Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Shwetank Anand
Roll Number: 211025
Date: November 17, 2023

QUESTION 1

The standard k-means loss function is given as:

$$L(X, Z, \mu) = \sum_{n=1}^{N} \sum_{k=1}^{K} X_{znk} \|x_n - \mu_k\|^2 \tag{1}$$

SGD K-means

Step 1: To make it online by taking a random example x_n at a time and then assigning x_n "greedily" to the "best" cluster. Using ALT-OPT technique, fix $\mu = \hat{\mu}$ and solve for z_n.

$$\hat{z} = \arg\min_{X_z} \|x_n - \hat{\mu}\|^2_{n\hat{k}} \tag{2}$$

To perform Step 1, we need to assign a cluster to x_n using the above equation for each of the examples $\{x_n\}_{n=1}^N$.

Step 2: To update the cluster means. Solving for μ using SGD on the objective function by fixing $z = \hat{z}$.

$$\hat{\mu} = \arg\min_{\mu} L(X, \hat{Z}, \mu) = \arg\min_{\mu} \sum_{n=1}^{N} \|x_n - \mu_{\hat{k}}\|^2_{n\hat{k}} \tag{3}$$

At any iteration t, choose an example x_n uniformly randomly and approximate g as:

$$g \approx g_n = \frac{\partial(\|x_n - \mu_{\hat{k}}\|^2)}{\partial \mu_{\hat{k}}} = -2(x_n - \mu_{\hat{k}}) \tag{4}$$

Now the mean can be updated as:

$$\mu^{(t+1)} = \mu^{(t)}_{\hat{k}} - \eta g^{(t)}_{\hat{k}} = \mu^{(t)}_{\hat{k}} + 2\eta(x^{(t)}_n - \mu^{(t)}_{\hat{k}}) \tag{5}$$

where $n : \hat{z}_n = \hat{k}$. The step size can be $\eta \propto \frac{1}{N_k}$, where N_k is the number of data points in the k-th cluster, so that the updated mean is in the ratio of the sum of features of every data point to the total number of data points in that cluster.

This intuitively makes sense as well because in the K-means algorithm, at each step, we recompute the mean of all the clusters, and here too we are recomputing the mean of the cluster that was changed by adding the new input. This is done in such a way that the new mean is now closer to the actual mean, ensuring that the loss function is minimized. The new mean takes into account the previous mean as well, as it gets closer to the new example.

3. When using the K-means algorithm, the new mean after each update looks like

$$\mu^{(n)}_k = \frac{\sum_{i=1}^{N} z^{(n)}_{ik} x_i}{N^{(n)}_k}$$

Taking inspiration from this, we suggest η to be $1/2N_k$, where N_k is the number of x_n in this cluster after the update operation. This is because this makes the mean update exactly the

same as that in the K-means algorithm. We can prove this by induction. The base case is true when we have just one input in the sample space:

$$\mu^{(1)} = \mu^{(0)} - (\mu^{(0)} - x_1) = x_1$$

Using the induction hypothesis, we get

$$\mu^{(n)} = \mu^{(n-1)} - \frac{\sum_i z_{ik'}^{(n)}(\mu^{(n-1)} - x_i)}{N'_k}\mu^{(n-1)} + \frac{\sum_i z_{ik'}^{(n)} x_i}{N'_k} = \mu^{(n-1)} - \frac{\sum_i z_{ik'}^{(n)} x_i}{N'_k} + \frac{\sum_i z_{ik'}^{(n)} x_i}{N'_k} = \mu^{(n-1)} - \frac{\sum_i z_{ik'}^{(n)} x_i}{N'_k}$$

Hence, this is good.

Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Shwetank Anand
Roll Number: 211025
Date: November 17, 2023

QUESTION 2

The objective function that needs to be maximized is given by:

$$L(\mathbf{w}) = \mathbf{w}^T \mathbf{S}_M \mathbf{w} - \mathbf{w}^T \mathbf{S}_I \mathbf{w}$$

where:

$$\mathbf{S}_M = (\mu_+ - \mu_-)(\mu_+ - \mu_-)^T$$

$$\mathbf{S}_I = \sum_{i \in \mathcal{C}_+} (\mathbf{x}_i - \mu_+)(\mathbf{x}_i - \mu_+)^T + \sum_{i \in \mathcal{C}_-} (\mathbf{x}_i - \mu_-)(\mathbf{x}_i - \mu_-)^T$$

Here, $\mathbf{w}$ is the projection vector (weight vector), μ_+ and μ_- are the mean vectors of the positive and negative classes, respectively, and $\mathcal{C}_+$ and $\mathcal{C}_-$ are the sets of indices for the positive and negative classes. The above loss function will yield the required results as the $\mathbf{S}_M$ will be maximized, which in turn maximises the distance between the mean of both the classes whereas $\mathbf{S}_I$ will be minimised so that the inputs within each class become as close to each other as possible.

Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Shwetank Anand
Roll Number: 211025
Date: November 17, 2023

QUESTION 3

Let $\mathbf{v}$ be an eigenvector of the covariance matrix $\mathbf{S} = \frac{1}{N}\mathbf{X}\mathbf{X}^T$. Then we have the following equation:

$$\frac{1}{N}\mathbf{X}\mathbf{X}^T\mathbf{v} = \lambda\mathbf{v}$$

Multiplying the equation on the left by $\mathbf{X}^T$, we get the following:

$$\left(\frac{1}{N}\mathbf{X}^T\mathbf{X}\right)(\mathbf{X}^T\mathbf{v}) = \lambda(\mathbf{X}^T\mathbf{v})$$

This implies that $\mathbf{u} = \mathbf{X}^T\mathbf{v}$ is an eigenvector of $\mathbf{X}^T\mathbf{X}$. So for every eigenvector $\mathbf{v}$ of $\frac{1}{N}\mathbf{X}\mathbf{X}^T$, we have $\mathbf{u} = \mathbf{X}^T\mathbf{v}$, which is an eigenvector of $\frac{1}{N}\mathbf{X}^T\mathbf{X}$.

This way of obtaining the eigenvector $\mathbf{u}$ of $N \times N$ matrix $\mathbf{X}^T\mathbf{X}$ from the eigenvector $\mathbf{v}$ of $\frac{1}{N}\mathbf{X}\mathbf{X}^T$ is much more efficient in this case because here we need to do an eigendecomposition of an $N \times N$ matrix instead of a $D \times D$ matrix. Since $D > N$, this method is more computationally efficient.

Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Shwetank Anand
Roll Number: 211025
Date: November 17, 2023

QUESTION 4

1. The model classifies data into distinct classes, employing unique sets of weight vectors to model the output of each class using the standard probabilistic model within each class. By utilizing a mixture of different weight vectors, it addresses the challenge of accurately modeling data that may not conform well to a single weight vector. The incorporation of class-specific weight vectors enhances the model's ability to better represent the underlying data.

Moreover, a standard linear model is limited to handling solutions that require regression on a linear curve. In contrast, this model can be a combination of K different linear curves. It essentially clusters the data into K distinct linear curves, allowing for predictions of y. This approach also contributes to the reduction of outliers in a linear curve, as clustering may separate outliers from the main trend.

2. The ALT-OPT algorithm is as follows:

(a) Initialise Θ as $\Theta(0)$ and $t = 0$

(b) For each $n = 1, 2, 3, \ldots, N$, we assign $z_n^{(t)}$ to be the class with the highest probability, i.e.,

$$z_n^{(t)} = \arg\max_k p(y_n, z_n = k|x_n, \Theta^{(t)}) = \arg\max_k p(y_n|z_n = k, x_n, \Theta^{(t)})p(z_n = k|\Theta^{(t)}) =$$

$\arg\max_k \pi_k^{(t)} N(y_n|\mu_k^{(t)}, \Sigma_k^{(t)})$

(c) Given Z_t, we will now estimate $\Theta^{(t+1)}$ by maximizing the log-likelihood function.

