n = 1, …10.
Then we have that the centroid for the first cluster is:

$$\frac{\sum_{i=1}^{n} i^2}{n} = \frac{(n+1)(2n+1)}{6}$$

the centroid for the second cluster is:

$$\frac{\sum_{i=1}^{10} i^2 - \sum_{i=1}^{n} i^2}{10-n} = \frac{\frac{10(11)(21)}{6} - \frac{n(n+1)(2n+1)}{6}}{10-n} = \frac{2n^2 + +23n + 231}{6}$$

and they must satisfy that:

$$\begin{cases} n^2 - \frac{(n+1)(2n+1)}{6} < \frac{2n^2 + +23n + 231}{6} - n^2 \\ (n+1)^2 - \frac{(n+1)(2n+1)}{6} > \frac{2n^2 + +23n + 231}{6} - (n+1)^2 \end{cases}$$

solve these two inequalities we obtain:

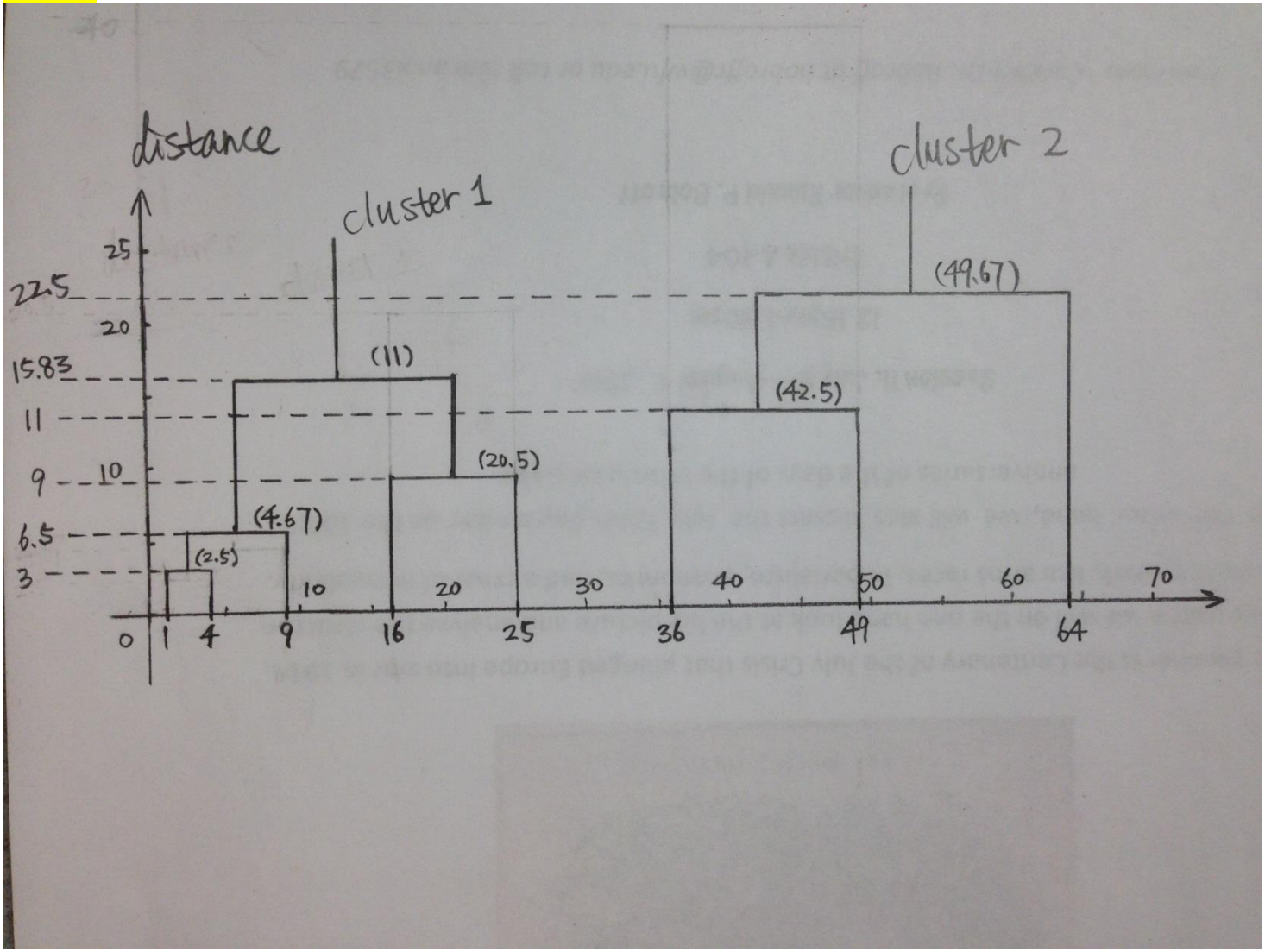
$$5.1 < n < 7.3$$

Hence n = 6 or 7.

The two initial clusters {1,4,9,16,25,36}{49,...,100} and {1,4,9,16,25,36,49}{64,...,100} have no points moved from one cluster to the other after reclustering and the centroid recalculated.

II. (10 pts) Suppose our data set consists of the perfect squares 1, 4, 9, 16, 25, 36, 49, and 64, which are points in one dimension. Perform an agglomerative hierarchical clustering on these points, as follows. Initially, each point is in a cluster by itself. At each step, merge the two clusters with the closest centroids, and continue until only two clusters remain. Show your result as a *dendogram*.

Answer:



III. (6 pts) In certain clustering algorithms, such as CURE, we need to pick a representative set of points in a supposed cluster, and these points should be as far away from each other as possible. That is, begin with the two furthest points, and at each step add the point whose minimum distance to any of the previously selected points is maximum.

