FMAN45 Machine Learning

ML Assignment-2

Praveenkumar Hiremath

August 17, 2023

Task T1: Solving a nonlinear kernel SVM with hard constraints - Compute kernel matrix

Binary classification

The primal form of the SVM given by

$$\underset{\mathbf{w},b}{minimize}\,\frac{1}{2}\|\mathbf{w}\|^2, \qquad \text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \forall i \tag{1}$$

works for linearly separable dataset. However, the given dataset (below) is linearly inseparable.

i	1	2	3	4
x_i	-2	-1	1	2
y_i	+1	-1	-1	+1

(2)

Therefore, this classification problem can be addressed using the Kernelization trick. The nonlinear feature map is given by

$$\phi(x) = \begin{pmatrix} x \\ x^2 \end{pmatrix} = \begin{pmatrix} -2 & -1 & 1 & 2 \\ 4 & 1 & 1 & 4 \end{pmatrix} \tag{3}$$

which has a kernel given by $k(x,y) = \phi(x)^T\phi(y)$. The kernel matrix is now

$$K = [k(x_i,x_j)]_{1\leq i,j\leq 4} = \begin{pmatrix} k(x_1,x_1) & k(x_1,x_2) & k(x_1,x_3) & k(x_1,x_4) \\ k(x_2,x_1) & k(x_2,x_2) & k(x_2,x_3) & k(x_2,x_4) \\ k(x_3,x_1) & k(x_3,x_2) & k(x_3,x_3) & k(x_3,x_4) \\ k(x_4,x_1) & k(x_4,x_2) & k(x_4,x_3) & k(x_4,x_4) \end{pmatrix} \tag{4}$$

$$\implies \phi(x)^T\phi(y) = \begin{pmatrix} -2 & 4 \\ -1 & 1 \\ 1 & 1 \\ 2 & 4 \end{pmatrix} \cdot \begin{pmatrix} -2 & -1 & 1 & 2 \\ 4 & 1 & 1 & 4 \end{pmatrix}$$

$$= \begin{pmatrix} 20 & 6 & 2 & 12 \\ 6 & 2 & 0 & 2 \\ 2 & 0 & 2 & 6 \\ 12 & 2 & 6 & 20 \end{pmatrix} \tag{5}$$

Task T2: Solving a nonlinear kernel SVM with hard constraints - Solve Lagrangian dual problem

Using the method of Lagrange multipliers, we arrive at the Lagrangian dual problem (shown below for the given dataset) for the hard margin SVM.

$$\underset{\alpha_1,..\alpha_4}{\text{maximize}} \sum_{i=1}^{4} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{4} \alpha_i \alpha_j y_i y_j k(x_i,x_j) \tag{6}$$

Using $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = \alpha$ and the kernel matrix in equation (5), equation (6) is re-written as,

$$\begin{aligned}
&\underset{\alpha}{\text{maximize}} \left(4\alpha - \frac{1}{2} \sum_{i,j=1}^{4} \alpha^2 y_i y_j k(x_i,x_j) \right) \\
&= \underset{\alpha}{\text{maximize}} (4\alpha - \frac{\alpha^2}{2}(1 \cdot 2 + (-1 \cdot 6) + (-1 \cdot 2) + 1 \cdot 12 + (-1 \cdot 6) + 1 \cdot 2 + 1 \cdot 0 + (-1 \cdot 2)... \\
&\quad +(-1 \cdot 2) + 1 \cdot 0 + 1 \cdot 2 + (-1 \cdot 6) + 1 \cdot 12 + (-1 \cdot 2) + (-1 \cdot 6) + 1 \cdot 20)) \\
&= \underset{\alpha}{\text{maximize}} \left(4\alpha - \frac{\alpha^2}{2} 2(20 + 12 - 2 - 12) \right) \\
&\implies \underset{\alpha}{\text{maximize}} \left(4\alpha - 18\alpha^2 \right)
\end{aligned} \tag{7}$$

To get the maximum of the function in equation (7), its differentiation w.r.t. α is equated to zero. Thus,

$$\begin{aligned}
\frac{d(4\alpha - 18\alpha^2)}{d\alpha} &= 4 - 36\alpha = 0 \\
\implies \alpha &= \frac{1}{9}
\end{aligned} \tag{8}$$

Because $\frac{d^2(4\alpha - 18\alpha^2)}{d\alpha^2} = -36 < 0$, the function in equation (7) has a maximum at $\alpha = 1/9$.

Task T3: Solving a nonlinear kernel SVM with hard constraints - obtain the classifier function $g(x)$

For any support vector x_s,

$$y_s \left(\sum_{j=1}^{4} \alpha_j y_j k(x_j,x_s) + b \right) = 1. \tag{9}$$

Here, the terms inside parentheses form the classifier function i.e.

