1 (a)

Given :

Store	Total Employees	% Women	Men	Women
A	50	50	25	25
B	75	60	30	45
C	100	70	30	70
Total	225	180	85	140

Also given, a woman employee resigned.
To find - P(Worked[Store C] | Woman)

= P(Worked[Store C]) ∩ P(Woman)/P(Woman)
= (70/225)/(140/225)
= 70/140
= 0.5

1 (b)
Given :
Note : In this question, present refers to disease is present.
P(detected | present) = 0.95
P(detected | not present) = 0.01
P(present) = 0.005
So, P(not present) = 1 – 0.005 = 0.995
To find : P(present | detected).
P (detected) = P(detected ∩ present) + P(detected ∩ not present)
= P(detected | present) X P(present) +P(detected | not present) X P(not present)
= 0.95 X 0.005 + 0.01 X 0.995
= 0.0147
P(present | detected) = P(present) X P(detected | present) / P(detected)
= 0.005 X 0.95 / 0.0147
P(present | detected) = 0.32313

1 (c-d)

Team	Won	Lost
Atlanta Braves	**87**	**72**
San Francisco Giants	**86**	**73**
Los Angeles Dodgers	**86**	**73**

Faceoff 1 – Giants vs. Dodgers
Faceoff 2 – **Atlanta Braves(AB)** vs. Padres

Total number of possibilities per faceoff for 1 game – 2 (W : L or L : W)
Total number of possibilities per faceoff for 1 game – 8 (W : L or L : W)
Total number of possibilities for both faecoffs for 3 games – 8 X 8 = 64

c) P(AB wins division) = P(AB wins division | playoff) + P(AB wins division | playoff)

AB wins division without playoff when
i) AB wins 3 with no restrictions – 1 X 8 = 8 possibilities
ii) AB wins 2 with other team winning 3 – 3 X 2 = 6 possibilities
iii) AB wins 1 with other team winning 2 or 3 games – 3 X 8 = 24 possibilities
Total possibilities – 38
So Probability that AB wins the division is **38/64 = 0.59375**

d) To find : the probability of a playoff.
Playoff will happen when :
a) AB wins 1 game and any of the other two teams win 2 games – **Total cases – 3 X 6 = 18**
b) AB wins 2 games and any of the other 2 teams win 3 games – **Total cases 3 X 2 = 6**
So, total probability of a playoff **= (18 + 6)/64 = 24/64 = 0.375**

2
Maximum Likelihood Estimation with respect to a variable is determining the value of the variable that maximizes the product of the probability across all random variables. We take the logarithm of the product and differentiate it with respect to the variable to achieve the optimum value.

2 (a)

Poisson distribution – $P(x^i = k) = \lambda^k e^{-\lambda} / (k!)$ (k = 0,1,2 ...)

For MLE wrt λ, find value of λ that maximizes $\prod P$ for all values of k.

Taking log on both sides ,

$\log(\prod P) = \log(P_1) + \log(P_2)$ .... where P_k refers to P with $x^i = k$.

$\log(\prod P) = \sum (\log(\lambda^k e^{-\lambda} / (k!)))$

Differentiating with respect to λ and equating to 0,

$\partial \log(\prod P) / \partial\lambda = \sum \partial (k \log \lambda - \lambda - \log(k!)) / \partial \lambda$ .... ∑ means summation from 1 ... n.

$0 = \sum (k/\lambda - 1)$

$0 = (\sum k)/\lambda - \sum 1$

$0 = (\sum k)/\lambda - n$

so, **$\hat{\lambda} = \sum k / n$. i.e. average over all k values.**

2 (b) Multinomial Distribution

$f(x_1, x_2, \dots, x_n; n, \theta_1, \theta_2, \dots, \theta_n)$ =(n! / x_1!x_2!...x_n!)$\prod \theta_j^{x_j}$

Taking log on both sides, we get

$\log(f) = \log(n!) - \sum\log(x_j)! + \sum\log\theta_j^{x_j}$
$\log(f) = \log(n!) - \sum\log(x_j)! + \sum\log\theta_j^{x_j}$

$= \log(n!) - \sum\log(x_j)! + \sum x_j\log\theta_j$

Because of the constraint $\sum \theta_j = 1$, we will use the lagrange multipliers.

$l(\theta_1, \theta_2 \dots \theta_n, \lambda) = l(\theta_1, \theta_2 \dots \theta_n) + \lambda(1- \sum\theta_j)$
$\log (f)= \log(n!) - \sum\log(x_j)! + \sum x_j\log(\theta_j) + \lambda(1 - \sum\theta_j)$

We differentiate with respect to θ_j and equate to 0, we get $\theta_j = (x_j)/ \lambda$.