Suppose you are given the following points in two-dimensional Euclidean space: x = (0,0); y = (10,10), a = (1,6); b = (3,7); c = (4,3); d = (7,7), e = (8,2); f = (9,5). Obviously, x and y are furthest apart, so start with these. Add four more points to this representative set and list them in the order added. **Use the normal Euclidean L_2-norm as the distance measure.**

Answer:
(1). Representative Set = {x, y}

	x	**y**	**min(x,y)**
a	$\sqrt{37}$	$\sqrt{97}$	$\sqrt{37}$
b	$\sqrt{58}$	$\sqrt{58}$	$\sqrt{58}$
c	$\sqrt{25}$	$\sqrt{85}$	$\sqrt{25}$
d	$\sqrt{98}$	$\sqrt{18}$	$\sqrt{18}$
e	$\sqrt{68}$	$\sqrt{68}$	$\sqrt{68}$
f	$\sqrt{106}$	$\sqrt{26}$	$\sqrt{26}$
maxmin	-	-	$\sqrt{68}$

Then, add point e into the Representative Set, we have Representative Set = {x, y, e}.

(2). Representative Set = {x, y, e}

	x	**y**	**e**	**min(x,y,e)**
a	$\sqrt{37}$	$\sqrt{97}$	$\sqrt{5}$	$\sqrt{5}$
b	$\sqrt{58}$	$\sqrt{58}$	$\sqrt{50}$	$\sqrt{50}$
c	$\sqrt{25}$	$\sqrt{85}$	$\sqrt{17}$	$\sqrt{17}$
d	$\sqrt{98}$	$\sqrt{18}$	$\sqrt{16}$	$\sqrt{16}$
f	$\sqrt{106}$	$\sqrt{26}$	$\sqrt{40}$	$\sqrt{26}$
maxmin	-	-	-	$\sqrt{50}$

Then, add point b into the Representative Set, we have Representative Set = {x, y, e, b}.

(3). Representative Set = {x, y, e, b}

	x	**y**	**e**	**b**	**min(x,y,e,b)**
a	$\sqrt{37}$	$\sqrt{97}$	$\sqrt{65}$	$\sqrt{5}$	$\sqrt{5}$
c	$\sqrt{25}$	$\sqrt{85}$	$\sqrt{17}$	$\sqrt{17}$	$\sqrt{17}$
d	$\sqrt{98}$	$\sqrt{18}$	$\sqrt{26}$	$\sqrt{16}$	$\sqrt{16}$
f	$\sqrt{106}$	$\sqrt{26}$	$\sqrt{10}$	$\sqrt{40}$	$\sqrt{10}$
maxmin	-	-	-	-	$\sqrt{17}$

Then, add point c into the Representative Set, we have Representative Set = {x, y, e, b, c}.