$$\Theta^{(t+1)} = \arg\max_\Theta \log(p(Y, Z|X, \Theta))$$

Since Z is known, we can write the above as

$$\Theta^{(t+1)} = \arg\max_\Theta \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}^{(t)} \log(p(y_n, z_n|x_n, \Theta))$$

$$= \arg\max_\Theta \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}^{(t)} \left(\log(p(y_n|z_n, x_n, \Theta)) + \log(p(z_n|x_n, \Theta))\right)$$

$$= \arg\max_\Theta \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}^{(t)} \left(\log(\pi_k) + \log N(w_k^T x_n, \beta^{-1})\right)$$

Thus, the update equation for $\pi_k^{(t+1)}$ will be the same as in the generative model, i.e.,

$$\pi_k^{(t+1)} = \frac{N_k}{N}$$

where N_k is the number of points assigned to class k in the t-th iteration.

The update equation for $w_k^{(t+1)}$ will be

$$w_k^{(t+1)} = \arg\max_{w_k} \sum_{n:z_{nk}^{(t)}=1} \log N(w_k^T x_n, \beta^{-1})$$

$$= \arg\max_{w_k} \sum_{n:z_{nk}^{(t)}=1} (y_n - w_k^T x_n)^2$$

Thus the update equation $\mathbf{w}_k^{(t+1)}$ is same as discriminative model but only for a subset of input. Hence

$$\mathbf{w}_k^{(t+1)} = \left(\sum_{n:z_{nk}=1} \mathbf{x}_n \mathbf{x}_n^T\right) \left(\sum_{n:z_{nk}=1} y_n \mathbf{x}_n\right)$$

If $\pi_k = \frac{1}{K} \forall k$ then $z_{nk} = 1$ is the class that has the highest probability of having $\mathbf{x}_n$ i.e. the class for which $\mathcal{N}\left(\mathbf{w}_k^T \mathbf{x}_n, \beta^{-1}\right)$ is maximum.

For each input we take that class that has the maximum probability of having that $\mathbf{x}_n$ along with taking into account the no. of inputs in that class.

In the second step we know the class for each input and we find the best probablistic model that can model the data.

Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Shwetank Anand
Roll Number: 211025
Date: November 17, 2023

QUESTION
5

1 Programming Problem 1

1.1

Observation: As alpha(regularization hyperparameter) increases, root mean square error increases/accuracy of the model decreases.

