IITM-CS5691 : Pattern Recognition and Machine Learning — Release Date: Oct 1, 2025

Assignment 2 — Due Date : Oct 15, 2025, **9pm**

Roll No: EE25S009

Collaborators (if any): — Name: RITABRATA MANDAL

References/sources (if any):

General Instructions:

- Use LaTeX to write-up your solutions (in the solution blocks of the source LaTeX file of this assignment), and submit the resulting **single pdf file** at Gradescope by the due date. (Note: **No late submissions** will be allowed, other than one-day late submission with 10% penalty or four-day late submission with 30% penalty! You can join Gradescope using the course entry code 6K4P43. **Within Gradescope, clearly mark your answer to each question**, else we won't be able to grade it.)
- For the programming question, please submit your code (rollno.ipynb file and rollno.py file in rollno.zip) directly in moodle, but provide your results/answers (including Jupyter notebook **with output**) in the pdf file you upload to Gradescope.
- Collaboration is encouraged, but all write-ups must be done individually and independently, and mention your collaborator(s) if any. Same rules apply for codes written for any programming assignments (i.e., write your own code; we will run plagiarism and AI detection checks on codes).
- If you have referred a book or any other online material for obtaining a solution, please cite the source. Again don't copy the source *as is* - you may use the source to understand the solution, but write-up the solution in your own words.
- For all the reasons explained in class, you cannot feed these questions into LLMs (Large Language Models like ChatGPT) and cannot use the LLMs' outputs to answer this assignment. Related to this, please also complete the self-declaration statement in the end of your answer sheet pdf.
- Please be advised that *the lesser your reliance on online materials or LLMs* for answering the questions, *the more your understanding* of the concepts will be and *the more prepared you will be for the course exams*.
- Points will be awarded based on how clear, concise and rigorous your solutions are, and how correct your answer is. The weightage of this assignment is 11% towards the overall course grade.

1. (10 points) [TOYING AROUND WITH SPECTRAL CLUSTERING] You have a dataset of four data points in two-dimensional space:
   Data Point 1: (1, 2)
   Data Point 2: (2, 3)
   Data Point 3: (3, 4)
   Data Point 4: (4, 5)

Perform spectral clustering to group these data points into two clusters. Similarity matrix can be calculated using this Gaussian kernel with bandwidth 1:

$$s(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right), \quad \sigma = 1.$$

Perform the following steps of spectral clustering, and report the output from each step.

(a) (2 points) Compute the similarity matrix A.

Solution: Given that

$$x_1 = \begin{bmatrix}1\\2\end{bmatrix}, \quad x_2 = \begin{bmatrix}2\\3\end{bmatrix}, \quad x_3 = \begin{bmatrix}3\\4\end{bmatrix}, \quad x_4 = \begin{bmatrix}4\\5\end{bmatrix}$$

Now we can write the similarity matrix A as

$$A = \begin{bmatrix} 1 & e^{-1} & e^{-4} & e^{-9} \\ e^{-1} & 1 & e^{-1} & e^{-4} \\ e^{-4} & e^{-1} & 1 & e^{-1} \\ e^{-9} & e^{-4} & e^{-1} & 1 \end{bmatrix} \approx \begin{bmatrix} 1 & 0.3679 & 0.0183 & 0.0001 \\ 0.3679 & 1 & 0.3679 & 0.0183 \\ 0.0183 & 0.3679 & 1 & 0.3679 \\ 0.0001 & 0.0183 & 0.3679 & 1 \end{bmatrix}$$

(b) (2 points) Compute the (unnormalized) Laplacian matrix L.

Solution: From similarity matrix A we can write the degree matrix

$$D = \begin{bmatrix} 1.3863 & 0 & 0 & 0 \\ 0 & 1.7541 & 0 & 0 \\ 0 & 0 & 1.7541 & 0 \\ 0 & 0 & 0 & 1.3863 \end{bmatrix}$$

Now we write the (unnormalized) Laplacian matrix L as

$$L = D - A = \begin{bmatrix} 0.3863 & -0.3679 & -0.0183 & -0.0001 \\ -0.3679 & 0.7541 & -0.3679 & -0.0183 \\ -0.0183 & -0.3679 & 0.7541 & -0.3679 \\ -0.0001 & -0.0183 & -0.3679 & 0.3863 \end{bmatrix}$$

(c) (2 points) Compute the eigenvalues and eigenvectors of L.