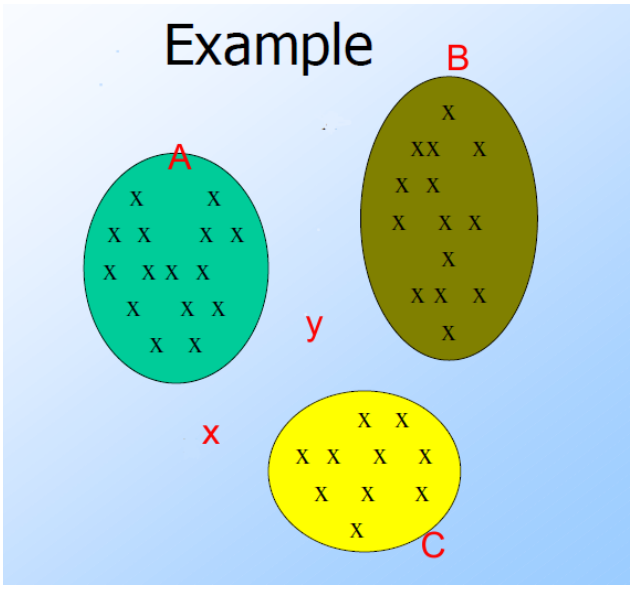
(4). Representative Set = {x, y, e, b, c}

	x	y	e	b	c	min(x,y,e,b,c)
a	$\sqrt{37}$	$\sqrt{97}$	$\sqrt{65}$	$\sqrt{5}$	$\sqrt{18}$	$\sqrt{5}$
d	$\sqrt{98}$	$\sqrt{18}$	$\sqrt{26}$	$\sqrt{16}$	$\sqrt{25}$	$\sqrt{16}$
f	$\sqrt{106}$	$\sqrt{26}$	$\sqrt{10}$	$\sqrt{40}$	$\sqrt{29}$	$\sqrt{10}$
maxmin	-	-	-	-	-	$\sqrt{16}$

Then, add point d into the Representative Set, we have Representative Set = {x, y, e, b, c, d}.

Hence, after adding four more points to this Representative Set, we have {x, y, e, b, c, d} in order.

IV. (12 pts) Think of A, B, C as three clusters identified in an application of the BFR variation of the k-means algorithm. The details are as follows: centroid of A is (2,4), centroid of B is (6, 4.5), centroid of C is (4.5, 1.25); standard deviations of the clusters are as follows, A (1.25 on x axis, 1.5 on y axis), B(1, 1.75), C(1.25, 1). Point **x** is at (2.5, 1.75). Point **y** is at (4,3). [Figure may not be to scale.]



a. Using the standard L_2 Euclidean norm, determine which cluster point **x** is "nearest" to; do the same for point **y**.
b. Using Mahalanobis distance, determine which cluster point **x** is "nearest" to; do the same for point **y**.
c. Does either point change clusters between part (a) to part (b)?
d. Explain why Mahalanobis distance may be a more meaningful distance measure than Euclidean distance in this application.

Answer:

(a). Under the standard L_2 Euclidean norm:

$$\text{dist}(x, A) = \sqrt{(2.5-2)^2 + (1.75-4)^2} = \sqrt{5.3125}$$

$$\text{dist}(x, B) = \sqrt{(2.5-6)^2 + (1.75-4.5)^2} = \sqrt{19.8125}$$

$$\text{dist}(x, C) = \sqrt{(2.5-4.5)^2 + (1.75-1.25)^2} = \sqrt{4.25}, \text{ which is the smallest}$$

$$\text{dist}(y, A) = \sqrt{(4-2)^2 + (3-4)^2} = \sqrt{5}$$

$$\text{dist}(y, B) = \sqrt{(4-6)^2 + (3-4.5)^2} = \sqrt{6.25}$$

$$\text{dist}(y, C) = \sqrt{(4-4.5)^2 + (3-1.25)^2} = \sqrt{3.3125}, \text{ which is the smallest}$$

Thus, both point x and y are nearest to cluster C under the standard L_2 Euclidean norm.

