CSE575 HW03, Wednesday, 11/12/2025, Due: Monday, 12/01/2025

Please note that you have to typeset your assignment using either LaTeX or Microsoft Word, and produce a PDF file for submission. Hand-written assignment (or photo of it) will not be graded. You need to submit a zip file for this assignment which contains: 1) a PDF file containing the solution/report of this assignment; 2) all of your implementation code. You should name your zip file CSE575-HW03-LastName-FirstName.zip. The zip file should be able to unzip into a folder named CSE575-HW03-LastName-FirstName.

1 Handwritten Digits Recognition with k-NN [30 pts]

In this problem, you need to implement the $\boldsymbol{k}$-NN algorithm for MNIST handwritten digits recognition. The data set is included in the “MNIST” folder of this assignment, which contains training set images, training set labels, test set images, and test set labels. Please find an introduction to the MNIST data in “MNIST.html” in the same folder.

You need to implement the algorithm from scratch using either MATLAB or Python, and to predict the digits in the test set. You will use the Euclidean distance to measure the distance between each pair of data points.

- For each value of k = 1, 3, 5, 10, 20, 30, 40, 50, 60, compute the prediction accuracy. Plot the accuracy vs the value of k.
- Write out your observations from the obtained results.

Write the observations and explanations of the result from the figure into your report (PDF) file, and include your implementation in the zip file.

Tips: You may find the following two ideas useful for this problem.

(1) We only need to compute the pairwise distance between each test data point and training data point once, and then store all the distances. Then KNN with different K values can refer to the same pairwise distances stored earlier to find the K nearest neighbors in the training data for each test data point. We do not need to recompute pairwise distances for each K. This can save a lot of time in computation.

(2) You can try library functions which compute pairwise distance between two sets of points (for example, sklearn.metrics.pairwise_distances or scipy.spatial.distance if you are using Python). Usually these library functions use matrix computation with a certain degree of parallelism so that the computation is much faster than processing the distances one at a time. If you do not have enough memory in your computer or if it is too slow to compute or store the pairwise similarity matrix between the entire training and test data, please split the training and test data into smaller blocks and compute/store the block-wise pairwise distances.

2 K-Means Clustering [30 pts]

In this problem, you will implement the K-means algorithm for clustering. You should implement from scratch, using either MATLAB or Python. Download the data set from Canvas. The data set contains 128 data points, each has 13 features. If you encounter missing values in the dataset, replace the missing entry with the mean of that feature. You should run your implementation with $\boldsymbol{K} = \mathbf{2}, \mathbf{3}, \ldots, \mathbf{9}$. For each run, initialize the cluster centers randomly among the given data, and terminate the iteration if the cluster assignment of all data points remains unchanged (in other words, each data point will be assigned to the same cluster if running more iterations). You could use slide 14 of Lecture17 as the reference for implementation.

- Plot the objective function as a function of K.
- For $K = 2$, plot the points using its first two features. Use two different colors or symbols to distinguish the two clusters.
- Write out your observations from the obtained results.

Write the observations and explanations of the result from the figure into your report (PDF) file, and include your implementation in the zip file.

3 Gaussian Mixture Model (GMM) [40 pts]

In this problem, you will implement the Gaussian Mixture Model (or Mixture of Gaussians). You should implement from scratch, using either MATLAB or Python. Use the same dataset as in the previous problem. You should run your implementation with $\boldsymbol{K} = \mathbf{2}$, and run until convergence. You can choose to terminate the iteration when the change of the log-likelihood, computed by equation (9.28) on page 439 of the textbook, is smaller than a small threshold (such as 1e-5). Please use the EM algorithm on page 438 and page 439 of the textbook as the reference for implementation. For your convenience, the Jupyter Notebook file “GMM.ipynb” is provided under the folder of this assignment, where the data is loaded and GMM parameters are initialized.

- For $K = 2$, plot the points using its first two features. Use two different colors or symbols to distinguish the two clusters. The cluster assignment is determined by the posterior $\gamma(z_{nk})$ computed by equation (9.23) of the textbook. A data point x_n is assigned to cluster 1 if $\gamma(z_{n1}) > \gamma(z_{n2})$, and it is assigned to cluster 2 otherwise.
- Write out your observations from the obtained results.

Write the observations and explanations of the result from the figure into your report (PDF) file, and include your implementation in the zip file.