# CS4442 & CS9542: Artificial Intelligence II – Assignment #2

Due: 23:59, March 3, 2025

## Instructions

Your submission should include: 1) a .pdf file that containing all the answers to the questions. For the questions requiring to plot figures, you should also include all the figures in the .pdf fie. 2) source code (e.g., .py files, python notebook, .m files...)

## 1 SVM and Margin [20 points]

Figure 1 shows a set of data points and a decision boundary returned by SVM. For each data point (points 1, 2, 3, 4, 5), please answer: 1) if the decision boundary will be changed if the data point is removed; 2) Explain your answer with at most 2 sentence (for each data point).

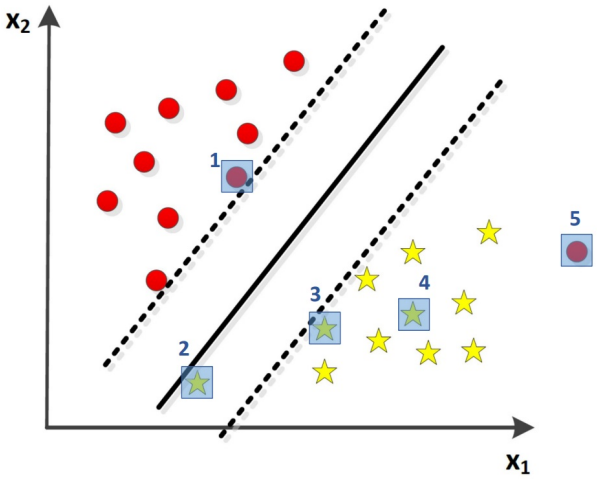


Figure 1: Support Vectors and Decision Boundary

## 2 Kernels [30 points]

In this problem, we consider constructing new kernels by combining existing kernels. Recall that for some function $k(x, z)$ to be a kernel, we need to be able to write it as a dot product of vectors in some high-dimensional feature space defined by $\phi$:

$$k(x, z) = \phi(x)^\top \phi(z)$$

Mercer's theorem gives a necessary and sufficient condition for a function $k$ to be a kernel function: its corresponding kernel matrix $K$ has to be symmetric and positive semidefinite.

Suppose that $k_1(x, z)$ and $k_2(x, z)$ are two valid kernels. For each of the cases below, state whether $k$ is also a valid kernel. If it is, prove it. If it is not, give a counterexample. You can use either Mercer's theorem, or the definition of a kernel as needed to prove it (If you use any properties on page 10 of Lecture 8, we need to prove them first).

(a) [10 points] $k(x, z) = a_1 k_1(x, z) - a_2 k_2(x, z)$, where $a_1, a_2 > 0$ are real numbers

(b) [10 points] $k(x, z) = \sqrt{k_1(x, z) k_2(x, z)}$

(c) [10 points] If $k(x,z) = e^{\frac{x^\top z}{\sigma^2}}$ is a valid kernel, prove that the Gaussian kernel $k(x,z) = e^{-\frac{||x-z||_2^2}{2\sigma^2}}$ is also a valid kernel.

# 3 PCA and Eigenface [50 points]

For this exercise, you will use principal component analysis (PCA) to analyze face images in any programming language of your choice (e.g., Python/Matlab/R). The data set `faces.dat`; each row represents an image (400 images), and each column represents a pixel ($64 \times 64 = 4096$ pixels).

(a) [5 points] Display the 200th image.

(b) [5 points] Remove the mean of the images, and then display the 100th image.

(c) [10 points] Perform PCA on the mean-centered data matrix. You can either implement PCA by yourself using eigenvalue decomposition over the sample covariance matrix, or use a existing machine learning toolbox. Sort the eigenvalues in a descending order and plot them.

(d) [5 points] You will find the last (i.e., 400th) eigenvalue is 0. Explain why.

(e) [5 points] Based on the eigenvalues, determine the dimensionality of the data you want to keep (i.e., how many principal components you want to keep), which accounts for most of the variance. Explain your reason.

(f) [10 points] Display the top-5 leading eigenvectors (corresponding to the top-5 largest eigenvalues) in 5 figures.

(g) [10 points] Display, respectively, the reconstructed 100th images using 10, 100, 200, and 399 principal components. (Hint: In Lecture 9 (page 19), we have learned that $\hat{x} = vv^\top x$ if we project $x$ into 1-dimensional space using the 1st principal component. Reconstructed $\hat{x}$ using top-$K$ principal components is a straightforward extension: $\hat{x} = \sum_{k=1}^{K} v_k v_k^\top x$)

Hint: while each vector is 4096-dimensional, in order to display the images and principal components (i.e., 3(a), 3(b), 3(f), 3(g)), you should first reshape the 4096-dimensional vector into a $64 \times 64$ matrix.