(b). Using Mahalanobis distance, we have:

$$\text{dist}(x, A) = \sqrt{(\frac{2.5-2}{1.25})^2 + (\frac{1.75-4}{1.5})^2} = \sqrt{2.41}, \text{ which is the smallest}$$

$$\text{dist}(x, B) = \sqrt{(\frac{2.5-6}{1})^2 + (\frac{1.75-4.5}{1.75})^2} = \sqrt{14.71939}$$

$$\text{dist}(x, C) = \sqrt{(\frac{2.5-4.5}{1.25})^2 + (\frac{1.75-1.25}{1})^2} = \sqrt{2.81}$$

$$\text{dist}(y, A) = \sqrt{(\frac{4-2}{1.25})^2 + (\frac{3-4}{1.5})^2} = \sqrt{3.0044}, \text{ which is the smallest}$$

$$\text{dist}(y, B) = \sqrt{(\frac{4-6}{1})^2 + (\frac{3-4.5}{1.75})^2} = \sqrt{4.7346}$$

$$\text{dist}(y, C) = \sqrt{(\frac{4-4.5}{1.25})^2 + (\frac{3-1.25}{1})^2} = \sqrt{3.2225}$$

Thus, both point x and y are nearest to cluster A under the Mahalanobis distance.

(c). Yeah, both point x and y change clusters from C to A.

(b). Instead of just measuring the distance from the centroid of each cluster, the Mahalanobis distance also take the deviations of each cluster into account, which can reduce the affection that some cluster might have very "weird shape" in some dimensions, thus Mahalanobis distance is dimensionless.

V. (12 pts) Below is a table representing eight transactions and five items: Beer, Coke, Pepsi, Milk, and Juice. The items are represented by their first letters; e.g., "M" = milk. An "x" indicates membership of the item in the transaction.

	B	C	P	M	J
1	x		x		
2		x		x	
3	x	x			x
4			x	x	
5	x	x		x	
6				x	x
7			x		x
8	x	x		x	x

a. Compute the support for each of the 10 pairs of items. If the support threshold is 2, which of the pairs are frequent itemsets?

Answer: The supports for those 10 pairs are:

support{B, C} = 3 support{B, P} = 1 support{B, M} = 2 support{B, J} = 2
support{C, P} = 0 support{C, M} = 3 support{C, J} = 2
support{P, M} = 1 support{P, J} = 1

support{M, J} = 2
If the threshold is 2, then{B, C}, {B, M}, {B, J}, {C, M}, {C, J} and {M, J} are frequent itemsets.

b. Find all of the rules of the form X→Y, where X and Y are single items (not sets of two or more items), that have confidence exactly 1/2 --- neither more nor less.

Answer: Based on the support of the 10 pairs of items in the question (a):

Since support{B} = 4, then $B \rightarrow M$ and $B \rightarrow J$ have confidence exactly $\frac{1}{2}$

Since support{C} = 4, then $C \rightarrow J$ has confidence exactly $\frac{1}{2}$

Since support{J} = 4, then $J \rightarrow B$, $J \rightarrow C$ and $J \rightarrow M$ have confidence exactly $\frac{1}{2}$

VI. (12 pts) Suppose we perform the PCY algorithm to find frequent pairs, with market-basket data meeting the following specifications:

- *s*, the support threshold, is 10,000.
- There are one million items, which are represented by the integers 0,1,...,999999.
- There are 250,000 frequent items, that is, items that occur 10,000 times or more. (bug???)
- There are one million pairs that occur 10,000 times or more.
- There are *P* pairs that occur exactly once and consist of 2 frequent items.
- No other pairs occur at all.
- Integers are always represented by 4 bytes.
- When we hash pairs, they distribute among buckets randomly, but as evenly as possible; i.e., you may assume that each bucket gets exactly its fair share of the *P* pairs that occur once.