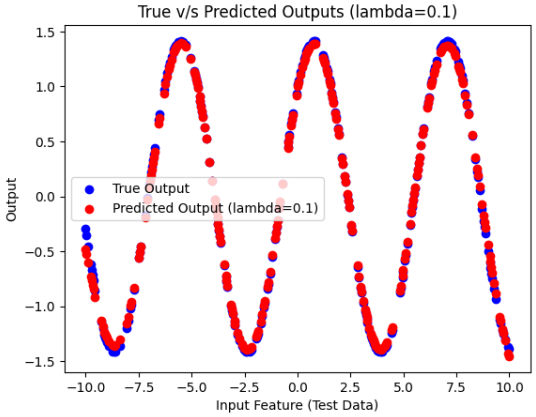


Figure 1: RMSE (lambda=0.1): 0.0325776702935743

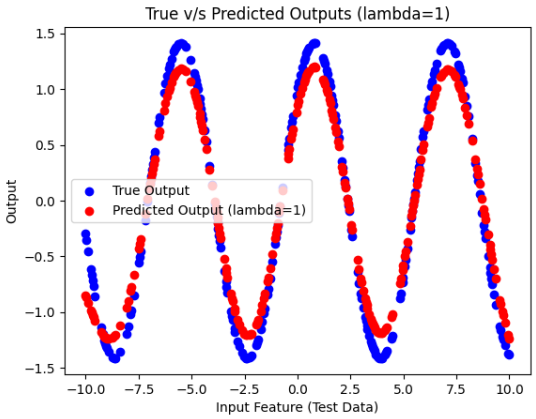


Figure 2: RMSE (lambda=1): 0.17030390344202534

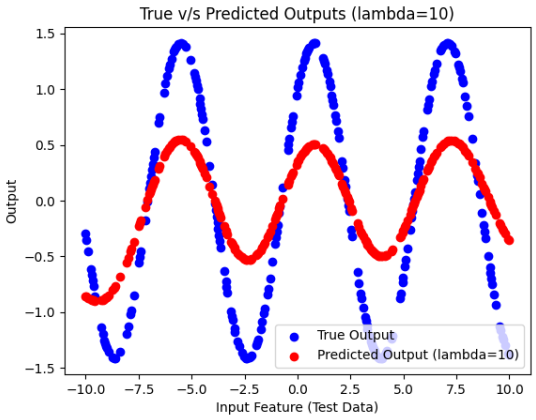


Figure 3: RMSE (lambda=10): 0.6092671596540066

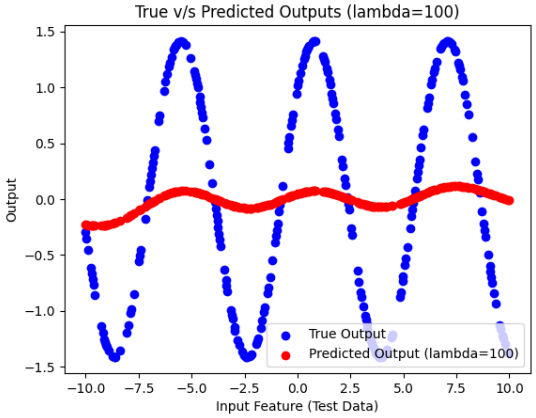


Figure 4: RMSE (lambda=100): 0.9110858052767243

1.2

Observation: As number of landmarks increase, RMSE decreases (or) accuracy increases. Note that results in the program might give slightly different results due to just choosing landmarks randomly without initializing the random seed, in our case (40). The value of L $\geq$ 50 seems good enough.

