CS412: Introduction to Data Mining, Fall 2017, Homework 2

Name: Nestor Alejandro Bermudez Sarmiento (nab6)

Worked individually

Assignment 2 deals with concepts of data cubes: effectively counting how many cuboids are in a full or iceberg data cube, how many cells are there in a given cuboid and finding max patterns and the confidence of given association rules.

Question 1

Assume that a base cuboid of 10 dimensions has the following 3 base cells:
$(a_1, a_2, a_3, c_4, c_5, ..., c_{10})$, $(b_1, b_2, b_3, c_4, c_5, ..., c_{10})$ and $(c_1, c_2, c_3, c_4, c_5, ..., c_{10})$
where $a_i \neq b_i, b_i \neq c_i, c_i \neq a_i$ for $i = 1, 2, 3$. There is no dimension with concept of hierarchy. The measure of the cube is count. The count of each base cell is 1.

a. There are **1024** cuboids in the full data cube.

This is easy to see from the fact that there are 10 dimensions. In this case the fact that there are a_i, b_i, c_i values in the first 3 dimensions doesn't matter. The number of cuboids only depend on the number of dimensions and it is given by the relation: 2^n where n is the number of dimensions. As we saw in class this can be seen as the sum of the different ways to choose k dimensions from the n:

$$\sum_{k=0}^{n} \binom{n}{k} = (1+1)^n = 2^n$$

.

b. Distinct aggregated cells in the complete data cube: **2813**.

Note that the first base cell can generate:

$$(2^{10} - 1)$$

aggregated cells and the same is true for the second and third base cell.
We can look at the aggregated cells for each of the base cells as a set. Lets call them A, B and C the set of aggregated cells generated by the base cells that start with a_1, b_1 and c_1 respectively. We want to count the number of unique (distinct) cells in the union of those sets.
Using the inclusion-exclusion property:

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

where $|A|, |B|$ and $|C|$ is $2^{10} - 1 = 1023$. And lets calculate the other terms in the equation above.
Note that for an aggregated cell to be in both A and B the first 3 dimensions have to be starred. Otherwise it will make them unique and it wouldn't be on both sets. Since the first 3 dimensions are stars that leaves us with 7 dimensions. This will generate 2^7 aggregated cells. The same argument applies for $|A \cap C|$ and $|B \cap C|$.
It is easy to see that it also applies for $|A \cap B \cap C|$ since they share all last 7 dimensions.

Putting everything together the total number of distinct aggregated cells is:

$$3(2^{10} - 1) - 2^7 - 2^7 - 2^7 + 2^7 = 3(1023) - 128 - 128 = 2813$$

c. An iceberg cube will contain **128** distinct aggregated cells, if the condition of the iceberg cube is count > 2.

In the previous item I used the argument that keeping the first 3 dimensions without a star will generate distinct cells so we need to star all 3 of them. That leaves us with 7 other dimensions we can star. So the number of cells is $2^7 = 128$.
Note that in this case we don't subtract 1 because the "base cell" of 7 dimensions is actually an aggregated cell with the first 3 dimensions starred.
For example: $(*, *, *, c_4, c_5, ..., c_{10})$

d. The closed cell with count = 3 has **7** non-star dimensions.

Just like for the previous items, this comes down to starring the first 3 dimensions. Also note that after doing so all the other 7 dimensions are the same so its count will be 3. This is the aggregated cell: $(*, *, *, c_4, c_5, ..., c_{10})$. And, as you can see, it has 7 non-star dimensions.

Question 2

This question aims to provide you a better understanding of measures as well as cuboid structures. The provided dataset has 50 transactions with the following fields: *Business id, State, City, Category, Price, Rating.* We are asked to construct a cube over four dimensions: *Location, Category, Price, Rating* with count as a measure. Note that in the Location dimension, there is a concept of hierarchy.

I decided to just SQL for answering this question so I didn't have to create some aggregation logic in the program I decide to use. I did create a small Python script that takes the CSV data file and imports its content into my local SQL server (MySQL). You can find it in the zip file under the name of *q2.py.*
From there on I just perform queries against the database with the appropriate *group by* clauses.

a. There are **24** cuboids in the cube.