Suppose there are *S* bytes of main memory. In order to run the PCY algorithm successfully, the number of buckets must be sufficiently large that most buckets are not large. In addition, on the second pass, there must be enough room to count all the candidate pairs. As a function of *S*, what is the largest value of *P* for which we can successfully run the PCY algorithm on this data? Demonstrate that you have the correct formula by indicating which of the following is a value for *S* and a value for *P* that is approximately (i.e., to within 10%) the largest possible value of *P* for that *S*.

Possible pairs:

i) S = 500,000,000 P = 10,000,000,000
ii) S = 1,000,000,000 P = 10,000,000,000
iii) S = 500,000,000 P = 5,000,000,000
iv) S = 200,000,000 P = 1,600,000,000
v) S = 300,000,000 P = 3,500,000,000

Discussion:

- Unlike the example in the textbook you don't know how many baskets there are and you don't know how many items are in each basket.
- We're interested in approximations accurate to within 10%. Therefore, if you come up with a value such as S/4 – 1,000,000 in your work, you can simplify that to S/4 since S/4 will be much larger than 1,000,000. In general, any term less than 10% of a sum can be ignored.
- You may want to figure out (approximately) how many of the P *infrequent* pairs will hash to a bucket.

- You may want to figure out (approximately) how many pairs will hash to a *frequent bucket*.
- You may want to figure out (approximately) how many candidate pairs will need to be considered in the second pass.
- As a simplification, ignore the space needed to store the bitmap between passes and ignore the space needed to store the pairs of frequent items on the second pass. Determine the size of the hash table needed in the second pass. That will be the dominant factor.

Answer: Given S to be the bytes of main memory, then in the first pass there can be at most $S/4 - 10^6 \approx S/4$ buckets, and in the second pass there can be at most $C = (S/4 - 10^6)/3 \approx S/12$ candidate pairs.

Assume all the P pairs that consist of 2 frequent items will be hashed into each buckets evenly, then there is going to be $\frac{P}{s/4} = 4P/S$ many infrequent pairs hashed into each buck.

Recall that there are 10^6 pairs that are real frequent pairs, and in each bucket they are hashed in, that bucket has extra $4P/S$ many infrequent pairs.

Thus the total candidate pairs from the first pass will be $10^6 * (4P/S + 1)$, and we know this must be less than $S/12$, thus we have

$$10^6 * (4P/S + 1) < S/12$$

Since $4P/S \gg 1$, then we obtain:

$$P < \frac{S^2}{48 * 10^6}$$

Verifying all the given 5 possible pairs, we conclude that:
i), iv) and v) will not work.
ii) and iii) and will work.

VII. (12 pts) A Web mail service (like gmail, e.g.) has 10^8 users, and wishes to create a sample of data about these users, occupying 10^{10} bytes. Activity at the service can be viewed as a stream of elements, each of which is an email. The element contains the ID of the sender, which must be one of the 10^8 users of the service, and other information, e.g., the recipient(s), and contents of the message. The plan is to pick a subset of the users and collect in the 10^{10} bytes records of length 100 bytes about every email sent by the users in the selected set (and nothing about other users).

[Refer to section Chapter 4] User ID's will be hashed to a bucket number, from 0 to 999,999. At all times, there will be a threshold *t* such that the 100-byte records for all the users whose ID's hash to t or less will be retained, and other users' records will not be retained. You may assume that each user generates emails at exactly the same rate as other users. As a function of *n*, the number of emails in the stream so far, what should the threshold *t* be in order that the selected records will not exceed the 10^{10} bytes available to store records?

Answer: Since there are n emails in the stream and each user generates emails at exactly the same rate as other users, then there are $\frac{n}{10^8}$ emails belong to each user ID. And in average, each buckets have $\frac{10^8}{10^6}$ user IDs, and email occupy 100 bytes, thus we have:

$$t * \left(\frac{10^8}{10^6}\right) * \left(\frac{n}{10^8}\right) * 100 \leq 10^{10}.$$

Then

$$t \leq \frac{10^{14}}{n}$$

VIII. (12 pts) We wish to use the Flagolet-Martin algorithm which uses the number of trailing 0's in the hash value to predict the number of distinct elements in a stream, specifically $2^{\text{trailing zeros in hash}}$ is the estimate. Suppose that there are ten possible elements, 1, 2,..., 10, that could appear in the stream, but only four of actually appear. To make our estimate of the count of distinct elements, we hash each element to a 4-bit binary number. The element *x* is hashed to 3*x* + 7 (modulo 11). For example, element 8 hashes to 3*8+7 = 31, which is 9 modulo 11 (i.e., the remainder of 31/11 is 9). Thus, the 4-bit string for element 8 is 1001.