Solution: The eigenvalues of L are

$$\lambda_1 = 0,\ \lambda_2 = 0.2468,\ \lambda_3 = 0.7724,\ \lambda_4 = 1.2616$$

The eigenvectors corresponding to the eigenvectors are

$$e_{\lambda_1} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix};\ e_{\lambda_2} = \begin{bmatrix} 0.6567 \\ 0.2623 \\ -0.2623 \\ -0.6567 \end{bmatrix};\ e_{\lambda_3} = \begin{bmatrix} 0.5 \\ -0.5 \\ -0.5 \\ 0.5 \end{bmatrix};\ e_{\lambda_4} = \begin{bmatrix} -0.2623 \\ 0.6567 \\ -0.6567 \\ 0.2623 \end{bmatrix}$$

(d) (2 points) Select the eigenvector corresponding to the smallest eigenvalue, normalize the eigenvector so that it is unit-norm, and check if applying a threshold on it will reveal the two clusters?

Solution: The normalized eigenvector corresponding to the smallest eigenvalue(λ_1) is

$$e_{\lambda_1} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}$$

As we can see applying threshold will not reveal two clusters.

(e) (2 points) Select the eigenvector corresponding to the second smallest eigenvalue of L, normalize this eigenvector, and check if choosing a threshold of 0.7 will reveal two similar clusters in the dataset?
In the last two parts, provide the clusters obtained after performing spectral clustering and explain your reasoning for the cluster assignments based on the eigenvector and threshold.

Solution: The normalized eigenvector corresponding to the second minimum eigenvalue (λ_2) is

$$e_{\lambda_2} = \begin{bmatrix} 0.6567 \\ 0.2623 \\ -0.2623 \\ -0.6567 \end{bmatrix}$$

By selecting a threshold of 0.7, data points 1, 2, 3 and 4 are grouped into one cluster. But if we choose 0 as a threshold then eigenvector e_{λ_2} cluster data points 1 & 2 in one cluster and data points 3 & 4 in another cluster.

2. (4 points) [SPECTRAL BOUNDS] Let G be a simple undirected graph over n nodes, with d_i denoting the degree of the ith node. If the eigen values of the graph Laplacian L of G are ordered from the smallest to the largest (e.g., second smallest eigenvalue is λ_2), then show that $\lambda_2 \leqslant \frac{n}{n-1}\bar{d} \leqslant \lambda_n$. Here, $\bar{d} = \frac{1}{n}\sum_i d_i$ is the average degree of a node.
(Hint: What is the sum of all eigenvalues of L in terms of $\bar{d}$?)

Solution: We know that

$$\text{tr}(L) = \sum_i \lambda_i = \sum_i d_i = n\bar{d}$$

Since $\lambda_1 = 0$, the average of the remaining eigenvalues $\lambda_2, \ldots, \lambda_n$ is

$$\frac{1}{n-1}\sum_{i=2}^{n} \lambda_i = \frac{n}{n-1}\bar{d}$$

Because the eigenvalues are ordered, the average must lie between the smallest and largest among them, i.e.

$$\lambda_2 \leqslant \frac{1}{n-1}\sum_{i=2}^{n} \lambda_i \leqslant \lambda_n$$

Substituting the expression for the average, we get

$$\lambda_2 \leqslant \frac{n}{n-1}\bar{d} \leqslant \lambda_n \qquad \square$$

3. (10 points) [SINGULARLY PCA!] Consider a dataset of N points with each datapoint being a D-dimensional vector in $\mathbb{R}^D$. Let's assume that:

- we are in a high-dimensional setting where D >> N (e.g., D in millions, N in hundreds).
- the N × D matrix X corresponding to this dataset is already mean-centered (so that each column's mean is zero, and the covariance matrix seen in class becomes $S = \frac{1}{N}X^TX$).
- the rows (datapoints) of X are linearly independent.

Under the above assumptions, please attempt the following questions.

(a) (3 points) Whereas X is rectangular in general, XX^T and X^TX are square. Show that these two square matrices have the same set of non-zero eigenvalues. Further, argue briefly why these equal eigenvalues are all positive and N in number, and derive the multiplicity of the zero eigenvalue for both these matrices.
(Note: The square root of these equal positive eigenvalues $\{\lambda_i := \sigma_i^2\}_{i=1,\ldots,N}$ are called the singular values $\{\sigma_i\}_{i=1,\ldots,N}$ of X.)

Solution: Let $\lambda \neq 0$ and $y \in \mathbb{R}^N$ be an eigenvalue and eigenvector of XX^T, respectively. Similarly, let $\beta \neq 0$ and $z \in \mathbb{R}^D$ be an eigenvalue and eigenvector of X^TX, respectively. Then we have

$$XX^Ty = \lambda y \Rightarrow y^TXX^T = \lambda y^T$$

Also,

$$X^TXz = \beta z$$

Premultiplying the second equation by y^TX, we get

$$\begin{aligned} y^TXX^TXz &= \beta y^TXz \\ \Rightarrow \lambda y^TXz &= \beta y^TXz \quad [\because XX^Ty = \lambda y] \\ \Rightarrow \lambda &= \beta \end{aligned}$$