$\sum\theta_j = 1$, so $\sum(x_j)/ \lambda = 1$; $\lambda = \sum(x_j) = n$

So $\theta_j = x_j / n$

2 (c) Gaussian Normal Distribution

N(x; u ; σ^2**) = 1 / (**$\sigma\sqrt{2}\pi$) e^p where $p = -(x_i-u)^2/ 2\sigma^2$.

For MLE, we have to take log of the product.

So,

$\prod(N) = \prod$**1 / (**$\sigma\sqrt{2}\pi$) e^p where $p = -(x_i-u)^2/2\sigma^2$ ...... N varies over the n sample data $x_1, x_2, \dots$

$= ((2\pi)^{-n/2} / \sigma^n) e^{\Sigma p}$

Taking log,

$\log(\prod N) = (-1/2)n \log(2\pi) - n\log(\sigma) - \sum p$

$\log(\prod N) = (-1/2)n \log(2\pi) - n\log(\sigma) - (\sum(x_i - u)^2)/2\sigma^2$.

Taking derivative wrt u,

$\partial(\log(\prod N)) / \partial(u) = (\sum(xi - u))/\sigma^2 = 0$

$\hat{u} = \sum x_i / n$
Similarily, differentiating $\log(\prod N)$ wrt σ.

$\partial(\log(\prod N)) / \partial(\sigma) = -(n/ \sigma) + (\sum(xi - u)^2 / \sigma^3) = 0$

Puttiing $\hat{u}$ in place of u and solving, this we get,

$\hat{\sigma} = \sqrt{\sum} (x_i - \hat{u})^2/n$

3)

Given :

$$\mathbf{x^n = \sum_{i=1}^{D} \alpha_i^n u_i = \sum_{i=1}^{D} (x^{nT} u_i) u_i}$$

$$\mathbf{\breve{x}^n = \sum_{i=1}^{M} z_i^n u_i + \sum_{i=M+1}^{D} b_i u_i}$$

$$\mathbf{J = (1/N) \sum_{n=1}^{N} ||x^n - \breve{x}^n||^2}$$

Replacing the value of $\breve{x}$ in J, we get,

$J = (1/N) \sum_{n=1}^{N} ||x^n - \sum_{i=1}^{M} z_i^n u_i + \sum_{i=M+1}^{D} b_i u_i||^2$

$=(1/N) \sum_{n=1}^{N} (x^n - \sum_{i=1}^{M} z_i^n u_i + \sum_{i=M+1}^{D} b_i u_i)(x^n - \sum_{i=1}^{M} z_i^n u_i + \sum_{i=M+1}^{D} b_i u_i)^T$

Replacing x^n with $\sum_{i=1}^{M} \alpha_i^n u_i + \sum_{i=M+1}^{D} \alpha_i^n u_{i,}$ we get,

$J = (1/N) \sum_{n=1}^{N} (\sum_{i=1}^{M} \alpha_i^n u_i + \sum_{i=M+1}^{D} \alpha_i^n u_i - \sum_{i=1}^{M} z_i^n u_i + \sum_{i=M+1}^{D} b_i u_i)^T(\sum_{i=1}^{M} \alpha_i^n u_i + \sum_{i=M+1}^{D} \alpha_i^n u_i - \sum_{i=1}^{M} z_i^n u_i + \sum_{i=M+1}^{D} b_i u_i)$

using $\alpha_i^n = x^{nT} u_i$ and combining for $\sum_{i=1}^{M}$ and $\sum_{i=M+1}^{D}$ separately.

$$\mathbf{J = (1/N) \sum_{n=1}^{N} (\sum_{i=1}^{M}(x^{nT} u_i - z_i^n)(x^{nT} u_i - z_i^n)^{T\,-} \sum_{i=M+1}^{D}(x^{nT} u_i - b_i)(x^{nT} u_i - b_i)^T)}$$

a)

Differentiating J wrt z_j^n and equating to 0, we get,

$$\partial(J)/ \partial(z_j^n)= (-2/N)\sum_{n=1}^{N} (x^{nT} u_j - z_j^n) = 0$$

$$0 = (x^{nT} u_j - z_j^n)$$

$$\mathbf{z_j^n = x^{nT} u_j}$$

b)

Differentiating wrt b_j and equating to 0, we get,

$$\partial(J)/ \partial(b_j) = (2/N) \sum_{n=1}^{N} \sum_{i=M+1}^{D} (x^{nT} u_j - b_j) = 0$$

$$= \sum_{n=1}^{N} x^{nT} u_j = b_j(N)$$

$= b_j = (\sum_{n=1}^{N} x^{nT}u_j) / N = \overline{\boldsymbol{x^{nT}}}\mathbf{u_j}$

c)