A set of four of the elements 1 through 10 could give an estimate that is exact (if the estimate is 4), or too high, or too low. Figure out under what circumstances a set of four elements falls into each of those categories. Then, identify in the following list the set of four elements that gives the exact correct estimate of the number of distinct values: { 4, 5, 6, 7 },{ 2, 4, 6, 10 },{ 2, 5, 7, 10 }, { 1, 5, 8, 9 }, {4, 6, 9, 10}, {1, 6, 7, 10 }, {3, 4, 8, 10}

Answer:
(1). After hashing and converting into binary numbers, when the largest number of trailing zeros of those binary numbers is 2, such as $h(10) = 4 = 0100_2$, then it gives an exact estimate; when the largest number of trailing zeros of those binary numbers is 3 or 4, such as $h(4) = 8 = 1000_2$ and $h(5) = 0 = 0000_2$, then it gives a too high estimate; otherwise, when the largest number of trailing zeros of those binary numbers is less then 2, then it gives a too high estimate.

(2). The number of distinct elements for all given sets is 4, and we also observed that when a set contains the elements 4 or 5, whose hash value will be $h(4) = 8 = 1000_2$ and $h(5) = 0 = 0000_2$, then hence it will at least give us an estimate no less than $2^3 = 8$ or even $2^4 = 16$, which is too high. Hence, sets { 4, 5, 6, 7 },{ 2, 4, 6, 10 },{ 2, 5, 7, 10 }, { 1, 5, 8, 9 }, {4, 6, 9, 10}, {3, 4, 8, 10} all contain 4 or 5 and will not give the exact correct estimate.

For the set {1, 6, 7, 10 }, which doesn't contain 4 or 5:

Elements	Hash-value	Binary	Trailing zeros	Estimate
1	10	1010	1	2
6	3	0011	0	0
7	6	0110	1	2
10	4	0100	2	4

Max(estimate) = 4, and yes, the set {1, 6, 7, 10 } gives an exact estimate.

(**UNDERGRADS ONLY**, 12 pts)
Here is a matrix representing the signatures of seven columns of data, C1 through C7.

C1	C2	C3	C4	C5	C6	C7
1	2	1	1	2	5	4

2	3	4	2	3	2	2
3	1	2	3	1	3	2
4	1	3	1	2	4	4
5	2	5	1	1	5	1
6	1	6	4	1	1	4

Suppose we use locality-sensitive hashing with three bands of two rows each. Assume there are enough buckets available that the hash function for each band can be the identity function (i.e., columns hash to the same bucket if and only if they are identical in the band). Find all the candidate pairs.

(**GRAD STUDENTS ONLY**, 12 pts)

This question relates to a tiny Bloom filter. It uses an array of 10 bits and two independent hash functions f and g. We want to test membership in a set S of three elements, so we hash each of the three elements using both f and g, and we set to 1 any bit that any of the three elements is hashed to by either of the hash functions. [Think about what states the array could be in after this step.]

When a new element x arrives, we compute f(x) and g(x), and we say x is in the set S if both f(x) and g(x) are 1. Assume x is not in the set S. What is the probability of a false positive; i.e., the probability of saying that x is in S when it is not? Choose one of these answers and explain your reasoning: 0.121, 0.178, 0.220, 0.282, 0.360, 0.780.

Answer:the formula for the computing the false positive of Bloom filter is $[1-\left(1-\frac{1}{x}\right)^{y}]^{k}$ and $y=km$, where k $=2$, the number of hash functions, $m=3$, the number of elements in Set S, and $x=n=10$, the length of the array.

Hence, $y=km=6$ and then false positive is 0.220.