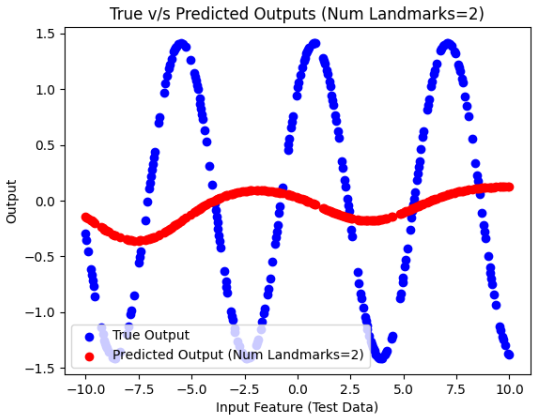


Figure 5: RMSE (Num Landmarks=2): 0.9754935831720197

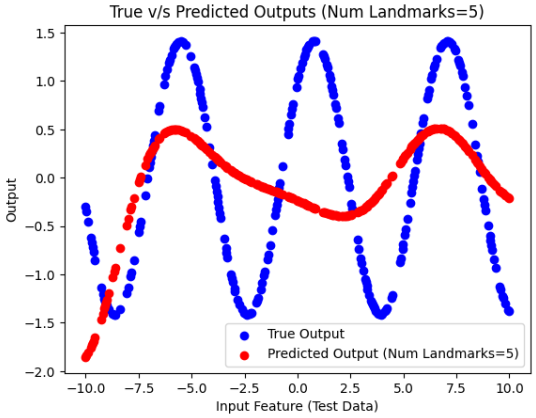


Figure 6: RMSE (Num Landmarks=5): 0.8834193658985148

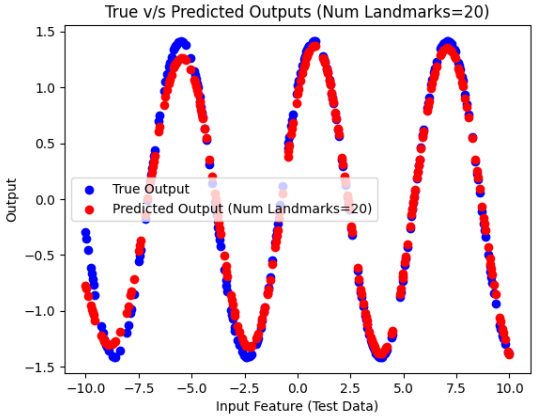


Figure 7: RMSE (Num Landmarks=20): 0.09498951158612662

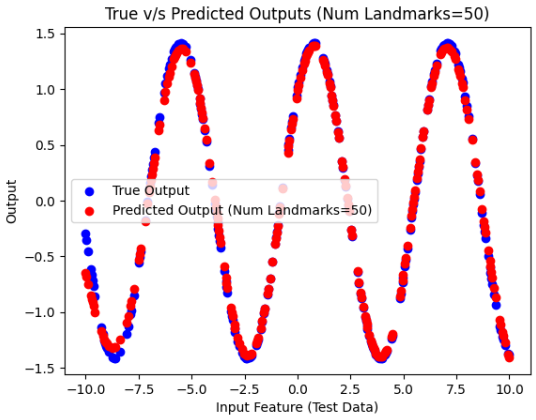


Figure 8: RMSE (Num Landmarks=50): 0.05663421142615575

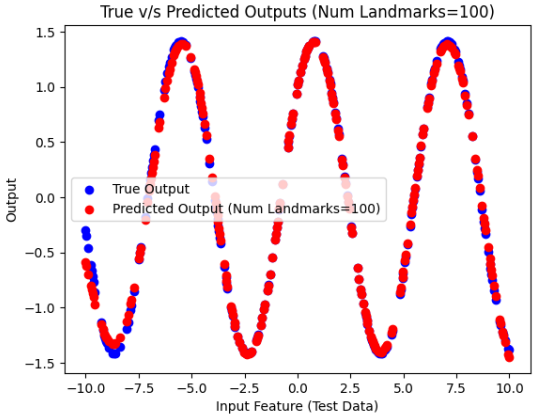
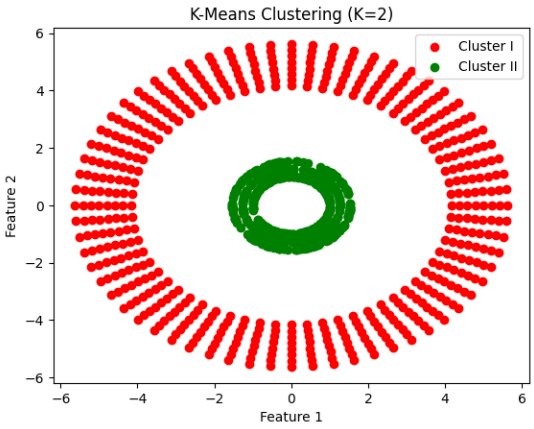


Figure 9: RMSE (Num Landmarks=100): 0.045710933316532636

2 Programming Problem 2

2.1

Feature Representation: Mapping each point (x,y) in 2D plane(training input) to its distance from origin (1D mapping). This would intake linear classification.



2.2

Justification: Here, we are mapping each point (x,y) in inputs to its distance from the landmark(which is one of the points from training data itself). Hence, as seen in part 2.1, we get a correct clustering if the landmark is closer to origin and not-so-correct clustering if the landmark is not closer to origin. In essence, the landmark creates a circular boundary about itself such that all points lying inside and points lying outside are separated into two clusters.

