COMP9318

17S1 Assignment1

Data Warehousing and Data Mining

Z5044541

Wangyujie

Q1: The tuples in the complete data cube of R in a tabular form:

|  |  |  |  |
| --- | --- | --- | --- |
| Location | Time | Item | SUM(quality) |
| Sydney | 2005 | PS2 | 1400 |
| Sydney | 2006 | PS2 | 1500 |
| Sydney | 2006 | Wii | 500 |
| Melbourne | 2005 | Xbox 360 | 1700 |
| Sydney | 2005 | ALL | 1400 |
| Sydney | 2006 | ALL | 2000 |
| Melbourne | 2005 | ALL | 1700 |
| Sydney | ALL | PS2 | 2900 |
| Sydney | ALL | Wii | 500 |
| Melbourne | ALL | Xbox 360 | 1700 |
| ALL | 2005 | PS2 | 1400 |
| ALL | 2006 | PS2 | 1500 |
| ALL | 2006 | Wii | 500 |
| ALL | 2005 | Xbox360 | 1700 |
| Sydney | ALL | ALL | 3400 |
| Melbourne | ALL | ALL | 1700 |
| ALL | 2005 | ALL | 3100 |
| ALL | 2006 | ALL | 2000 |
| ALL | ALL | PS2 | 2900 |
| ALL | ALL | Wii | 500 |
| ALL | ALL | XBOX360 | 1700 |
| ALL | ALL | ALL | 5100 |

Q2: The SQL statement:

SELECT Location, Time, Item, SUM(Quality) FROM Sales(R)

GROUP BY Location, Time, Item

UNION ALL

SELECT Location, Time, ALL, SUM(Quality) FROM Sales(R)

GROUP BY Location, Time

UNION ALL

SELECT Location, ALL, Item, SUM(Quality) FROM Sales(R)

GROUP BY Location, Item

UNION ALL

SELECT ALL, Time, Item, SUM(Quality) FROM Sales(R)

GROUP BY Time, Item

UNION ALL

SELECT Location, ALL , ALL , SUM(Quality) FROM Sales(R)

GROUP BY Location

UNION ALL

SELECT ALL , Time, ALL, SUM(Quality) FROM Sales(R)

GROUP BY Time

UNION ALL

SELECT ALL, ALL, Item, SUM(Quality) FROM Sales(R)

GROUP BY Item

UNION ALL

SELECT ALL, ALL, ALL, SUM(Quality) FROM Sales(R)

UNION ALL

Q3: Due to the SQL statements , we only need to find the cells of data cube in aggregate condition, so the result tabular is :

|  |  |  |  |
| --- | --- | --- | --- |
| Location | Time | Item | SUM(Quality) |
| Sydney | 2006 | ALL | 2000 |
| Sydney | ALL | PS2 | 2900 |
| Sydney | ALL | ALL | 3400 |
| ALL | 2005 | ALL | 3100 |
| ALL | 2006 | ALL | 2000 |
| ALL | ALL | PS2 | 2900 |
| ALL | ALL | ALL | 5100 |

Q4：

The cube contains totally 22 cells ,so the function should be 1 to 1 function so that it can contain all the cells with unique array index :

We assume Function = x \* Location + y \* Time + z\* item ( x, y ,z >=1 ),then we look at the last five rows (0,0,0),(0,0,1),(0,0,2),(0,0,3),(0,1,0) etc. What we want is to give an appropriate x, y ,z to represent index, then we assume (0 , 0 ,0 )= 0 (0, 0, 1) =1 ( 0,0 ,2) =2 ,(0,0,3) =3 ,(0,1 ,0 ) = 4 , we can get y = 4 an z = 1, then we should give x an appropriate number to enlarge the index size . the maximum of 4\*time + Item = 4\*2 +3 =11 , so if x =12, the function seems satisfy the requests.