Hence, these two square matrices have same set of non zero eigenvalues.
Moreover we can also write

$$\lambda = \frac{\left\|X^Ty\right\|^2}{\|y\|^2} > 0; \quad \beta = \frac{\|Xz\|^2}{\|z\|^2} > 0$$

all the eigenvalues are all positive.
As the $\text{rank}(X) = N$, we can write

$$\begin{aligned} \text{rank}(XX^T) &= N = \text{rank}(X^TX) \\ \Rightarrow \text{nullity}(XX^T) = 0; &\quad \text{nullity}(X^TX) = D - N \;\; [\because \text{By rank-nullity theorem}] \end{aligned}$$

so, for XX^T the multiplicity of eigenvalue 0 is zero and for X^TX the multiplicity is $D - N$

(b) (3 points) We can choose the set of eigenvectors $\{u_i\}_{i=1,\ldots,N}$ of XX^T to be an orthonormal set and similarly we can choose an orthonormal set of eigenvectors $\{v_j\}_{j=1,\ldots,D}$ for X^TX. Briefly argue why this orthonormal choice of eigenvectors is possible. Can you choose $\{v_i\}$ such that each v_i can be computed easily from u_i and X alone (i.e., without having to do an eigenvalue decomposition of the large matrix X^TX; assume $i = 1, \ldots, N$ so that $\lambda_i > 0$ and $\sigma_i > 0$)?

(Note: $\{u_i\}, \{v_i\}$ are respectively called the left,right singular vectors of X, and computing them along with the corresponding singular values is called the Singular Value Decomposition or SVD of X.)

Solution: As XX^T and X^TX are symmetric matrices. By Spectral Theorem we conclude that it will have orthonormal set of eigenvectors.
Also given in question that $\{u_i\}_{i=1,\cdots,N}$ are the orthonormal eigenvectors of XX^T and $\{v_i\}_{i=1,\cdots,D}$ are the orthonormal eigenvectors of X^TX. Now we can write

$$XX^Tu_i = \lambda_i u_i \quad \forall\, i \in \{1, 2, \cdots, N\}$$

By premultiplying with X^T we get

$$X^TXX^Tu_i = \lambda_i X^Tu_i \Rightarrow X^TX\tilde{v}_i = \lambda_i \tilde{v}_i \quad \forall\, i \in \{1, 2, \cdots, N\}$$

where $\tilde{v}_i = X^Tu_i \quad \forall\, i \in \{1, 2, \cdots, N\}$
Now by normalizing we write

$$v_i = \frac{\tilde{v}_i}{\|v_i\|} = \frac{X^Tu_i}{\sqrt{u_i^TXX^Tu_i}} = \frac{X^Tu_i}{\sqrt{\lambda_i u_i^T u_i}} = \frac{X^Tu_i}{\sqrt{\lambda_i}} \quad \forall\, i \in \{1, 2, \cdots, N\}$$

Hence, we can easily compute v_i from u_i and X alone.

(c) (4 points) Applying PCA on the matrix X would be computationally difficult as it would involve finding the eigenvectors of $S = \frac{1}{N}X^TX$, which would take $O(D^3)$ time. Using answer to the last question above, can you reduce this time complexity to $O(N^3)$? Please provide the exact steps involved, including the exact formula for computing the normalized (unit-length) eigenvectors of S.

Solution: Below are the given steps involved to compute the eigenvalues of S

Step-1: We compute the smaller matrix

$$S'_{N\times N} = \frac{1}{N}X_{N\times D}X^T_{D\times N}$$

this step has a time complexity of $O(N^2D)$

Step-2: This step finds the eigenvalues λ_i and the corresponding normalized eigenvectors u_i that satisfies

$$S'u_i = \lambda_i u_i$$

this step has a time complexity of $O(N^3)$

Step-3: Now computing the orthonormal eigenvectors of the matrix S as follows(as computed before)

$$v_i = \frac{X^T u_i}{\sqrt{\lambda_i}}$$

this step has a time complexity of $O(ND)$

Hence, the overall complexity is $O(N^3 + N^2D + ND) = O(N^3)$. This is a massive improvement over the original $O(D^3)$.

4. (8 points) [NUMERICALLY PCA!] Consider the following dataset D of 8 datapoints:

data pt. #	x	y
1	5.51	5.35
2	20.82	24.03
3	-0.77	-0.57
4	19.30	19.38
5	14.24	12.77
6	9.74	9.68
7	11.59	12.06
8	-6.08	-5.22

You need to reduce the dataset into a single-dimensional representation. You are given the first principal component: $\text{PC1} = (-0.69, -0.72)$.