$$g(x) = \sum_{j=1}^{4} \alpha_j y_j k(x_j,x_s) + b. \tag{10}$$

Substituting $\alpha = 1/9$ from exercise (2) and using $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = \alpha$ in equation (10) gives,

$$g(x) = \alpha \sum_{j=1}^{4} y_j k(x_j, x_s) + b$$

$$= \left[\alpha \sum_{j=1}^{4} y_j ((x_j x) + (x_j x)^2) + b \right]$$

Using the data of x_j and y_j from equation (2),

$$g(x) = \alpha 6x^2 + b = \frac{1}{9} 6x^2 + b = \frac{2x^2}{3} + b \tag{11}$$

Now using a data point ($x = -2$, $y = 1$) from (2) (i.e. dataset shown in exercise 1) and inserting the g(x) value (i.e. equation (11)) in equation (9), gives,

$$y_s \left(\frac{2x^2}{3} + b \right) = 1 \left(\frac{2(-2)^2}{3} + b \right) = 1$$

$$\implies b = 1 - \frac{8}{3} = \frac{-5}{3}$$

Thus the classifier $g(x)$ is

$$g(x) = \frac{2x^2}{3} - \frac{5}{3}. \tag{12}$$

Task T4: Solving a nonlinear kernel SVM with hard constraints - obtain the classifier function $g(x)$ for a new dataset.

Now we have a new dataset as shown below.

i	1	2	3	4	5	6	7
x_i	-3	-2	-1	0	1	2	4
y_i	+1	+1	-1	-1	-1	+1	+1

(13)

It can be noticed that the pairs (x_1,y_1), (x_2,y_2), (x_3,y_3) and (x_4,y_4) from the previous dataset in (2) are now $(x_2,y_2)_{new}$, $(x_3,y_3)_{new}$, $(x_5,y_5)_{new}$ and $(x_6,y_6)_{new}$ pairs, respectively, in the new dataset. This means that the previous dataset is a subset of the new dataset. Therefore the classifier for the new dataset is the same as that of the previous dataset i.e.

$$g(x)_{new} = \frac{2x^2}{3} - \frac{5}{3}. \tag{14}$$

Task T5: Lagrangian dual of the soft margin SVM

With $\boldsymbol{\xi}$ as the vector of allowed error ξ_i for each datapoint $\mathbf{x}_i$ in the dataset, the primal form of the linear soft margin classifier is given by

$$\underset{\mathbf{w},b,\boldsymbol{\xi}}{\text{minimize}} = \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i, \qquad \text{subject to } y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0, \forall i \tag{15}$$

Here, for the i^{th} datapoint that gets classified correctly, $\xi_i = 0$. Else, $\xi_i > 1$. By replacing $\|\mathbf{w}\|^2$ as $\mathbf{w}^T\mathbf{w}$, the Langrangian can be written as

$$\mathcal{L}(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\lambda}) = \frac{\mathbf{w}^T\mathbf{w}}{2} + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{n}\lambda_i\xi_i \tag{16}$$
$$\text{and the Lagrange multipliers } \alpha_i,\lambda_i \geq 0$$

Now,

$$\underset{\alpha_1,..\alpha_n}{\text{maximize}}\ \underset{\mathbf{w},b,\boldsymbol{\xi}}{\text{minimize}}\ \mathcal{L}(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\lambda}) \tag{17}$$

To get equation (17), $\mathcal{L}(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\lambda})$ should be differentiated w.r.t $\mathbf{w}$,b and $\boldsymbol{\xi}$.

$$\text{Firstly, } \frac{\partial\mathcal{L}}{\partial\mathbf{w}} = 0, \qquad \text{using equation (16)}$$
$$\implies \mathbf{w} - \sum_{i=1}^{n}\alpha_i y_i \mathbf{x_i} = 0$$
$$\implies \mathbf{w} = \sum_{i=1}^{n}\alpha_i y_i \mathbf{x_i} \tag{18}$$

$$\text{Then, } \frac{\partial\mathcal{L}}{\partial b} = 0, \qquad \text{using equation (16)}$$
$$\implies \frac{\partial(-\sum_{i=1}^{n}\alpha_i y_i b)}{\partial b} = -\sum_{i=1}^{n}\alpha_i y_i = 0$$
$$\implies \sum_{i=1}^{n}\alpha_i y_i = 0 \tag{19}$$

$$\text{Finally, } \frac{\partial\mathcal{L}}{\partial\boldsymbol{\xi}} = \frac{\partial(C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\xi_i - \sum_{i=1}^{n}\lambda_i\xi_i)}{\partial\boldsymbol{\xi}} = 0, \qquad \text{using equation (16)}$$
$$\implies C - \alpha_i - \lambda_i = 0 \text{ or } \lambda_i = C - \alpha_i \tag{20}$$

Now the Lagrangian dual can be written as

$$\begin{aligned}\underset{\alpha_1,..\alpha_n}{\text{maximize}}\ \mathcal{L}(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\lambda}) &= \frac{\mathbf{w}^T\mathbf{w}}{2} + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{n}\lambda_i\xi_i \\ &= \frac{1}{2}\left(\sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i\right)^T\left(\sum_{j=1}^{n}\alpha_j y_j \mathbf{x}_j\right) + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\xi_i - \sum_{i=1}^{n}\lambda_i\xi_i + \sum_{i=1}^{n}\alpha_i... \\ &\quad - \sum_{i=1}^{n}\alpha_i y_i\left(\sum_{j=1}^{n}\alpha_j y_j \mathbf{x}_j\right)^T\mathbf{x}_i - \sum_{i=1}^{n}\alpha_i y_i b\end{aligned} \tag{21}$$