   In chapter 4 of the textbook, Equation 4.1 says:

$$Total\ number\ of\ cuboids = \prod_{i=1}^{n}(L_i + 1)$$

   where L_i is the number of levels associated with the dimension i. In this particular case we have four dimensions, the Location dimension has 2 levels and the others only one. By using the previous formula we have:

$$Total\ number\ of\ cuboids = (2 + 1)(1 + 1)(1 + 1)(1 + 1) = 24$$

b. There are **48** cells in the cuboid *(Location(city), Category, Price, Rating.*

   Now that the transactions are imported into the SQL server this is easily answered by running this query:

```
SELECT count(*) as total
FROM (
  SELECT city, category, rating, price, count(*) as support
  FROM cs412_hw2.q2
  GROUP BY city, category, rating, price
) as cuboid
```

   The number of records of the inner *SELECT* are the different cells inside that cuboid and **support** is the count of each cell.

c. Lets drilling up the by climbing up in the Location dimension, from City to State. There are **34** cells in the cuboid *(Location(state), Category, Price, Rating.*

   To get this number it is only necessary to group by the mentioned dimensions.

```
SELECT count(*) as total
FROM (
  SELECT state, category, rating, price, count(*) as support
  FROM cs412_hw2.q2
  GROUP BY state, category, rating, price
) as cuboid
```

d. There are **23** cells in the cuboid *(*, Category, Rating, Price)*.

Note that this is basically just grouping by the non-star dimensions.

```
SELECT count(*) as total
FROM (
  SELECT category, rating, price, count(*) as support
  FROM cs412_hw2.q2
  GROUP BY category, rating, price
) as cuboid
```

e. The count for the cell *(Location(state) = ’Illinois’, *, Rating = 3, Price = ’moderate’)* is **2**.

For this it is just necessary to filter by the given conditions and group by the same dimensions, note that there is no grouping by Category since it is starred.

```
SELECT state, rating, price, count(*)
FROM cs412_hw2.q2
WHERE state = ’Illinois’ AND rating = 3 AND price = ’moderate’
GROUP BY state, rating, price
```

f. The count for the cell *(Location(city) = ’Chicago’, Category = ’food’, *, *)* is **2**.

Just like in the previous item, I just need to filter the transactions based on the given conditions and then group by the non-starred dimensions (City and Category).

```
SELECT city, category, count(*)
FROM cs412_hw2.q2
WHERE city = ’Chicago’ AND category = ’food’
GROUP BY city, category
```

Question 3

For this question we were given a dataset that contains 100 transactions. Each transaction contains items separated by spaces. I'll do some pattern mining in the data to answer the questions below.
I decided to do the pattern filtering using the Apriori method. You can find the implementation under the *q3.py* file.
The Apriori method is an iterative process that starts with finding the frequent 1-itemset, prunning the ones below the *minsup* threshold and then it generates some candidate patterns from the previous k-itemset, checks that the patterns are present in the transactional database and if so it will count the number of patterns and start the process again until there is no pattern left.

Once the frequent patterns (for all k size) are found it is easy to count them to answer item (a) and (b).
To answer item (c) I loop through the possible values of k, for each value of k I look at the itemsets of at least $k+1$ items trying to find a superset. If a superset is found, the current pattern is disregarded as I cannot be a max pattern. My Python program also prints out the actual patterns but it is not included in the report for brevity.
Finally, to answer items (d) and (e) we know that we can use the following formula:

$$c = \frac{sup\{X \cup Y\}}{sup\{X\}}$$

For item (d), $X = (C, E)$ and $Y = (A)$ so it follows that $X \cup Y = (C, E, A)$.
For item (e), $X = (A, B, C)$ and $Y = (E)$ so it follows that $X \cup Y = (A, B, C, E)$.

All the necessary support values have already been calculated when finding all the frequent patterns so my function only finds the right value based on the length of the set. For item (e) turns out that (A, B, C, E) is not a frequent pattern and then the confidence is 0.

a. Suppose the minimum support is 20.

   (a) The number of frequent patterns is **18**
   (b) The number of frequent patterns with length 3 is **5**
   (c) The number of max patterns is **6**

b. Suppose the minimum support is 10.

   (a) The number of frequent patterns is **24**
   (b) The number of frequent patterns with length 3 is **9**
   (c) The number of max patterns is **6**
   (d) The confidence measure of the association rule $(C, E) \rightarrow A$ is **0.769**
   (e) The confidence measure of the association rule $(A, B, C) \rightarrow E$ is **0.742**