(a) (2 points) What is the xy coordinate for the datapoint reconstructed (approximated) from data pt. #2 (x=20.82, y=24.03) using the first principal component of D? What is the reconstruction error of this PC1-based approximation of data pt. #2?

Solution: The mean of the given data points is

$$\bar{x} = \begin{bmatrix} 9.29375 \\ 9.685 \end{bmatrix}$$

And as we know the reconstruction of a data point(x_2) using PC1 is given by

$$\begin{aligned}\tilde{x}_2 &= \bar{x} + \left((x_2 - \bar{x})^T PC_1\right) PC1 \\ &= \begin{bmatrix} 9.29375 \\ 9.685 \end{bmatrix} + \left(\left(\begin{bmatrix} 20.82 \\ 24.03 \end{bmatrix} - \begin{bmatrix} 9.29375 \\ 9.685 \end{bmatrix}\right)^T \begin{bmatrix} -0.69 \\ -0.72 \end{bmatrix}\right) \begin{bmatrix} -0.69 \\ -0.72 \end{bmatrix} \\ &= \begin{bmatrix} 21.908 \\ 22.8477 \end{bmatrix}\end{aligned}$$

Now, the reconstruction error of this $PC1$-based approximation of data pt. #2 is given by

$$\|\tilde{x}_2 - x_2\|_2^2 = \left\| \begin{bmatrix} 21.908 \\ 22.8477 \end{bmatrix} - \begin{bmatrix} 20.82 \\ 24.03 \end{bmatrix} \right\|^2 \approx 2.5816$$

(b) (2 points) What is the second principal component of the dataset D? How will you represent data pt. #2 as a linear combination of the two principal components? What is the reconstruction error of this $(PC1, PC2)$-based representation of data pt. #2?

Solution: The second principle component PC2 of the dataset D is $\begin{bmatrix} 0.72 \\ -0.69 \end{bmatrix}$ as PC1 $\perp$ PC2
Now, representing the data pt. #2 as a linear combination of the two principal component we get

$$\tilde{x}_2 = \bar{x} + \left((x_2 - \bar{x})^T PC1\right) PC1 + \left((x_2 - \bar{x})^T PC2\right) PC2 = x_2$$

so, the reconstruction error is 0.

(c) (2 points) Let D' be the mean-subtracted version of D. What will be the first and second principal components $PC1$ and $PC2$ of D'? What is the xy coordinate of data pt. #2 and its $PC1$-based reconstruction in D'? What is the associated reconstruction/approximation error of data pt. #2?

Solution: Since the covariance matrix already mean centered so the PC1 and PC2 will be same with the dataset D'.The xy coordinate of data pt. #2 in dataset D' is

$$x_2' = x_2 - \bar{x} = \begin{bmatrix} 11.526 \\ 14.345 \end{bmatrix}$$

Now, the PC1-based reconstruction in D′ is

$$\tilde{x}_2' = \left(x_2'^{\mathsf{T}}\mathrm{PC1}\right)\mathrm{PC1} = -18.28\mathrm{PC1} = \begin{bmatrix}12.6142\\13.1647\end{bmatrix}$$

The reconstruction error of data pt.#2 is

$$\left\|x_2' - \tilde{x}_2'\right\|^2 \approx 2.5816$$

(d) (2 points) Let D'' be a dataset extended from D by adding a third feature z to each datapoint. It so happens that this third feature is a constant value (3.5) across all 8 datapoints. Then, what will be the three principal components of D'', and what is the xyz coordinate of the PC1-based reconstruction of data pt. #2 in D'' and the associated reconstruction error?

Solution: The three principle components of D'' are

$$\mathrm{PC1} = \begin{bmatrix}-0.69\\-0.72\\0\end{bmatrix}; \quad \mathrm{PC2} = \begin{bmatrix}0.72\\-0.69\\0\end{bmatrix}; \quad \mathrm{PC3} = \begin{bmatrix}0\\0\\1\end{bmatrix}$$

Because the new feature is constant, so it has no variance and therefore contributes nothing to the first two principal directions; it simply becomes the third principal component, corresponding to the zero-variance direction.
PC1 based reconstruction of data pt. #2 in D'' is

$$\begin{aligned}\tilde{x}_2'' &= \bar{x}'' + \left(\left(x_2'' - \bar{x}''\right)^{\mathsf{T}}\mathrm{PC}_1\right)\mathrm{PC1}\\ &= \begin{bmatrix}9.29375\\9.685\\3.5\end{bmatrix} + \left(\left(\begin{bmatrix}21.908\\22.8477\\3.5\end{bmatrix} - \begin{bmatrix}9.29375\\9.685\\3.5\end{bmatrix}\right)^{\mathsf{T}}\begin{bmatrix}-0.69\\-0.72\\0\end{bmatrix}\right)\begin{bmatrix}-0.69\\-0.72\\0\end{bmatrix}\\ &= \begin{bmatrix}21.908\\22.8477\\3.5\end{bmatrix}\end{aligned}$$