Substituting the equations (18), (19), and (20) in equation (21) gives,

$$\underset{\alpha_1,..\alpha_n}{\text{maximize}}\ \mathcal{L}(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\lambda}) = \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j + \sum_{i=1}^{n}\alpha_i - \sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$$

$$\begin{gathered}\underset{\alpha_1,..\alpha_n}{\text{maximize}}\, \mathcal{L}(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\lambda}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\ \text{under the constraints, } \alpha_i,\lambda_i \geq 0 \\ \text{and } \lambda = C - \alpha_i \implies 0 \leq \alpha_i \leq C \text{ (also called box constraints).}\end{gathered} \tag{22}$$

Task T6: Lagrangian dual of the soft margin SVM - show that support vectors with $y_s(\mathbf{w}^T\mathbf{x}_i + b) < 1$ have $\alpha_i = C$.

According to the complementary slackness of the KKT conditions,

$$\begin{gathered}\alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right) = 0 \\ \lambda_i\xi_i = 0\end{gathered} \tag{23}$$

But from equation (20) we have $\lambda_i = C - \alpha_i$. Therefore, $(C - \alpha_i)\xi_i = 0$. But $\xi > 0$ which suggests $\alpha_i = C$. Additionally,

$$\begin{gathered}\alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right) = 0 \\ \implies \xi_i = 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)\end{gathered}$$

For $\xi > 0$, we must have

$$\begin{gathered}\left(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)\right) > 0 \\ \implies y_i(\mathbf{w}^T\mathbf{x}_i + b) < 1.\end{gathered} \tag{24}$$

Thus, the support vectors with $y_s(\mathbf{w}^T\mathbf{x}_i + b) < 1$ have $\alpha_i = C$.

Task E1: Dimensionality reduction on MNIST using PCA. (File "Exercise_E1.m")

This exercise contains MNIST image dataset of handwritten digits. We have images $\mathbf{X}_i \in [0,1]^{28X28}$ and targets (categories) $t_i \in 0,1,..,9$. The given "A2_data.mat" file contains 12665 training data-target pairs {train_data, train_labels} and 2115 test data-target pairs {test_data, test_labels}. In these data sets, the images are stacked column-wise i.e. $\mathbf{x}_i = vec(\mathbf{X}_i) \in \mathbb{R}^{28^2}$. This means that the images are of dimension $D = 28^2 = 784$. To transform this high-dimensional data into lower-dimensional data without losing the key features that describe it, we use the method of Principal Component Analysis (PCA). Using singular value decomposition (SVD) of the matrix $\mathbf{X} \in \mathbb{R}^{DXN}$ of N data points of D dimensions, with the mean of every row of $\mathbf{X}$ equal to zero, gives

$$\begin{gathered}\mathbf{X} = \mathbf{USV^T} \text{ where,} \\ \mathbf{U} \in \mathbb{R}^{DXP} \rightarrow \text{left singular vectors with } P = min(D,N) \\ \mathbf{S} = diag(\mathbf{s}),\mathbf{s} \in \mathbb{R}^P \rightarrow \text{ diagonal matrix of } P \text{ many singular values and} \\ \mathbf{V} \in \mathbb{R}^{NXP} \rightarrow \text{ right singular vectors}\end{gathered} \tag{25}$$

The dimensionality reduction of $\mathbf{X}$ is achieved by projecting it onto the d left singular vectors. Before performing SVD, zero mean data is generated as

$$\mathbf{X}_{zero\_mean} = [\mathbf{x}_{1,:},\mathbf{x}_{2,:},...,\mathbf{x}_{D,:}] - [mean(\mathbf{x}_{1,:}),mean(\mathbf{x}_{2,:}),...,mean(\mathbf{x}_{D,:})]$$

