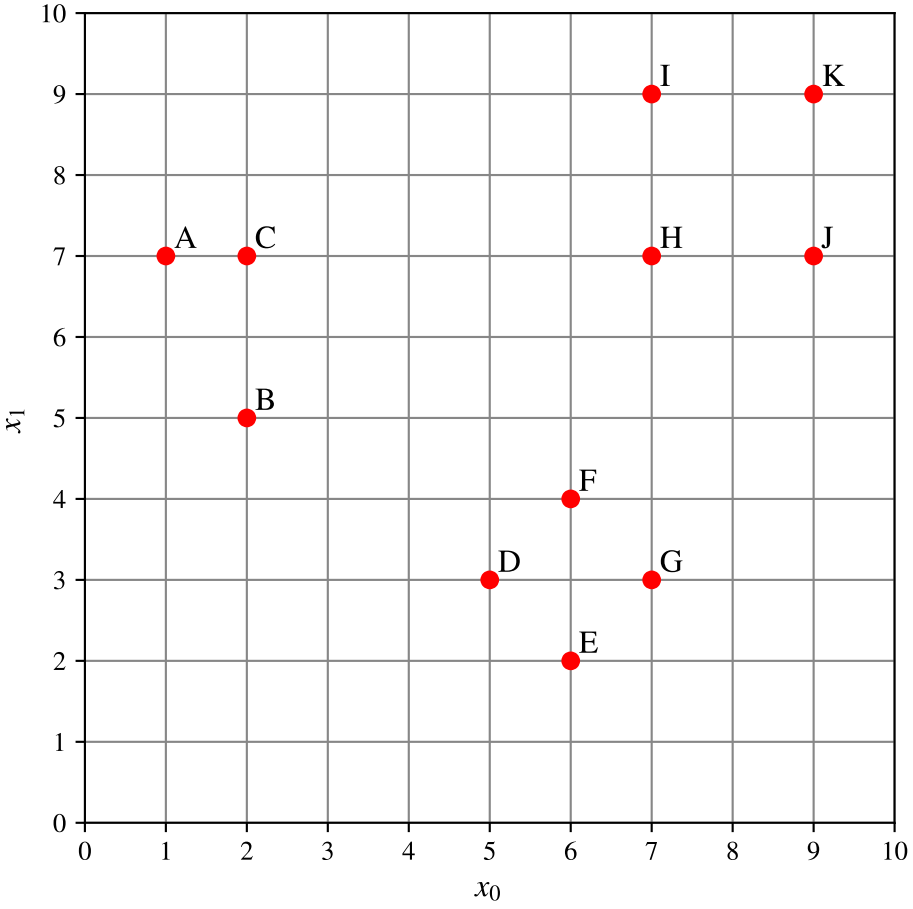
7 Unsupervised Learning

To obtain extra points for the written exam, prepare the solution for **Programming Task 7.3** using the Jupyter notebook provided and upload your work to StudOn. Students may discuss with each other while preparing the solution, but each student must submit their work individually. Refer the Supplements page in StudOn for details regarding the submission deadline.

7.1 K-Means Clustering



The scatterplot of a dataset containing 11 samples is given in the figure above. The samples are annotated alphabetically. The goal of this task is to use the K-means clustering algorithm with Euclidean distance as the distance metric to divide the dataset into 3 clusters. Assume that the points E, I and H were randomly selected as cluster centers in the initialization step. Perform the first iteration of the K-Means algorithm and answer the following questions.

i. List the labels for the points that belong to each cluster with cluster centers from initialization step.

ii. What are the new cluster centers after the first iteration of the algorithm.

iii. What is value of the objective function after the first iteration?

iv. List the labels for the points that belong to each cluster with cluster centers after the first iteration.

7.2 Programming Task: Visualization of Neural Network Activation using PCA and T-SNE

One of the motivations to perform dimensionality reduction is to visualize high dimensional data. The goal of this task is to visualizing the high dimensional activation output from the hidden layers of a fully connected network. The neural network from subtask 5.2 is provided along with the model weights. You can directly access the activations of the individual layers using the provided function: get_MNIST_activations(). This function uses the model weights to obtain model predictions on the test data and returns the activations from each layer as a sample-wise flattened vector.

i. Implement the PCA algorithm using Numpy/Scipy.

ii. Apply the PCA algorithm and reduce the input and the layer activations to two dimensions. Make a scatter plot indicating samples from each class with a different color.

iii. Using the T-SNE implementation from the scikit-learn library, reduce the input and the layer activations to two dimensions and obtain the corresponding scatter plots.

7.3 Programming Task: Color Quantization using K-Means clustering

A 24-bit color image (with 8 bits in each color channel) can have about 16.77 million (i.e. 2^{24}) unique colors. However, low power displays such as color E-ink screens support much a smaller range of colors. Color quantization refers to the task of reducing the number of unique colors used in an image. This can treated as a clustering problem where the cluster centers represent the smaller set of colors used in the quantized output image. The goal of this task is to perform color quantization using the K-Means clustering algorithm.

i. Implement the K-Means clustering algorithm using Numpy/SciPy.

ii. Read the given image. Consider each pixel as a 3-dimensional vector and run the K-Means clustering algorithm to get 64 clusters.

iii. Assign each pixel the color value of its nearest cluster center. Visualize the result.