Now, the reconstruction error of this PC1-based approximation of data pt. #2 is given by

$$\left\|\tilde{x}_2'' - x_2''\right\|_2^2 = \left\|\begin{bmatrix}21.908\\22.8477\\3.5\end{bmatrix} - \begin{bmatrix}20.82\\24.03\\3.5\end{bmatrix}\right\|^2 \approx 2.5816$$

5. (8 points) [TIES BETWEEN TWO ML TASKS] We will consider two tasks involving a dataset of n points, denoted as $D = \{(x_i, y_i)\}_{i=1}^{n}$. Assume that the following standard statistics have already been computed from D:

- $\bar{x} := \frac{1}{n}\sum_{i=1}^{n} x_i$, (also $\bar{y}$ defined similarly),
- $S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2$, (also S_{yy} defined similarly), and finally
- $S_{xy} = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$.

(a) (1 point) The two minimization problems in parts (b) and (c) below are each related to which ML task seen in class?
(Note: You can use the matrix-vector notation formulas relevant to these ML tasks to solve parts (b) and (c) reasonably quickly.)

Solution: Indeed. The problem in part (b) can be seen as a least square linear regression problem. And in part (c) the minimization problem can be see as orthonormal regression or 2D case of PCA.

(b) (2 points) Find the line $y = wx + b$ that minimizes the squared vertical distance of the datapoints to the line (i.e., the squared errors in y). Specifically, find the w, b that minimizes

$$\sum_{i=1}^{n}((wx_i + b) - y_i)^2.$$

Express the optimal w, b in the simplest form possible in terms of $\bar{x}, \bar{y}, S_{xx}, S_{yy}, S_{xy}$, etc.

Solution: Let

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \Phi(X) = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} = \begin{bmatrix} | & | \\ \mathbb{1} & X \\ | & | \end{bmatrix}, \quad W = \begin{bmatrix} b \\ w \end{bmatrix}$$

Then, the objective function can be written as

$$\sum_{i=1}^{n}\left((wx_i + b) - y_i\right)^2 = \|\Phi W - Y\|^2$$

The first-order necessary condition gives

$$W = (\Phi^T\Phi)^{-1}\Phi^T Y$$

Expanding each term

$$W = \left(\begin{bmatrix} -\mathbb{1}^T- \\ -X^T- \end{bmatrix} \begin{bmatrix} | & | \\ \mathbb{1} & X \\ | & | \end{bmatrix} \right)^{-1} \begin{bmatrix} -\mathbb{1}^T- \\ -X^T- \end{bmatrix} Y \Rightarrow \begin{bmatrix} b \\ w \end{bmatrix} = \left(\begin{bmatrix} n & \mathbb{1}^T X \\ X^T \mathbb{1} & X^T X \end{bmatrix} \right)^{-1} \begin{bmatrix} \mathbb{1}^T Y \\ X^T Y \end{bmatrix}$$

Now, simplifying the matrix entries

$$\mathbb{1}^T X = X^T \mathbb{1} = \sum_{i=1}^{n} x_i = n\bar{x}, \quad \mathbb{1}^T Y = \sum_{i=1}^{n} y_i = n\bar{y}$$

$$X^T X = \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} (x_i - \bar{x})^2 + 2 \sum_{i=1}^{n} x_i \bar{x} - \sum_{i=1}^{n} \bar{x}^2 = S_{xx} + n\bar{x}^2$$

$$X^T Y = \sum_{i=1}^{n} x_i y_i = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) + \bar{x} \sum_{i=1}^{n} y_i + \bar{y} \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} \bar{x}\bar{y} = S_{xy} + n\bar{x}\bar{y}$$

Substituting these back, we get

$$\begin{bmatrix} b \\ w \end{bmatrix} = \begin{bmatrix} n & n\bar{x} \\ n\bar{x} & S_{xx} + n\bar{x}^2 \end{bmatrix}^{-1} \begin{bmatrix} n\bar{y} \\ S_{xy} + n\bar{x}\bar{y} \end{bmatrix}$$

Simplifying

$$\begin{bmatrix} b \\ w \end{bmatrix} = \frac{1}{nS_{xx}} \begin{bmatrix} S_{xx} + n\bar{x}^2 & -n\bar{x} \\ -n\bar{x} & n \end{bmatrix} \begin{bmatrix} n\bar{y} \\ S_{xy} + n\bar{x}\bar{y} \end{bmatrix} = \frac{1}{nS_{xx}} \begin{bmatrix} n\bar{y}S_{xx} - n\bar{x}S_{xy} \\ nS_{xy} \end{bmatrix}$$

Finally,

$$\begin{bmatrix} b \\ w \end{bmatrix} = \begin{bmatrix} \bar{y} - \dfrac{S_{xy}}{S_{xx}} \bar{x} \\ \dfrac{S_{xy}}{S_{xx}} \end{bmatrix}$$

(c) (2 points) Find the line $y = mx + c$ that minimizes the squared perpendicular distance (i.e., shortest distance) of the datapoints to the line. Specifically, find the m, c that minimizes

$$\sum_{i=1}^{n} \frac{((mx_i + c) - y_i)^2}{m^2 + 1}.$$

Express the optimal m, c in the simplest form possible in terms of $\bar{x}, \bar{y}, S_{xx}, S_{yy}, S_{xy}$, etc.