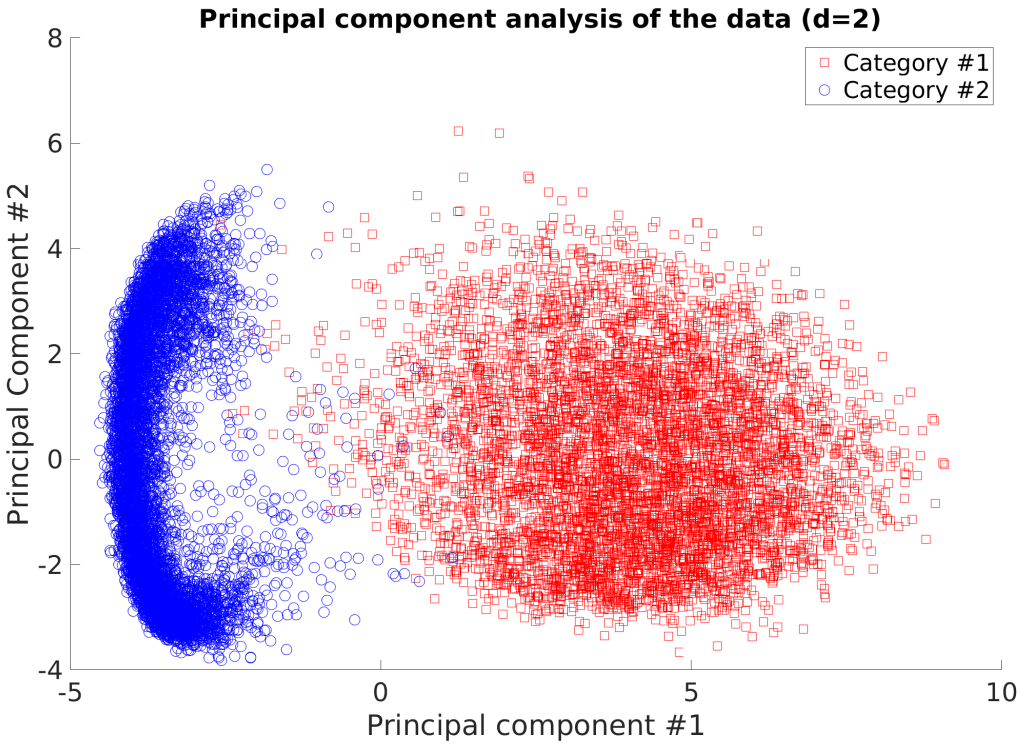
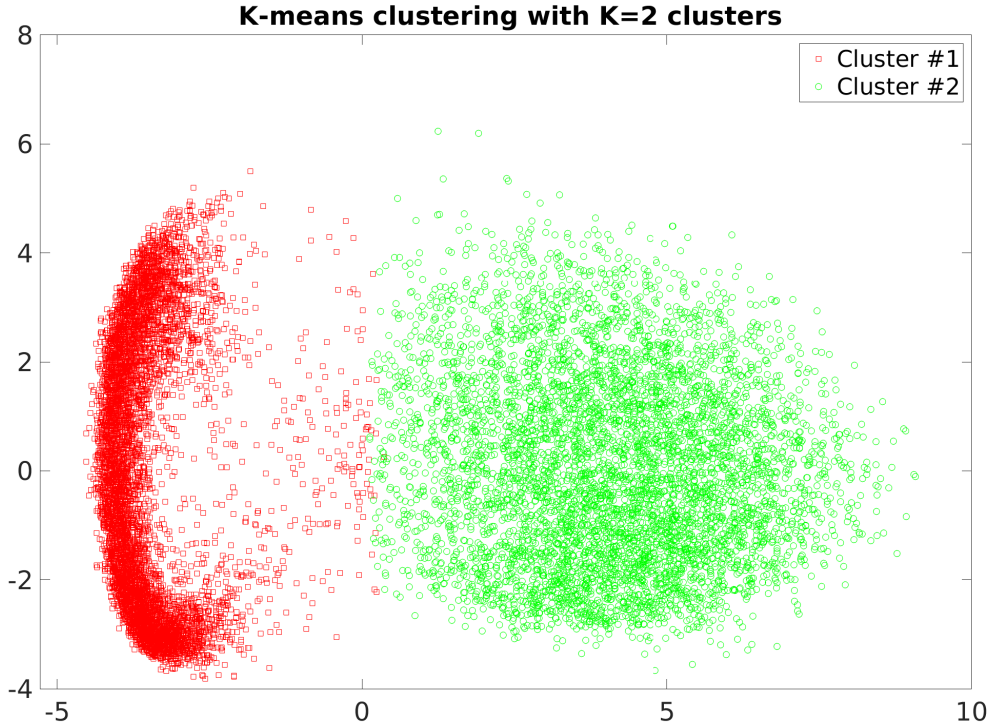


Figure 1: Principal component analysis of the training data. The training data dimension is reduced from (784X12665) to (2X12665). $d = 2$ indicates the first two principal components.

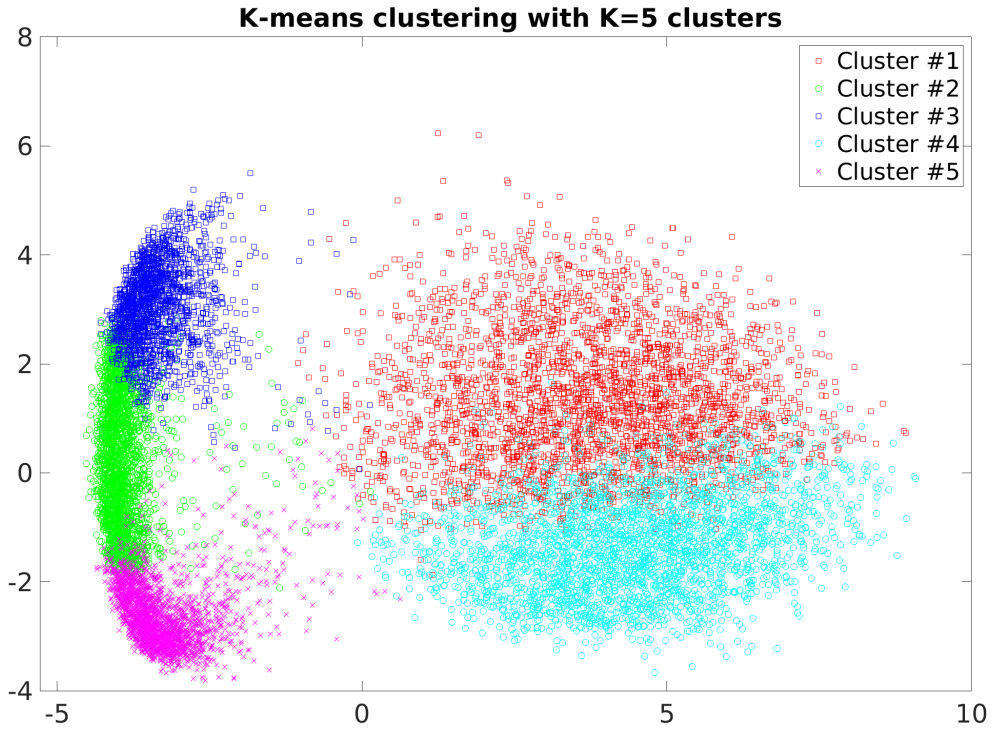
where $\mathbf{x}_{1,:}$, $\mathbf{x}_{2,:}$and $\mathbf{x}_{D,:}$ represent all the elements in rows 1, 2 and D, respectively. Then SVD of $\mathbf{X}_{zero\_mean}$ is obtained using the matlab function "svd($\mathbf{X}_{zero\_mean}$)" which yields $\mathbf{U}$, $\mathbf{S}$ and $\mathbf{V}$. Following this, $\mathbf{X}_{zero\_mean}$ is projected onto first 2 ($d = 2$) principal components (vectors) of $\mathbf{U}$, to reduce the dimension of given data $\mathbf{X}$ from $\mathbb{R}^{784X12665}$ to $\mathbb{R}^{2X12665}$. The dimensionality reduced data is presented in Figure 1. **Linear PCA algorithm is implemented in the file "linear_pca.m"**.