So the function : 12 \* Location + 4\* time + 1\* item

The last final

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Location | Time | Item | SUM(quality) | Function |
| 1 | 1 | 1 | 1400 | 17 |
| 1 | 2 | 1 | 1500 | 21 |
| 1 | 2 | 3 | 500 | 23 |
| 2 | 1 | 2 | 1700 | 30 |
| 1 | 1 | 0 | 1400 | 16 |
| 1 | 2 | 0 | 2000 | 20 |
| 2 | 1 | 0 | 1700 | 28 |
| 1 | 0 | 1 | 2900 | 13 |
| 1 | 0 | 3 | 500 | 15 |
| 2 | 0 | 2 | 1700 | 26 |
| 0 | 1 | 1 | 1400 | 5 |
| 0 | 2 | 1 | 1500 | 9 |
| 0 | 2 | 3 | 500 | 11 |
| 0 | 1 | 2 | 1700 | 6 |
| 1 | 0 | 0 | 3400 | 12 |
| 2 | 0 | 0 | 1700 | 24 |
| 0 | 1 | 0 | 3100 | 4 |
| 0 | 2 | 0 | 2000 | 8 |
| 0 | 0 | 1 | 2900 | 1 |
| 0 | 0 | 3 | 500 | 3 |
| 0 | 0 | 2 | 1700 | 2 |
| 0 | 0 | 0 | 5100 | 0 |

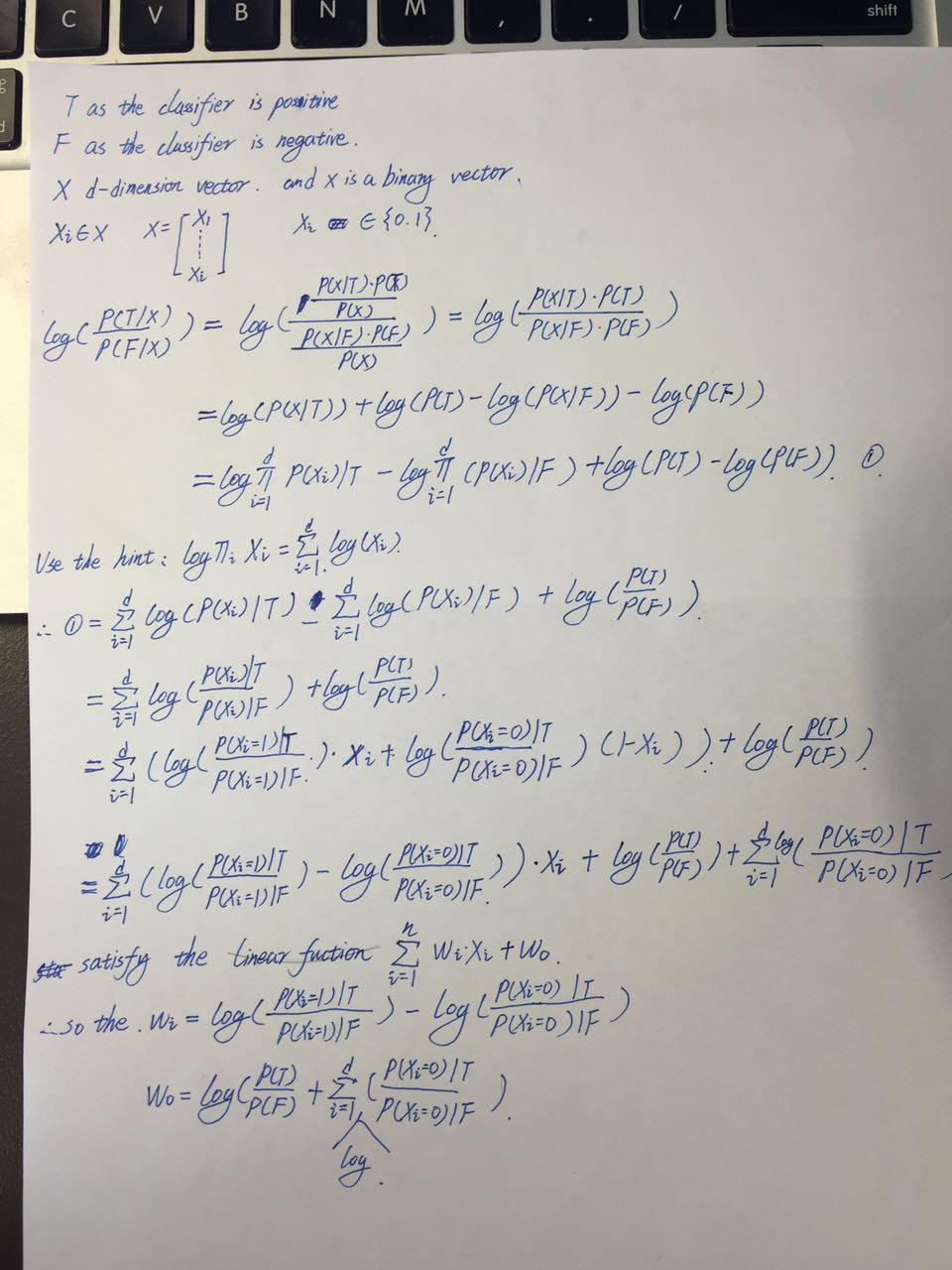
The index result satisfy the 1 to 1 request, the maximum index is 30 and the total number is 22 , only 8 cells are not used . so this function works