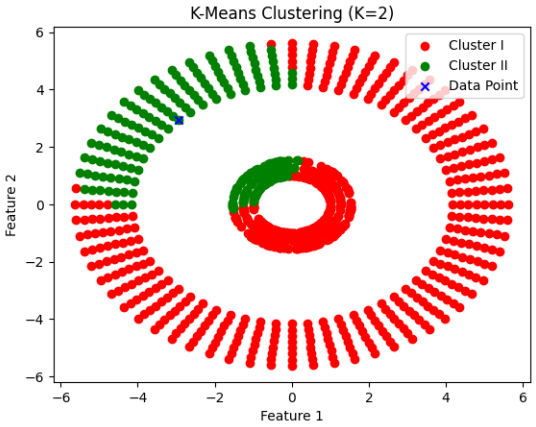


Figure 10: Iteration-1

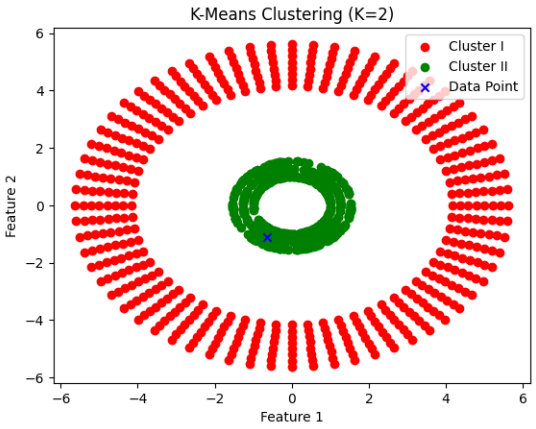


Figure 11: Iteration-2

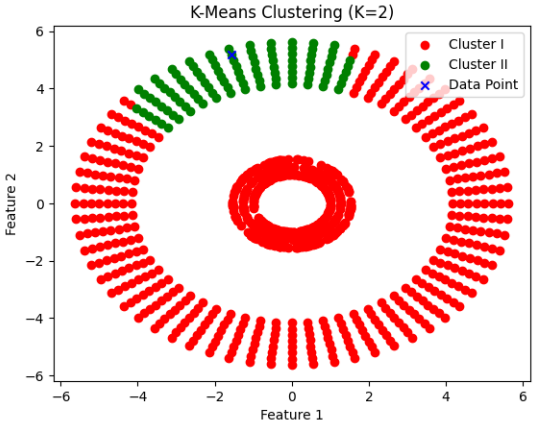


Figure 12: Iteration-3

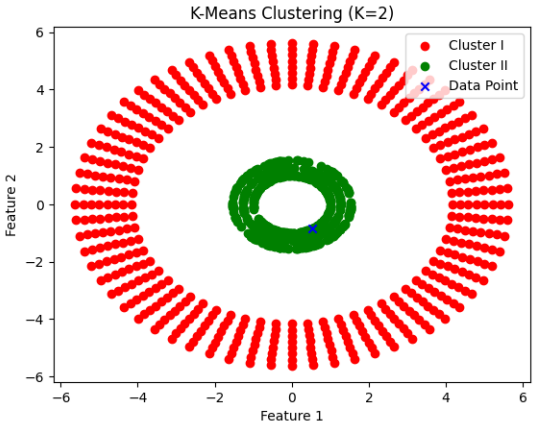


Figure 13: Iteration-4

Figure 14: Iteration-5

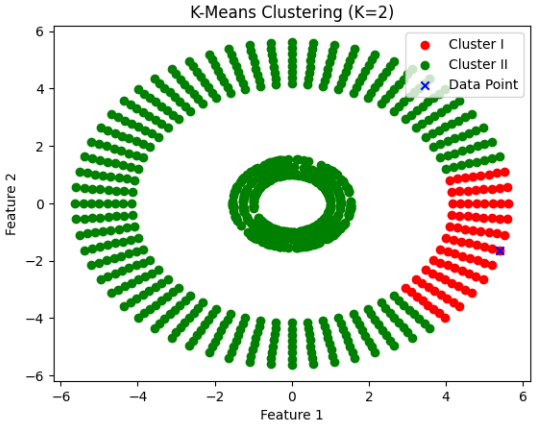


Figure 15: Iteration-6

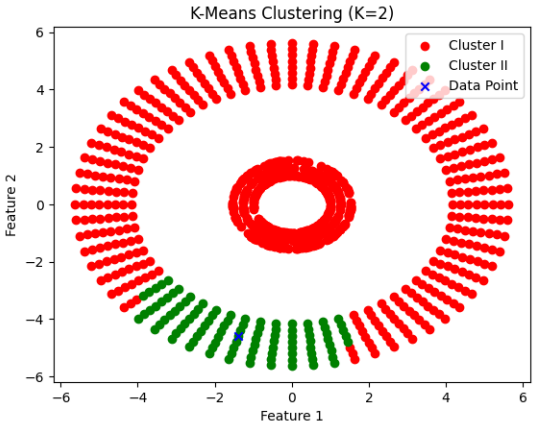


Figure 16: Iteration-7

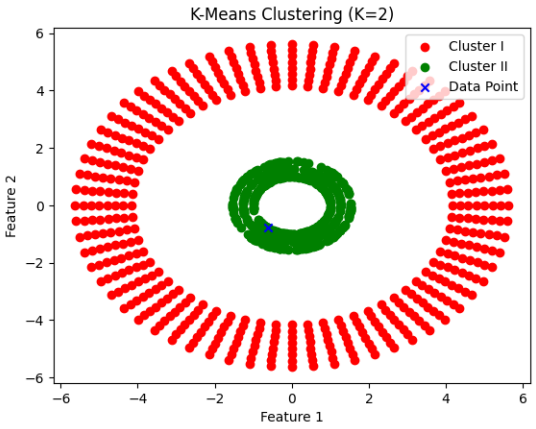


Figure 17: Iteration-8

Figure 18: Iteration-9

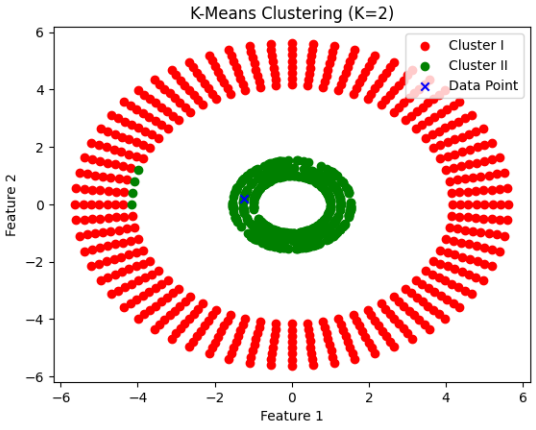


Figure 19: Iteration-10

3 Programming Problem 3

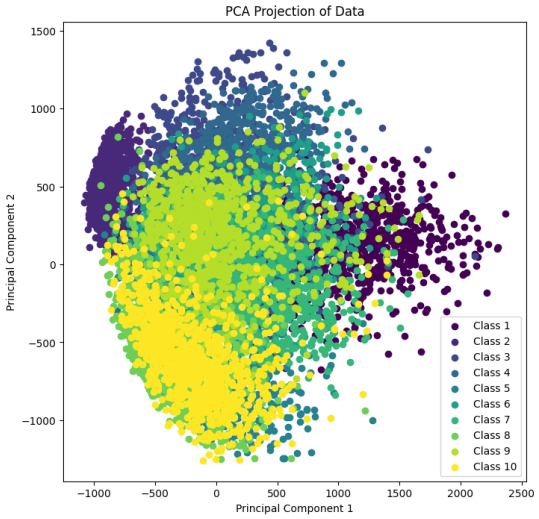