Task E2: Clustering of unsupervised data using K-means.

In this exercise, the MNIST images are clustered using K-means clustering which is an unsupervised ML technique. The K-means clustering algorithm provided in the assignment description is implemented in the file "K_means_clustering.m". Running the K-means clustering with K = 2 and K = 5, and then applying PCA resulted in the clusters as shown in Figure 2. However, when the algorithm was run for K = 5 multiple times, it yielded different clustering as shown in Figures 2(b) and 3. This different clustering is because of the random initialization of the centroids of the clusters. The given training data has only two labels (even though we use the unsupervised technique K-means algorithm, performing "**unique(train_labels_01)**" in Matlab gives two labels). Using K = 5 despite only two labels is justifiable because of the variations in the images belonging to a cluster. E.g. a written digit '1' could be perfectly straight or slanted. Similarly, a '0' could be a perfect circle or oval shaped or slanted, etc. In fact, this is what we see in the next task "Task E3".

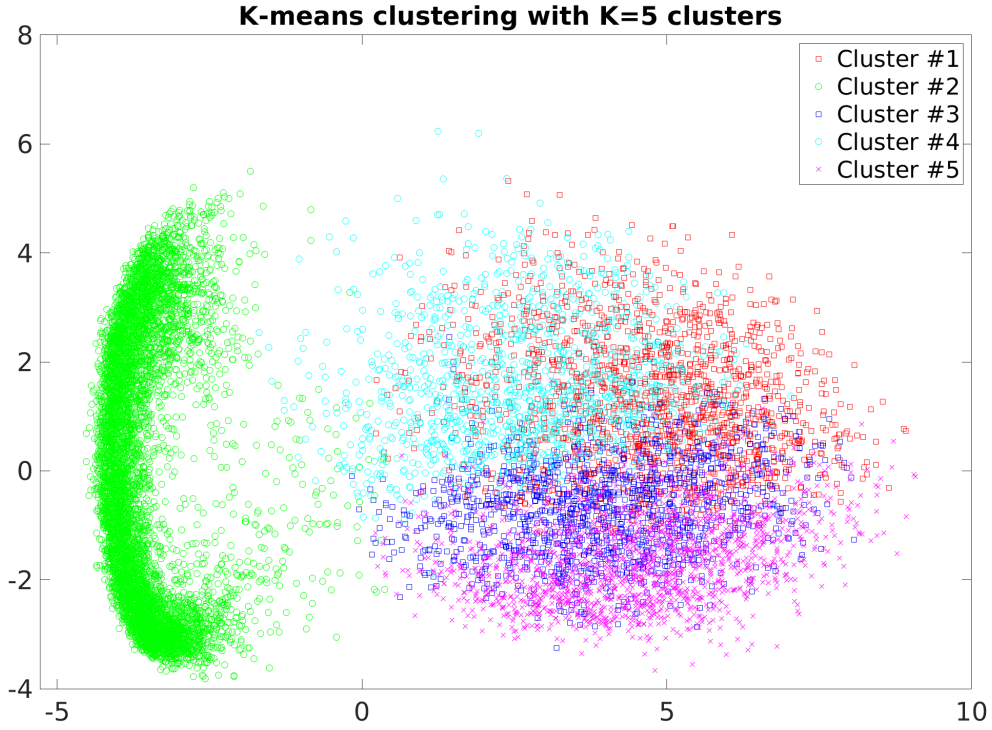


(a)

