582631 Introduction to Machine Learning, Fall 2016
Exercise set 6

Due December 15th–16th. NB: Deadline for returning solutions my email (in case you can't attend a session) is 12:15 on Friday.

Problem 1 (2+2 points)

(a) (2 points) Given a set of n numbers $x_1, \ldots, x_n$, with $x_i \in \mathbb{R}$, show that the value x^* that minimizes the *sum of squared errors*, i.e.

$$x^* = \arg\min_{x'} \sum_{i=1}^{n} (x_i - x')^2 \tag{1}$$

is given by the *average* of the x_i, i.e. $x^* = \sum_i x_i/n$.

Hint: Take the derivative of the sum of squared errors and set it to zero.

(b) (2 points) Do the same for vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$, where $\mathbf{x}_i \in \mathbb{R}^p$. In other words, show that the minimum sum of squared distances

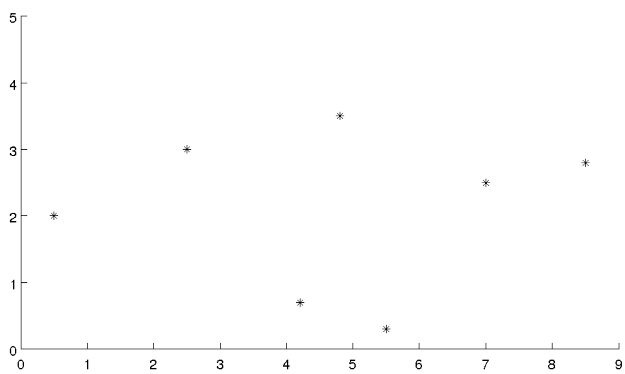
$$\mathbf{x}^* = \arg\min_{\mathbf{x}'} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{x}'\|_2^2 \tag{2}$$

is given by the average vector $\mathbf{x}^* = \sum_i \mathbf{x}_i/n$.

Hint: Recall that the squared Euclidean norm is given by $\|\mathbf{z}\|_2^2 = \sum_{j=1}^{p} \mathbf{z}_j^2$. You can break down the sum of squared errors in terms of the different dimensions and inspect the effect of the choice of $\mathbf{x}^*$ on each of the p parts in the sum.

Problem 3 (3+3 points)

We consider hierarchical clustering on a toy data set consisting of seven data points in the Euclidean plane. The data points are $p_1 = (0.5, 2.0)$, $p_2 = (2.5, 3.0)$, $p_3 = (4.2, 0.7)$, $p_4 = (5.5, 0.3)$, $p_5 = (4.8, 3.5)$, $p_6 = (7.0, 2.5)$ and $p_7 = (8.5, 2.8)$, or as a picture:



Exercises continued on the next page...

The matrix of Euclidean distances between the data points is then as follows:

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_1	0	2.24	3.92	5.28	4.55	6.52	8.04
p_2	2.24	0	2.86	4.04	2.35	4.53	6.00
p_3	3.92	2.86	0	1.36	2.86	3.33	4.79
p_4	5.28	4.04	1.36	0	3.28	2.66	3.91
p_5	4.55	2.35	2.86	3.28	0	2.42	3.77
p_6	6.52	4.53	3.33	2.66	2.42	0	1.53
p_7	8.04	6.00	4.79	3.91	3.77	1.53	0

(a) (3 points) Simulate the basic agglomerative hierarchical clustering by hand (so not using R or other software) to this data using the single linkage notion of dissimilarity between clusters. Visualise the result as a dendrogram.

Hint: Feel free to skip some calculations if you can clearly see what the next step in the algorithm is, but always calculate at least the cost of the join that you select.

(b) (3 points) Repeat the clustering using now complete linkage dissimilarity. Compare the results.

Problem 4 (8+3+3 points)

In this problem you will implement the K-means algorithm, so don't use an existing implementation (such as `kmeans` in R). Remember that you are allowed to use any programming language, but even in that case, don't use an existing implementation of the algorithm itself.

(a) (8 points) Write your own implementation of the so called Lloyd's algorithm for K-means. The algorithm is explained in the slides and Algorithm 10.1 in the textbook.

This should be a function that takes as inputs the data matrix, and outputs the final cluster means and the assignments specifying which data vectors are assigned to which cluster after convergence of the algorithm. (Use matrix operations wherever possible, avoiding explicit loops, to speed up the algorithm sufficiently for running the algorithm on the MNIST data below.)

Test the algorithm by clustering $n = 100$ random data points drawn from a bivariate standard normal distribution with mean $\boldsymbol{\mu} = (0, 0)$ and covariance $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Plot the results as a scatter plot where different clusters are shown in different colors. Repeat the experiment with the same data to see how much the clusters differ.

Hint: You may recall from the previous exercises that normal distirbuted feature vectors with diagonal covariance can be drawn by independently generating one-dimensional normal distributed features.

(b) (3 points) Using the hints and the example solutions for Exercise Set 2, load the MNIST data, and keep the first 500 training data points.[1] Run your K-means algorithm, using $K = 10$ clusters, with the initial cluster means equal to the first 10 images in the dataset. After convergence, visualize the cluster prototypes as an image showing the mean vector (mean grayscale value of each pixel).

Also visualize some of the images belonging to each cluster.

To what extent do the 10 clusters correspond to the 10 different digits?

(c) (3 points) Re-run K-means but selecting the first instance of each class as the initial cluster mean (so that the initial cluster means all represent distinct digits), and compare with the previous results.

Hint: There should be some correpondence, but some digits are definitely clustered together, and it should be clear from the visualization that the clustered images are similar even if they do not necessarily represent the same digits.

[1] You can try including more data if the run-time is tolerable.