Figure 20: Using PCA

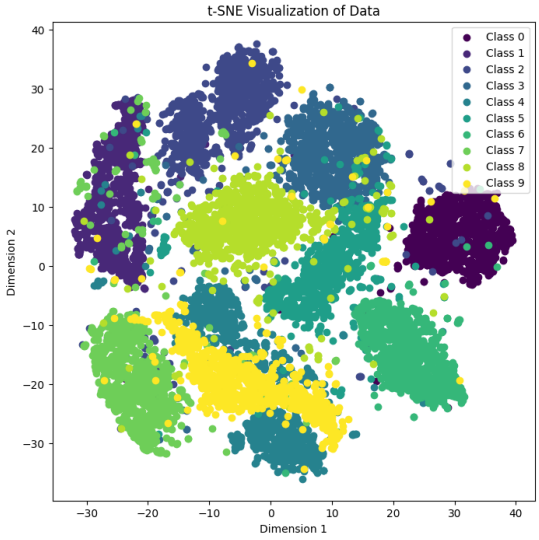


Figure 21: Using tSNE

Observation: PCA(being a linear projection method, plotting in the direction of maximum variance) doesn't capture the intrinsic non-linearities and hence the relative position of points seem to be destroyed. The story is different for tSNE where it is able to clusterize the data set into all the classes due to preserving the non-linear intrinsic geometry of the training data. Clearly tSNE works better than PCA. In the plots as well the 2-D plot with PCA has many overlapping classes whereas in tSNE classes are somewhat seperated and do not overlap as much.