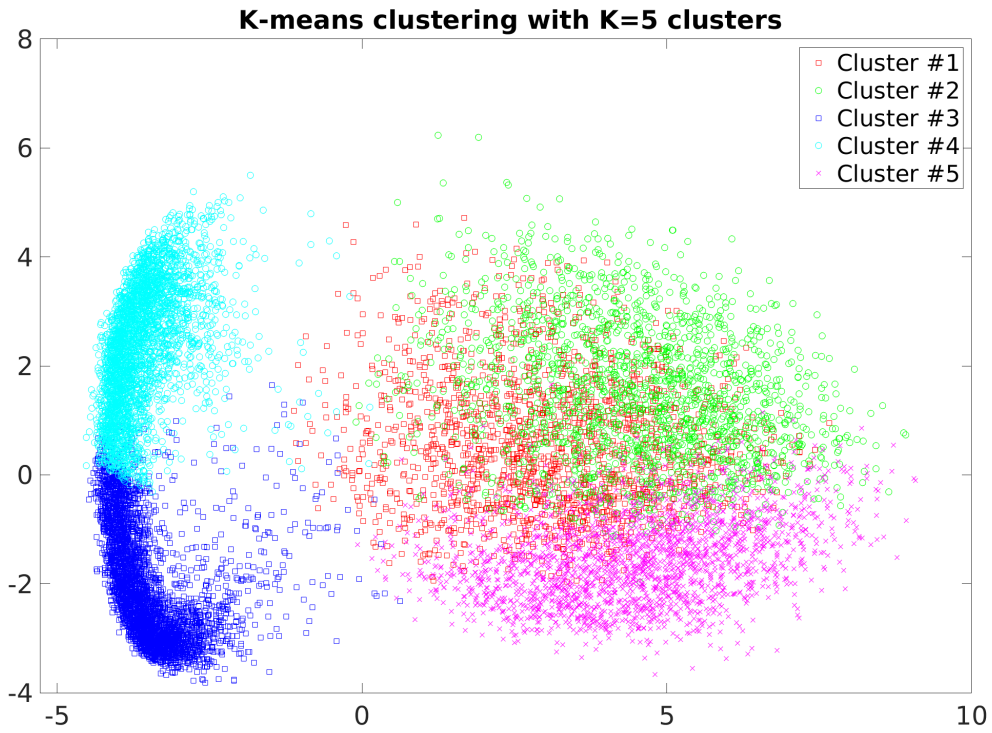


(b)

Figure 2: Task E2: Visualizing clusters with PCA for K = 2 and K = 5 (Script run iteration = 1).



(a)



(b)

Figure 3: Task E2: Visualizing clusters with PCA for K = 5 (Script run (a) iteration = 2 and (b) iteration = 3).

Further, it is noticed that the clusters overlap in the case of K = 5. This can be attributed to the PCA applied after the K-means algorithm. The K-means clustering algorithm used original (not dimensionality reduced) training data and identified the clusters according to K = 5. However, applying PCA such that the higher dimensional data is reduced to only the first two principal components ($d = 2$) causes information loss. Consequently, we see the clusters overlapping. In higher dimensions, there will not be any such overlapping of clusters as there will be very less information loss.

Task E3: Clustering of unsupervised data using K-means. (File "Exercise_E3.m")

The centroids for K = 2 and K = 5 are displayed as images in Figure 4. The centroids' images for K = 5 correspond to the clusters seen in Figure 3(b).

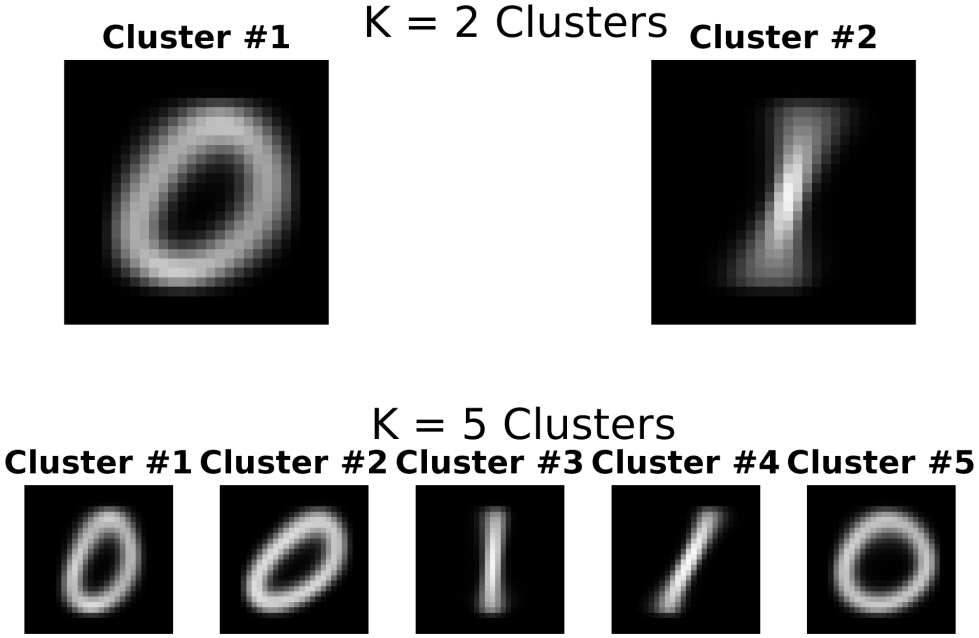


Figure 4: Task E3: Centroids corresponding to K = 2 and K = 5 as images.

