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):
References/sources (if any):
Name: RITABRATA MANDAL

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:**

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

**Solution:**

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

**Solution:**

(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:**

(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:**

2. (4 points) [SPECTRAL BOUNDS] Let $G$ be a simple undirected graph over $n$ nodes, with $d_i$ denoting the degree of the $i$th 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 $\lambda_2$), 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:**

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 $N \times D$ 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 briefy 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:**

   (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:**

   (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:**

| 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 |

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

   You need to reduce the dataset into a single-dimensional representation. You are given the first principal component: 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:**

   (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:**

   (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:**

   (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:**

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:**

(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:**

(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:**

(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:**

(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:**

(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:**

6. (15 points) [CODING LIFE IN LOWER DIMENSIONS] You are provided with a dataset of 1797 images in a folder here - each image is $8 \times 8$ 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:**

   (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:**

(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:**

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.