Substituing $\mathbf{z_j^n = x^{nT}u_j}$ **and** $\mathbf{b_{j\,=}} = \overline{\boldsymbol{x^{nT}}}\mathbf{u_j}$ **in** $\mathbf{x^n} - \tilde{\mathbf{x}}^n$ for optimality, we get ,

$\mathbf{x^n} - \tilde{\mathbf{x}}^n = \sum_{i=1}^{D} (x^{nT}u_i)u_i - \sum_{i=1}^{M} (x^{nT}u_i)u_i - \sum_{i=M+1}^{D} (\overline{x^{nT}}u_i)u_i$

$= \sum_{i=M+1}^{D} (x^{nT}u_i)u_i - - \sum_{i=M+1}^{D} (\overline{x^{nT}}u_i)u_i$

$= \boldsymbol{\sum_{i=M+1}^{D} ((x^n - \overline{x^n})^T u_i)u_i}$

d)

Following from above, distortion function can be written as,

$J = (1/N) \sum_{n=1}^{N} ||x^n - \tilde{\mathbf{x}}^n||^2$

$= (1/N) \sum_{n=1}^{N} \sum_{i=M+1}^{D} ||((x^n - \overline{x^n})^T u_i)u_i||^2$

$= (1/N) \sum_{n=1}^{N} \sum_{i=M+1}^{D} (u_i)^T((x^n - \overline{x^n})^T u_i)^T((x^n - \overline{x^n})^T u_i)(u_i)$

$= (1/N) \sum_{n=1}^{N} \sum_{i=M+1}^{D} (u_i)^T(x^n - \overline{x^n})(x^n - \overline{x^n})^T(ui)$

Using the hint : covariance matrix $S = (1/N) \sum_{n=1}^{N} (x^n - \overline{x^n})(x^n - \overline{x^n})^T$

$J = \sum_{i=M+1}^{D} (u_i)^T S(u_i)$

Minimizing the cost function for u_j (u_j is orthonormal and $u_j(u_j)^T = 1$). Using lagrange multiplier λ_j,

$J = (u_j)^T S(u_j) + \lambda_j(1 - (u_j)^T u_j)$

Differentiating J wrt u_j, we get,

$\partial(J)/ \partial(u_j) = (u_j)^T S - \lambda_j (u_j)^T = 0$

$(u_j)^T S = \lambda_j (u_j)^T$

Taking transpose,

$Su_j = \lambda_j\ u_j$ ..... [S is a symmetric matrix.]

Thus u_j is the eigen vector of S. S contains a total of D eigen vectors.

The value of $J = \sum_{i=M+1}^{D} \lambda_i$. The minimum value of J is obtained by selecting eigenvectors corresponding to D - M smallest eigenvalues (corresponding to coefficient b_i) and the principal component of u_j is given by the eigenvectors corresponding to M eigenvalues.

4

Given N data points x^n (n=1,2,3 ...N) and the K-means clustering distortion function :

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r^{nk} ||x_n - u^k||^2$$

where r^{nk} =1 for the point n only if the point n belongs to the k^{th} cluster.

4(a)

To prove that $u^k = (\sum_n r^{nk} x_n) / (\sum_n r^{nk})$

Differentiating the distortion function wrt u^k keeping r^{nk} fixed for k='k' (this is the maximization step in K- Means algorithm), we get

$\partial(J)/ \partial (u^k) = -2 \sum_{n=1}^{N} r^{nk} (x_n - u^k) = 0$

$0 = \sum_{n=1}^{N} r^{nk} x_n - u^k \sum_{n=1}^{N} r^{nk}$

$0 = \sum_n r^{nk} x_n - u^k \sum_n r^{nk}$

so, $u^k = (\sum_n r^{nk} x_n) / (\sum_n r^{nk})$

4(b) To prove that K-means converges in finite number of steps.

First, we take the derivative of J wrt $u^{k.}$

$P = \partial(J)/ \partial (u^k) = -2 \sum_{n=1}^{N} r^{nk} (x_n - u^k)$

Differentiating again with respect to u^k, we get

$\partial(P)/ \partial (u^k) = \partial^2(J)/ \partial (u^k)^2 = 2$

The double derivative of the function with respect to u^k is positive, proving that function converges to local minimum in finite steps.

Also, theoretically, for each of the steps, finding the assigning the points to a cluster and calculating the cluster center are motivated by minimizing the distortion function, hence the converging of the algorithm is promised.

4(c) Average linkage distance metric will most likely result in same clusters as produced by the k means. Using average linkage, we will find the average distance between all the points of the two clusters which is almost equal to saying that we find the distance between the mean of all points in a cluster (cluster center for K - means).