Task E4: Clustering of unsupervised data using K-means. (File "Exercise_E4.m")

The function assigning the label of the closest centroid to an example is implemented in the file "K_means_classifier.m". Then for K = 2, the classification results on training and test data sets are presented in the Table 1. A missclassification rate of 0.9475% is seen for the training dataset while the test dataset showed missclassification rate of 0.4728%, for K = 2.

Table 1: K-means classification results for K = 2

Training data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	5809	6	0	6
	2	114	6736	1	114
$N_{\text{train}} = 12665$				Sum misclassified:	120
				Misclassification rate (%):	0.9475
Testing data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	970	0	0	0
	2	10	1135	1	10
$N_{\text{test}} = 2115$				Sum misclassified:	10
				Misclassification rate (%):	0.4728

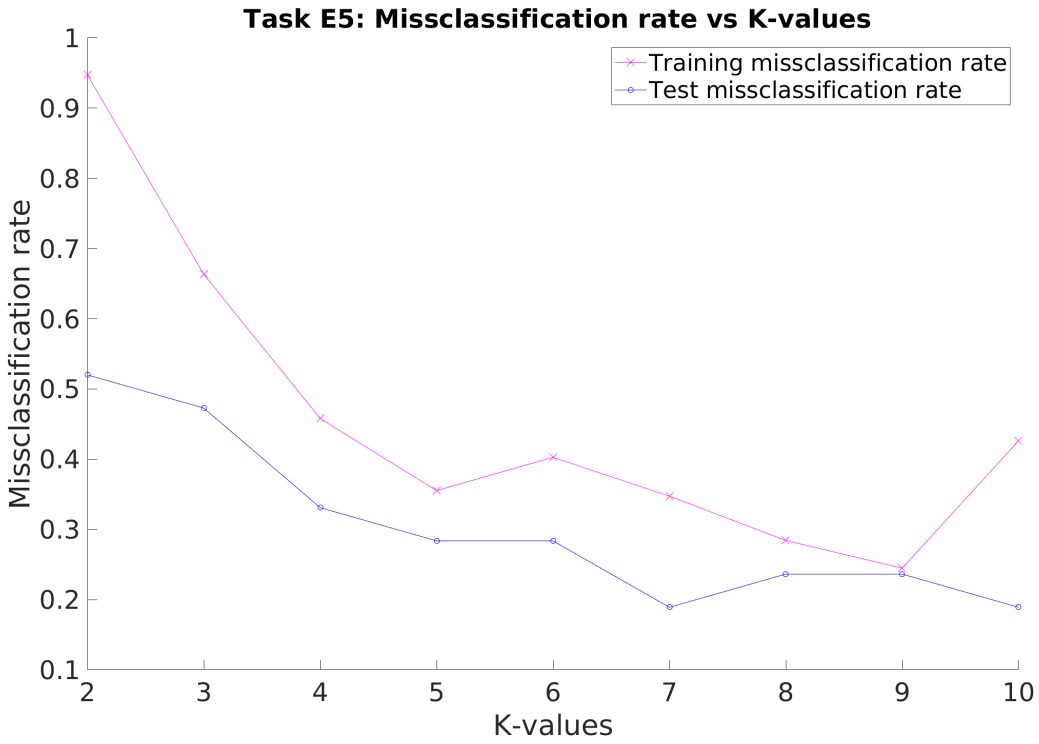


Figure 5: Task E5: Missclassification rate vs K-values.

Task E5: Clustering of unsupervised data using K-means. (Files "Exercise_E5.m" and "Exercise_E5_K7_clusters.m")

In task E5, several values are tried for K, and missclassification rate for training and test datasets is calculated. A plot of missclassification rate and K-value is presented in Figure 5. It can be seen that among the selected values, K = 7 gives lowest missclassification rate on the test dataset (file Exercise_E5_K7_clusters.m). The classification results for K = 7 are provided in Table 2.

Table 2: K-means classification results for K = 7

Training data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	11	2421	1	11
	2	19	2520	1	19
	3	9	1792	1	9
	4	1496	6	0	6
	5	1552	0	0	0
	6	1211	3	0	3
	7	1625	0	0	0
$N_{\text{train}} = 12665$				Sum misclassified:	48
				Misclassification rate (%):	0.379
Testing data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	0	394	1	0
	2	257	0	0	0
	3	270	0	0	0
	4	3	315	1	3
	5	271	0	0	0
	6	2	426	1	2
	7	177	0	0	0
$N_{\text{test}} = 2115$				Sum misclassified:	5
				Misclassification rate (%):	0.2364

Task E6: Classification of MNIST digits using SVM. (File "Exercise_E6_Svm.m")

In this task, a supervised classifier namely the support vector machine (SVM) is used. Specifically, the soft margin SVM is used for binary classification. It solved the optimization problem

$$\begin{gathered} \text{minimize}_{\mathbf{w},b,\xi_1,..\xi_N} = \frac{1}{2}|\mathbf{w}|_2^2 + C\sum_{i=1}^{N}\xi_i \\ \text{subject to } y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1 - \xi_i, \forall i \\ \xi_i \geq 0, \forall i \\ \text{where } y_i = 1 \text{ for target } t_i = 1, \text{ and } y_i = -1 \text{ for target } t_i = 0 \end{gathered} \tag{26}$$

Further, the hyperparameter $C = 1$ (default value used in Matlab implementation). To train the binary soft margin SVM, Matlab's built-in function **fitcsvm(X,T)** is used. Because the given training data ($\mathbf{X}$) is of DXN dimension, it is transposed to match the input format of the **fitcsvm(X,T)**. For class predictions, built-in Matlab function **predict(model,X)** is used and the classification results are presented in Table 3. Using SVM, no missclassifications are observed.

