Question 1

```
Q1.
1.

Location  Time  Item   SUM(Quantity)
Any       Any   Any    4100

2.

SELECT Location, Time, Item, Sum(Quantity)
          FROM Sales
          GROUP BY Location, Time, Item;

3.

SELECT  Location, Time, Item, SUM(Quantity)
          FROM    Sales
          CUBE BY Location, Time, Item
          HAVING  COUNT(*) > 1

Location  Time  Item  SUM(Quantity)
```

```
4.

Step1:
Location  Time Item Quantity
1         1    1    1400
1         2    1    1500
1         2    3    500
2         1    2    1700
-----------------------------------------------------------
Step2:
f(Location, Time, Item) = 3*Location + 3*Time + 4*Item

Location  Time Item Quantity      Offset
1         1    1    1400          10
1         2    1    1500          13
1         2    3    500           21
2         1    2    1700          17
-----------------------------------------------------------
Step3:
ArrayIndex  Value
10          1400
13          1500
17          1700
21          500
```

Question 3

```
Q3.

1.

Data: D is a dataset of n d-dimensional points; k is the number of clusters.
  Initialize k centers C = [c1, c2, . . . , ck];
  canStop ← false;
  while canStop = false do
    Initialize k empty clusters G = [g1, g2, . . . , gk];
    for each data point p ∈ D do
      cx ← NearestCenter(p, C);
      gcx .append(p);
    C_old ← C;
    C ← [];
    for each group g ∈ G do
      ci ← ComputeCenter(g);
      C.append(ci);
    if C_old == C then
      canStop ← true;
  return G;
```

2.

$$cost(g_i) = \sum_{p \in g_i} dist^2(p, c_i) \quad - (1)$$

$$dist(p, c_i) = \sqrt{\sum_{i=1}^{n} (c_i - p)^2} \quad - (2) \text{ Euclidean dist.}$$

Substituting (2) in (1)

$$cost(g_i) = \sum_{p \in g_i} \left(\sum_{i=1}^{n} (c_i - p)^2 \right)$$

Lemma 1. For any $C \in R^d$ and any $p \in R^d$,

$$cost(C, p) = cost(C, mean(C)) + |C| \cdot \|p - mean(C)\|^2$$

Therefore when $p = mean(C)$, $cost(C, p)$ is minimized

At $c_i \leftarrow ComputeCenter(g)$; we are doing exactly the same. Therefore cost of k cluster ~~of~~ never increases.

3. Convergence - During the course of k-means ~~algo~~ algorithm the cost decreases.

Let $c_1^{(i)} \dots c_k^{(i)}$, $G_1^{(i)}, \dots G_k^{(i)}$ denote centers & clusters at the start of i-iteration. The first step of iteration assigns each data point to its closest center: Therefore: $cost(G_{1:k}^{(i+1)}, c_{1:k}^{(i)}) \le cost(G_{1:k}^{(i)}, c_{1:k}^{(i)})$

On second step, each cluster is re-centered at its mean, by Lemma 1 $cost(G_{1:k}^{(i+1)}, c_{1:k}^{(i+1)}) \le cost(G_{1:k}^{(i+1)}, c_{1:k}^{(i)})$.

Hence Proven.

Question 2

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

Scale document down

Let $x = (x_1, x_2, \ldots, x_d)$. Features x_j are binary.

Our classifier will predict the label 1 if

$$P(y=1|x) \geqslant P(y=0|x).$$

$$\frac{P(x|y=1)\,P(y=1)}{P(x|y=0)\,P(y=0)} \geqslant 1 \quad ----(1)$$

By Naive Bayes assumption $P(x|y) = \prod_{j=0}^{d} (P x_j|y)$.

$$\therefore \frac{P(y=1)}{P(y=0)} \cdot \prod_{j=0}^{d} \frac{P(x_j|y=1)}{P(x_j|y=0)} \geqslant 1 \quad ----(2)$$

Let's denote $P(y=1)$ by P, $P(x=1|y=1)$ by a_j and $P(x_j=1|y=0)$ by b_j.

$$\therefore P(x_j|y=1) = a_j^{x_j}(1-a_j)^{(1-x_j)}$$

Since our features are binary and one of x_j or $1-x_j$ will be zero. Similarly $P(x_j|y=0) = b_j^{x_j}(1-b_j)^{(1-x_j)}$.

Using this notion in (2), we get following for y = 1.

$$\frac{P}{1-P} \cdot \prod_{j=0}^{d} \frac{a_j^{x_j}(1-a_j)^{(1-x_j)}}{b_j^{x_j}(1-b_j)^{(1-x_j)}} \geqslant 1 \quad -(3)$$

Taking log & simplying (3), we get

$$\log\left(\frac{P}{1-P} \prod_{j=0}^{d} \frac{1-a_j}{1-b_j}\right) + \sum_{j=0}^{d} x_j \log\left(\frac{a_j}{b_j} \cdot \frac{1-b_j}{1-a_j}\right) \geqslant 0$$

$-(5)$

For any input x, the first term in this sum is constant, because it does not have any x_j terms.

Let us denote it by $b = \log\left(\frac{p}{1-p} \prod_{j=0}^{d} \frac{1-a_j}{1-b_j}\right)$.

Further, let us denote $\log\left(\frac{a_j}{b_j} \cdot \frac{1-b_j}{1-a_j}\right)$ by w_j.

Substituting these, we get

$$b + \sum_{j=0}^{d} x_j w_j \geq 0 \quad \text{— (6)}$$

Therefore our classifier is a linear classifier.

$w = (w_1, w_2, \ldots, w_d)$ where $w_j = \log\left(\frac{a_j}{b_j} \frac{1-b_j}{1-a_j}\right)$

where $a_j = P(x_j = 1 \mid y = 1)$

& $b_j = P(x_j = 1 \mid y = 0)$.

2.

Naive Bayes and Logistic Regression converge toward their asymptotic accuracies at different rates. Naive Bayes parameter estimates converge toward their asymptotic values in order logn examples, where n is the dimension of X.
In contrast, Logistic Regression parameter estimates converge more slowly, requiring order n examples. Even though Logistic Regression outperforms Naive Bayes when many training examples are available, but Naive Bayes outperforms Logistic Regression when training data is scarce.