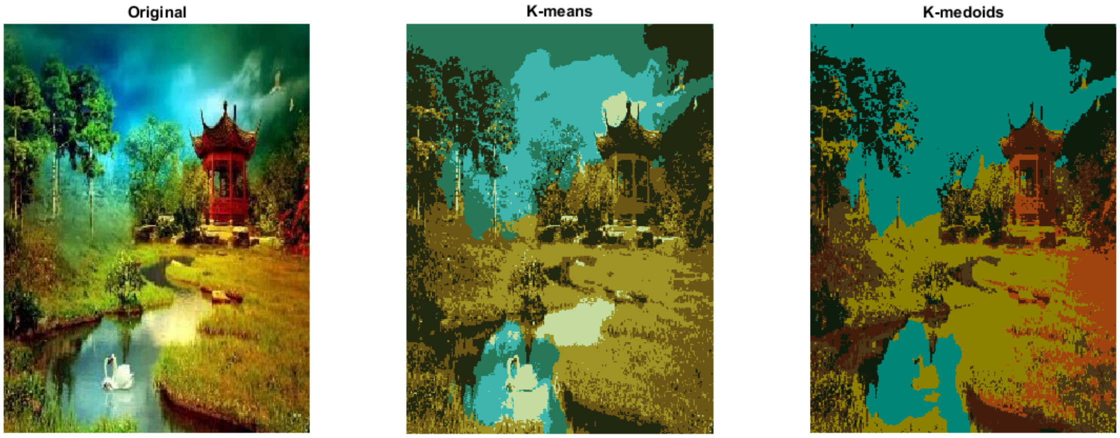
4(d) Using bottom-up clustering, single linkage will be able to separate the two moons. The data is distributed in such a way that points in the cluster in any moon are closer than the points in cluster of other moon.

5)

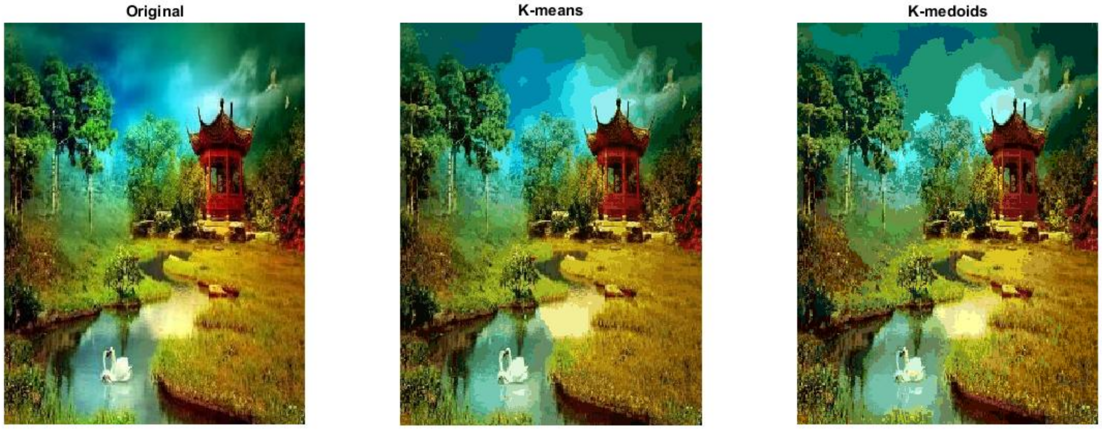
1. For the k-medoids algorithm, we start by initializing the K cluster points with random pixel points. In the expectation phase, I calculate the cluster for a pixel using euclidean distance metric. In the maximization phase, I calculate the mean of the cluster points - **"Mean Point"** and assign the cluster center as the pixel point closest(euclidean distance) to the Mean Point. The algorithm stops when there is no change in the cluster centers or when the total number of iterations exceed 500 for k-means and 800 for k-medoids. Both k-medoids and k-means were executed using euclidean distance as the distance metric. I tried the mahalanobis, jaccard and hamming distance metrics for k-medoids but the time for execution exceeded 5 minutes even for 16 clusters.

2.

For 6 clusters,



For 64 clusters,



3. For 3 clusters,

kmeansTime = 0.1289s and 11 iterations

kmedoidsTime = 16.0949s and 136 iterations

For 16 clusters,

kmeansTime = 3.4777s and 11 iterations

kmedoidsTime = 31.7200s and 284 iterations

4. I am initializing the clusters randomly by selecting random pixel points. For the same figure and same algorithm, I can see variance in the running time. K - means showed a maximum difference of approximately 3s and 25 iterations for my testcases. K-medoids showed a maximum difference of approximately 15 s and 56 iterations depending on different initial centroid. I also tried initializing clusters with another method like sorting pixels based on R, G or B values and selecting the first pixel values as cluster centers.

5. K-means performs better than the k-medoids in terms of output quality. The image rendered by k-means preserves more color than k-medoids. Also k-means runs significantly faster than k-medoids as seen in part 3.