Table 3: Linear SVM classification results

Training data	Predicted class	True class	# '0'	# '1'	# misclassified
	0	0	5923	0	0
	1	1	0	6742	0
$N_{\text{train}} = 12665$				Sum misclassified:	0
				Misclassification rate (%):	0.0
Testing data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	0	0	980	0	0
	1	1	0	1135	0
$N_{\text{test}} = 2115$				Sum misclassified:	0
				Misclassification rate (%):	0.0

Task E7: Classification of MNIST digits using SVM - Gaussian Kernel function. (File "Exercise_E7_Svm.m")

In this task, the 'linear' kernel function used in task E6 is replaced by the 'gaussian' kernel function. Additionally, $\beta = \sqrt{1/\sigma^2} = [0.2,0.4,0.6,...4]$ are tested to investigate the influence of β on missclassification rate. It is seen that β does not influence the missclassification rate. All the β values considered gave 0% missclassificatio rate, see Figure 6. Table 4 presents classification results using Gaussian kernel SVM with $\beta = 1$.

Table 4: Gaussian kernel function ($\beta = 1$) SVM classification results

Training data	Predicted class	True class	# '0'	# '1'	# misclassified
	0	0	5923	0	0
	1	1	0	6742	0
$N_{\text{train}} = 12665$				Sum misclassified:	0
				Misclassification rate (%):	0.0
Testing data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	0	0	980	0	0
	1	1	0	1135	0
$N_{\text{test}} = 2115$				Sum misclassified:	0
				Misclassification rate (%):	0.0

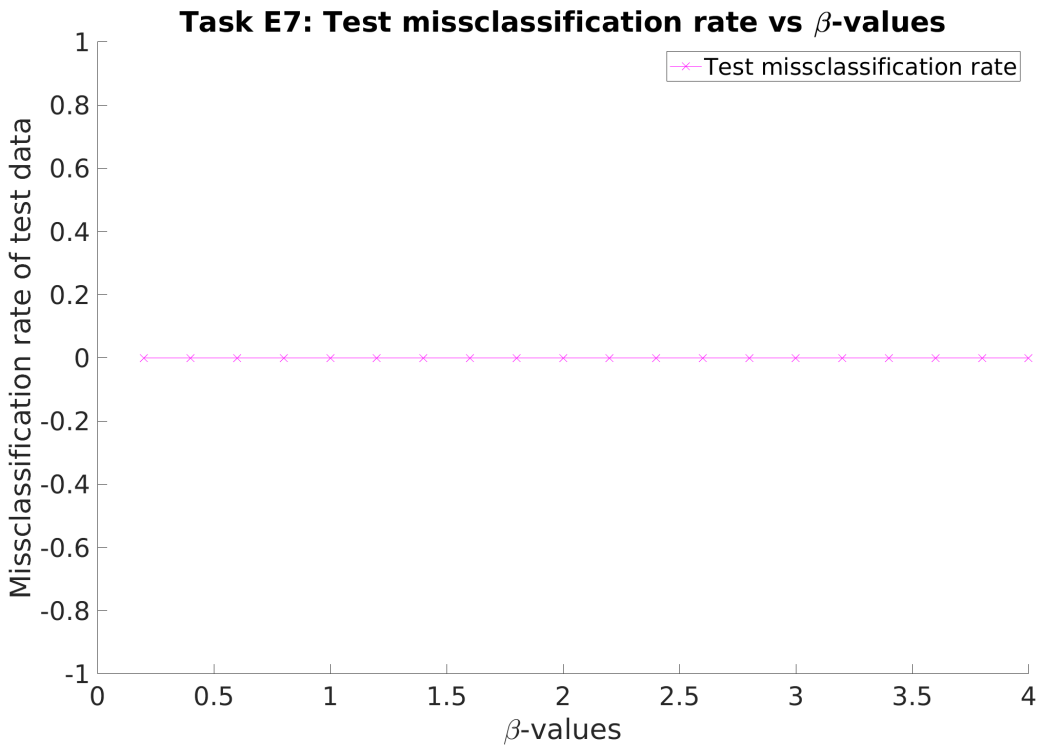


Figure 6: Task E7: Missclassification rate vs β-values.

Task E8: Classification of MNIST digits using SVM

When parameters are tuned to achieve a really low missclassification rate on both train and test datasets, the model is in effect overfitted to these datasets. Thus, the model has a higher variance. For a model to be able to classify new unseen images, i.e. model's generalization property, a balance between variance and bias is needed. This is achieved only through a trade-off between test and training errors in other words bias and variance trade-off (high bias means underfitting and high variance means overfitting). Therefore, Gaussian kernel SVM parameters that are tuned to give very low missclassification rates on both test and train datasets, cannot be expected to give the same error on new images.