So the result :

|  |  |
| --- | --- |
| Index | SUM（Quality） |
| 0 | 5100 |
| 1 | 2900 |
| 2 | 1700 |
| 3 | 500 |
| 4 | 3100 |
| 5 | 1400 |
| 6 | 1700 |
| 8 | 2000 |
| 9 | 1500 |
| 11 | 500 |
| 12 | 3400 |
| 13 | 2900 |
| 15 | 500 |
| 17 | 1400 |
| 20 | 2000 |
| 21 | 1500 |
| 23 | 500 |
| 24 | 1700 |
| 26 | 1700 |
| 28 | 1700 |
| 30 | 1700 |

2:In Naïve Bayes classifier, the connection bet ween two can be expressed by log-odds( logit ) , so we assume the log-odds function, and if the log-odds function is larger than 0 , we define the classifier positive , otherwise if the log-odds function is smaller than 0 , we define the classifier negative.

so as the function in the function: we get the result W



Q2 ： In this question, we only contains the those four situation

P(x=0|y=0) ,P(x=0|y =1 ), P(x=1 | y=0),P(x = 1 | y=1 )

Naïve Bayes uses relatively strict conditional independent hypothesis. Each condition must be independent. Naïve Bayes generate the model so that it need less model than logistic regression ,In this question , the complexity of naïve Bayes is O(log(n))

Logistic Regression judge the model . Each condition do not need to be independent.

LR method need to continuously adjust the arguments to satisfy the function distribution ,then training to get an optimal model under the existing data set. So each attributes are learned together, the complexity of Logistic Regression is O(n)

So Naïve Bayes is much easier than Logistic Regression

3: Q3(1):

Data: D is a dataset of n d-dimensional points; k is the number of clusters.

1 Initialize k centers C = [c1, c2, . . . , ck];

2 canStop ← false;

3 while canStop = false do

4 Initialize k empty clusters G = [g1, g2, . . . , gk];

5 for each data point p ∈ D do

6 cx ← NearestCenter(p, C);

7 gcx .append(p);

8 canStop = true

9 for each group g ∈ G do

10 temp = ci

11 ci ← ComputeCenter(g);

12 if ci != temp:

13 canStop = False;

14 return G

Q3.(2):

First of all, in line 6 (cx ← NearestCenter(p, C);) , it proved that each point are classified to the nearest cluster. Each classifier choose the minimal dist^2(p,c) to classify so the cost in this step is reduced .

In line 9 (ci ← ComputeCenter(g)); line 9 computes the new center point of each C . In this situation, all the points have been re-classified and already found the nearest Current center. It is obvious that it can not increase the cost.

The what we need to do is as that function:



the C(x ,y ) is the min value of new centers which is converged by K-means at the end of each iteration

Q3 (3)

Due to Q2 we can find the cost never increase at the end of each iteration , so the function is Monotonically decreasing and we also proved its convergence, so that it must have a lower bound and it must has local minimal.