Solution: Writing the first order necessary condition w.r.to c we get

$$\frac{\partial L}{\partial c} = \frac{\partial}{\partial c}\left(\sum_{i=1}^{n} \frac{(mx_i + c - y_i)^2}{m^2+1}\right) = 0 \Rightarrow \frac{1}{m^2+1}\sum_{i=1}^{n} 2(mx_i + c - y_i) = 0$$

$$\Rightarrow m\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} c - \sum_{i=1}^{n} y_i = 0 \Rightarrow mn\bar{x} + nc - n\bar{y} = 0$$

$$\Rightarrow c = \bar{y} - m\bar{x}$$

We substitute $c = \bar{y} - m\bar{x}$ back into the objective function. We get

$$\begin{aligned} L(m) &= \sum_{i=1}^{n} \frac{(m(x_i - \bar{x}) - (y_i - \bar{y}))^2}{m^2+1} \\ &= \frac{\sum_{i=1}^{n}\left[m^2(x_i-\bar{x})^2 - 2m(x_i-\bar{x})(y_i-\bar{y}) + (y_i-\bar{y})^2\right]}{m^2+1} \\ &= \frac{m^2 S_{xx} - 2mS_{xy} + S_{yy}}{m^2+1} \end{aligned}$$

Now writing first order condition w.r.to m we get

$$\begin{aligned} &\frac{\partial L}{\partial m} = \frac{(2mS_{xx} - 2S_{xy})(m^2+1) - 2m(m^2S_{xx} - 2mS_{xy} + S_{yy})}{(m^2+1)^2} = 0 \\ &\Rightarrow S_{xy}m^2 + (S_{xx} - S_{yy})m - S_{xy} = 0 \\ &\Rightarrow m = \frac{(S_{yy} - S_{xx}) \pm \sqrt{(S_{xx} - S_{yy})^2 + 4S_{xy}^2}}{2S_{xy}} \end{aligned}$$

Now checking second order sufficient condition we get

$$\left.\frac{\partial^2 L}{\partial^2 m}\right|_{m_+} = \sqrt{(S_{xx} - S_{yy})^2 + 4S_{xy}^2} > 0; \quad \left.\frac{\partial^2 L}{\partial^2 m}\right|_{m_-} = -\sqrt{(S_{xx} - S_{yy})^2 + 4S_{xy}^2} < 0$$

Hence, optimal values are

$$\begin{aligned} m &= \frac{(S_{yy} - S_{xx}) + \sqrt{(S_{xx} - S_{yy})^2 + 4S_{xy}^2}}{2S_{xy}} \\ c &= \bar{y} - m\bar{x} \end{aligned}$$

such that the objective is minimized.

(d) (1 point) Looking at the expressions you've derived for the above two parts, give a small example dataset where the optimizer (w, b) is identical to the optimizer (m, c), and another dataset/scenario where these two optimizers are quite different.

Solution:

1. Consider the dataset which are collinear:

$$\{(1,3),(2,5),(3,7)\}$$

For the dataset we have

$$\bar{x} = 2; \quad \bar{y} = 5$$
$$S_{xx} = 2; \quad S_{yy} = 8; \quad S_{xy} = 4$$

The optimizer (w, b) is therefore

$$w = \frac{S_{xy}}{S_{xx}} = 2; \quad b = \bar{y} - \frac{S_{xy}}{S_{xx}}\bar{x} = 1$$

And the optimizer (m, c) is therefore

$$m = \frac{3 + \sqrt{3^2 + 4 \times 2 \times 2}}{2 \times 2} = 2; \quad c = 1$$

2. The optimizers will be different whenever the data points are not perfectly collinear. Consider the dataset:

$$\{(1,0),(4,4),(1,5)\}$$

For the dataset we have

$$\bar{x} = 2; \quad \bar{y} = 3$$
$$S_{xx} = 6; \quad S_{yy} = 14; \quad S_{xy} = 3$$

The optimizer (w, b) is therefore

$$w = \frac{S_{xy}}{S_{xx}} = 0.5; \quad b = \bar{y} - \frac{S_{xy}}{S_{xx}}\bar{x} = 2$$

And the optimizer (m, c) is therefore

$$m = \frac{(14-6) + \sqrt{(6-14)^2 + 4 \times 3^2}}{2 \times 3} = 3; \quad c = -3$$

(e) (2 points) Note that the (perpendicular/shortest) distance of a point to a line from geometry is used to get each summation term above in part (c). In this question, use the method of Lagrange multiplier to algebraically derive the same results as follows.
Consider a point $p = (p_x, p_y)$. Of all the points on a line $y = mx + c$, what is the closest point to p? Also show that the distance between this closest point and p is $\frac{|(mp_x+c)-p_y|}{\sqrt{m^2+1}}$.

