COMP9318 - Assignment 1

Tianwei Zhu, z5140081

Q1

1. *List the tuples in the complete data cube of R in a tabular form with 4 attributes.*

|  |  |  |  |
| --- | --- | --- | --- |
| Location | Time | Item | Sum(Quantity) |
| Sydney | 2005 | PS2 | 1400 |
| Sydney | 2006 | PS2 | 1500 |
| Sydney | 2006 | Wii | 500 |
| Sydney | 2005 | \* | 1400 |
| Sydney | 2006 | \* | 2000 |
| Sydney | \* | PS2 | 2900 |
| Sydney | \* | Wii | 500 |
| Sydney | \* | \* | 3400 |
| Melbourne | 2005 | Xbox360 | 1700 |
| Melbourne | 2005 | \* | 1700 |
| Melbourne | \* | Xbox360 | 1700 |
| Melbourne | \* | \* | 1700 |
| \* | 2005 | PS2 | 1400 |
| \* | 2005 | Xbox360 | 1700 |
| \* | 2005 | \* | 3100 |
| \* | 2006 | PS2 | 1500 |
| \* | 2006 | Wii | 500 |
| \* | 2006 | \* | 2000 |
| \* | \* | PS2 | 2900 |
| \* | \* | Wii | 500 |
| \* | \* | Xbox360 | 1700 |
| \* | \* | \* | 5100 |

1. *Write down an equivalent SQL statement that computes the same result*

**SELECT** Location, Time, Item, SUM(Q) **AS** Total

**FROM** Table1

**GROUP** **BY** Location, Time, Item **WITH ROLLUP**

**UNION**

**SELECT** Location, (null) **AS** Time, Item, SUM(Q) **AS** Total

**FROM** Table1

**GROUP** **BY** Location, Item **WITH ROLLUP**

**UNION**

**SELECT** Location, Time, (null) **AS** Item, SUM(Q) **AS** Total

**FROM** Table1

**GROUP** **BY** Location, Time **WITH ROLLUP**

**UNION**

**SELECT** (null) **AS** Location, Time, Item, SUM(Q) **AS** Total

**FROM** Table1

**GROUP** **BY** Time, Item **WITH** **ROLLUP**

**UNION**

**SELECT** (null) **AS** Location, Time, Item, SUM(Q) **AS** Total

**FROM** Table1

**GROUP** **BY** Item, Time **WITH ROLLUP**

1. *Draw the result of the query in a tabular form.*

|  |  |  |  |
| --- | --- | --- | --- |
| Location | Time | Item | Sum(Quantity) |
| Sydney | 2006 | \* | 2000 |
| Sydney | \* | PS2 | 2900 |
| Sydney | \* | \* | 3400 |
| \* | 2005 | \* | 3100 |
| \* | 2006 | \* | 2000 |
| \* | \* | PS2 | 2900 |
| \* | \* | \* | 5100 |

1. *Draw the MOLAP cube in a tabular form of (ArrayIndex, Value).*

**F(Location,Time,Item) = 10\*Location + 4\*Time + 1\*Item**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Location | Time | Item | Sum(Quantity) | Offset |
| 1 | 1 | 1 | 1400 | 15 |
| 1 | 2 | 1 | 1500 | 19 |
| 1 | 2 | 3 | 500 | 21 |
| 1 | 1 | 0 | 1400 | 14 |
| 1 | 2 | 0 | 2000 | 18 |
| 1 | 0 | 1 | 2900 | 11 |
| 1 | 0 | 3 | 500 | 13 |
| 1 | 0 | 0 | 3400 | 10 |
| 2 | 1 | 2 | 1700 | 26 |
| 2 | 1 | 0 | 1700 | 24 |
| 2 | 0 | 2 | 1700 | 22 |
| 2 | 0 | 0 | 1700 | 20 |
| 0 | 1 | 1 | 1400 | 5 |
| 0 | 1 | 2 | 1700 | 6 |
| 0 | 1 | 0 | 3100 | 4 |
| 0 | 2 | 1 | 1500 | 9 |
| 0 | 2 | 3 | 500 | 12 |
| 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 |

|  |  |
| --- | --- |
| ArrayIndex | Value |
| 0 | 5100 |
| 1 | 2900 |
| 2 | 1700 |
| 3 | 500 |
| 4 | 3100 |
| 5 | 1400 |
| 6 | 1700 |
| 8 | 2000 |
| 9 | 1500 |
| 10 | 3400 |
| 11 | 2900 |
| 12 | 500 |
| 13 | 500 |
| 14 | 1400 |
| 15 | 1400 |
| 18 | 2000 |
| 19 | 1500 |
| 20 | 1700 |
| 21 | 500 |
| 22 | 1700 |
| 24 | 1700 |
| 26 | 1700 |

Q2

1. *Prove that if the feature vectors are d-dimension, then a Naive Bayes classifier is a linear classifier in a d + 1-dimension space.*

Consider binary classification where y = 1 or 0, A Naïve Classifier is:

To make it simpler, let’s set as , and as .

Then the Classifier will look like:

When implying log to formula (2):

To make it more intuitive, we can use b and w instead of complicated form:

Now, we have a linear classifier, and set parameters as below:

The Naïve Bayes can have a linear decision boundary in feature **x**, and the boundary takes the form of a hyperplane function.

1. *Briefly explain why learning* ***w****NB is much easier than learning* ***w****LR.*

First, we should know that **Naïve Bayes** is built on Conditional independent hypothesis, which means features X1, X2, X3… are independent. We can use statistical methods to calculate the frequency of P(x|y) and P(y), so as to obtain P(x|y) and P(y). In this case, the vectors **w**NB we need to calculate are about approach by **O(logn).**

For **Logistic Regression**, it calculates the whole linear space to generate **w**LR and this makes the complexity become **O(n).**

It is obvious then learning **w**NB is easier than **w**LR, but **Logistic Regression** gives better result than **Naïve Bayes** when the training data is limited.

Q3

1. *Prove the loss function for logistic regression.*

Since , we have

Then the likelihood for training dataset is:

Log-likelihood will be:

1. *Write out its loss function where .*

With the function f, we now have:

Write the likelihood for training set:

Finally, we have loss function for :