Solution: Let the closest point is (x, y). So, the optimization problem is

$$\min_{x,y} \sqrt{(x-p_x)^2 + (y-p_y)^2}$$
$$\text{st,} \quad y = mx + c$$

Writing the objective function with Lagrange multiplier

$$L(x, y, \lambda) = (x-p_x)^2 + (y-p_y)^2 + \lambda(y - mx - c)$$

Now, the necessary conditions

$$\frac{\partial L}{\partial x} = 0 \Rightarrow x = p_x + \frac{\lambda m}{2}$$
$$\frac{\partial L}{\partial y} = 0 \Rightarrow y = p_y - \frac{\lambda}{2}$$
$$\frac{\partial L}{\partial \lambda} = 0 \Rightarrow \lambda = 2\left(\frac{p_y - mp_x - c}{1+m^2}\right)$$

Substituting λ in x, y and get

$$x = \frac{p_x + mp_y - mc}{1+m^2}; \quad y = \frac{mp_x + m^2 p_y + c}{1+m^2}$$

So, the distance between this closest point and p is

$$\sqrt{(x-p_x)^2 + (y-p_y)^2} = \sqrt{\frac{(1+m^2)\lambda^2}{4}} = \sqrt{\frac{(p_y - mp_x - c)^2}{1+m^2}} = \frac{|(mp_x + c) - p_y|}{\sqrt{m^2+1}} \qquad \square$$

(f) (0 points) [OPTIONAL UNGRADED] What is the optimal (w, b) and (m, c) for the dataset shown below.

i	x (i.e., x_i)	y (i.e., y_i)
#1	1	1
#2	2	2
#3	4	3
#4	5	4

Solution: From the dataset we have

$$\bar{x} = 3; \quad \bar{y} = 2.5$$
$$S_{xx} = 10; \quad S_{yy} = 5; \quad S_{xy} = 7$$

The optimizer (w, b) is therefore

$$w = \frac{S_{xy}}{S_{xx}} = 0.7; \quad b = \bar{y} - \frac{S_{xy}}{S_{xx}}\bar{x} = 0.4$$

And the optimizer (m, c) is therefore

$$m = \frac{-5 + \sqrt{5^2 + 4 \times 7 \times 7}}{2 \times 7} = 0.7047; \quad c = 0.3859$$

6. (15 points) [CODING LIFE IN LOWER DIMENSIONS] You are provided with a dataset of 1797 images in a folder here - each image is 8 × 8 pixels and provided as a feature vector of length 64. You will try your hands at transforming this dataset to a lower-dimensional space using PCA.

   Please use the template.ipynb file in the same folder to prepare your solution. Provide your results/answers in the pdf file you upload to Gradescope, and submit your code separately in this moodle link. The code submitted should be a rollno.zip file containing two files: rollno.ipynb file (including your code as well as the exact same results/plots uploaded to Gradescope) and the associated rollno.py file. Write the code from scratch for PCA. The only exception is the computation of eigenvalues and eigenvectors for which you could use the numpy in-bulit function (specifically, do NOT use other functions like numpy.cov).

   (a) (5 points) Run the PCA algorithm on the given dataset. Plot the cumulative percentage variance explained by the principal components. Report the minimum number of principal components that contribute to at least 90% of the variance in the dataset.

   Solution: As we see in Figure 1 minimum 21 principle components are needed to capture 90% of the variance of the dataset.

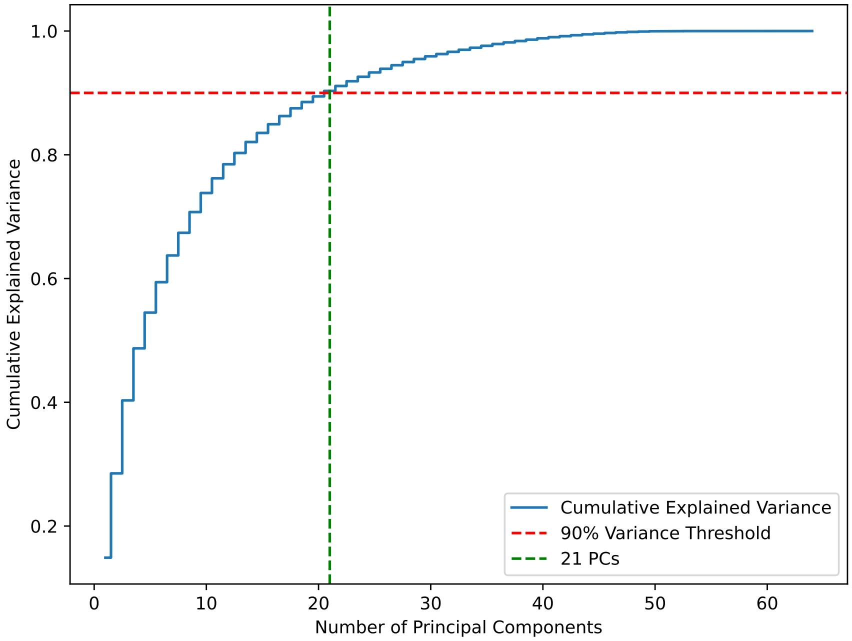

Figure 1: Explained Variance

(b) (5 points) Perform reconstruction of the dataset using a small number of components: $M \in \{2, 4, 8, 16\}$. Report the Mean Square Error (MSE) between the original data and reconstructed data, and interpret the optimal dimension $\widehat{d}$ based on the MSE values.

Solution: The reconstruction error is as follows:

M	MSE
2	13.421012
4	13.421012
8	6.121793
16	2.827183

So for $M \in \{2, 4, 8, 16\}$ the optimal dimension $\hat{d} = 16$ as the MSE is smallest among all the four cases.

(c) (5 points) Let's now apply the same code that you've written above to analyze images to understand text. Large language models (LLMs) typically analyze text by representing words as vectors, also known as embeddings. You are provided with 768-dimensional embeddings (extracted from a LLM called BERT) of 10 words in the same folder. Apply your PCA code with

$M = 2$ principal components on these embeddings to visualize the 10 words in 2-D (you are allowed to use python plotting functions for this task). Report how much percentage of variation is captured by these two principal components. What does this (2D scatterplot) visualization tell you about the embeddings of related vs. unrelated words?

Solution: The $M = 2$ principle components captures 45.47% variation. Figure 2 shows the 2D scatter plot of word embedding. From the plot I see that the humans like father, mother, woman, man, queen and king are in the left part of the plot where as foods and animals in the right part of the plot. Moreover the plot shows tight groupings of (king, queen), (man, woman), (father, mother) and (apple, banana). Which implies the PC2 captures gender or hierarchy.

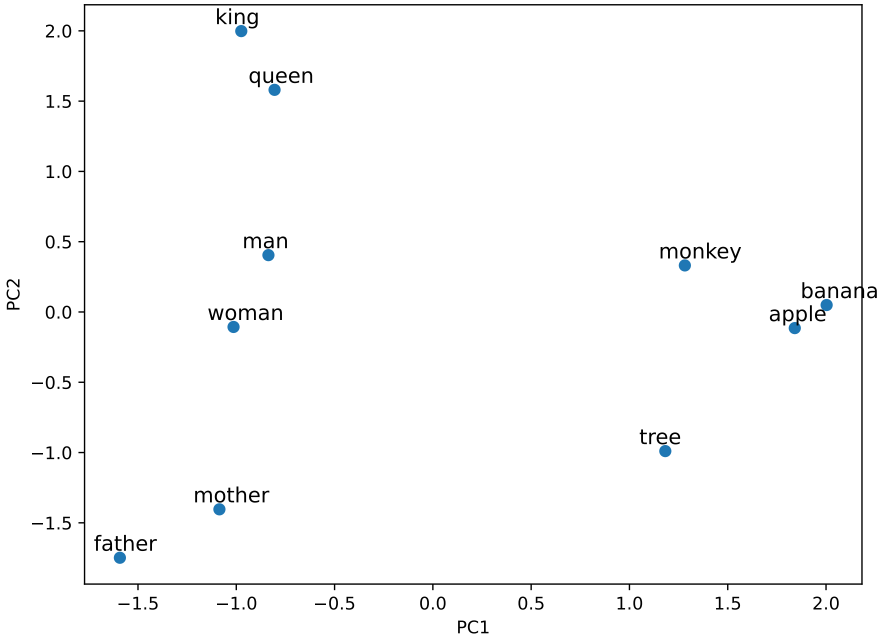

Figure 2: 2D Visualization of Word Embeddings using PCA

7. [SELF DECLARATION]
I, Ritabrata Mandal, swear on my honour that I have prepared and written the answers for this assignment and associated code by myself and have not copied it from the internet, any LLM's output, or other students.