Introduction to Machine Learning
Problems: K-NN, Kernel Methods, and Support Vector Machines

Prof. Sundeep Rangan

1. *K-NN in 1D.* Consider the following dataset consisting of one-dimensional input-output pairs:

i	1	2	3	4	5
x_i	0.8	2.2	3.6	4.1	5.5
y_i	2.0	2.5	3.5	5.0	4.5

   Compute the K-NN regression estimate at the query point $x = 3.7$ for:

   (a) $K = 1$

   (b) $K = 2$

2. *Kernel smoother in 1D.* Consider the kernel

$$K(x_i, x) = \max\{0, 1 - |x - x'|\}$$

   (a) Draw the kernel $K(x_i, x)$ as a function of x for $x_i = 0$.

   (b) Using the same data as in Problem 1, compute the kernel smoothing estimate for $x = 3.7$

3. *K-NN with cosine similarity.* Consider the following dataset consisting of two-dimensional input vectors and scalar output values:

i	1	2	3	4	5
x_{i1}	1	0	1	1.5	1
x_{i2}	0	1	1	1	2
y_i	2.0	2.5	3.5	5.0	4.5

   Let the query point be $\boldsymbol{x} = [1.5, 1.5]$. Use cosine similarity to determine the nearest neighbors.

   (a) Plot the data points $\boldsymbol{x}_i$ in 2D space. Add the query point $\boldsymbol{x}$ with a different marker.

   (b) Compute the cosine similarity between the query point and each data point:

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = \frac{\boldsymbol{x}_i^{\mathsf{T}} \boldsymbol{x}}{\|\boldsymbol{x}\| \|\boldsymbol{x}_i\|}$$

   (c) Compute the K-NN regression estimate at $\boldsymbol{x}$ by averaging the y_i values of the K most similar neighbors:

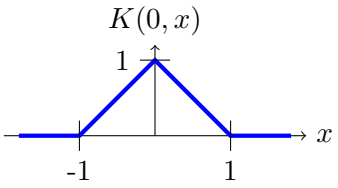

Figure 1: Problem 2. Triangular kernel

- $K = 1$
- $K = 2$

4. *Bias and variance.* Suppose that we have data

$$y_i = f_0(x_i) + w_i, \quad w_i \sim \mathcal{N}(0, \sigma^2),$$

where $f_0(x)$ is the unknown true function:

$$f_0(x) = x^2.$$

Suppose that at some test point $x_0 = 1.3$, we have the estimate

$$\widehat{f}(x_0) = \alpha_1 y_1 + \alpha_2 y_2,$$

with

$$x_1 = 1, \quad x_2 = 2, \quad \alpha_1 = 0.7, \quad \alpha_2 = 0.3.$$

(a) What is the bias: $\text{Bias}(x_0) = f_0(x_0) - \mathbb{E}[\widehat{f}(x_0)]$

(b) What is the variance: $\text{Var}(x_0) = \mathbb{E}[\widehat{f}(x_0) - \mathbb{E}(\widehat{f}(x_0))]^2$.

5. *Kernel smoother implementation.* Implement the following function kernel smoother in python. Avoid using for-loops for efficient code.

```
def kernel_smoother(Xdat, ydat, X, gam=1.0):
        """
        Kernel smoother with an RBF kernel

        Parameters
        ----------
        Xdat:  (ndat,d) array of data inputs
        ydat:  (ndat,) array of data outputs
        X:  (n,d) array of query inputs

        Returns
        -------
        yhat: (n,) array of predicted outputs at the queries
        """
        ...
        return yhat
```

6. *Outlier detection.* Suppose one uses kernel smoothing with an RBF kernel and a query value $\boldsymbol{x}$ satisfies:

$$\sum_{i=1}^{n} K(\boldsymbol{x}, \boldsymbol{x}_i) \leq \epsilon,$$

for some small $\epsilon > 0$. This condition generally implies that the query sample $\boldsymbol{x}$ is far from all the training data samples $\boldsymbol{x}_i$. This sort of check can be used to detect *outliers* that do not follow the training data. Hence, reliable predictions cannot be made. Modify the function in Problem 5 to add an output vector `outlier` that indicates which samples are outliers. Use a default value of $\epsilon = 10^{-3}$.

7. *KNN implementation.* Write a simple function in `numpy` to implement K-NN. You can use the function:

```
ind = np.argsort(A, axis=1)
```

which finds the indices for the sorting in each row. That is,

```
ind[i,k] = index of k—th smallest value in A[i,:]
```

Use the following function definition:

```
def knn(Xdat, ydat, X,k=1):
        ...
        return yhat
```

8. *Margin in 1d* Consider the data set for four points with features $\mathbf{x}_i = (x_{i1}, x_{i2})$ and binary class labels $y_i = \pm 1$.

x_{i1}	0	1	1	2
x_{i2}	0	0.3	0.7	1
y_i	-1	-1	1	1

(a) Find a linear classifier that separates the two classes. Your classifier should be of the form

$$\hat{y} = \begin{cases} 1 & \text{if } b + w_1 x_1 + w_2 x_2 > 0 \\ -1 & \text{if } b + w_1 x_1 + w_2 x_2 < 0 \end{cases}$$

State the intercept b and weights w_1 and w_2 for your classifier. Note there is no unique answer as there are multiple linear classifiers that could separate the classes.

(b) Find the maximum γ such that

$$y_i(b + w_1 x_{i1} + w_{i2} x_{i2}) \geq \gamma, \text{ for all } i,$$

for the classifier in part (a)?

(c) Compute the margin of the classifier

$$m = \frac{\gamma}{\|\mathbf{w}\|}, \quad \|\mathbf{w}\| = \sqrt{w_1^2 + w_2^2}.$$

(d) Which samples i are on the margin for your classifier?

9. *Minimization of the hinge loss.* Consider the data set with scalar features x_i and binary class labels $y_i = \pm 1$.

x_i	0	1.3	2.1	2.8	4.2	5.7
y_i	-1	-1	-1	1	-1	1

Consider a linear classifier for this data of the form,

$$\hat{y} = \begin{cases} 1 & z > 0 \\ -1 & z < 0, \end{cases} \qquad z = x - t,$$

where t is a threshold. For each threshold t, let $J(t)$ denote the sum hinge loss,

$$J(t) = \sum_i \epsilon_i, \quad \epsilon_i = \max(0, 1 - y_i z_i).$$

(a) Write a short python program to plot $J(t)$ vs. t for 100 values of t in the interval $t \in [0, 5]$.

(b) Based on the plot, what is one value of t that minimizes $J(t)$.

(c) For the value of t in part (b), find the corresponding slack variables ϵ_i.

(d) Which samples i violate the margin ($\epsilon_i > 0$) and which samples i are misclassified ($\epsilon_i > 1$).

10. *Images as matrices.* Consider an image recognition problem, where an image $\mathbf{X}$ and filter $\mathbf{W}$ are 4×4 matrices:

$$\mathbf{X} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

(a) Recall that in linear classification, the 4×4 image matrices $\mathbf{X}$ and $\mathbf{W}$ can be represented as 16-dimensional vectors, $\mathbf{x} = \text{vec}(\mathbf{X})$ and $\mathbf{w} = \text{vec}(\mathbf{W})$ by stacking the columns of the matrices vertically. What are $\mathbf{x}$ and $\mathbf{w}$ for the matrices above.

(b) What is the inner product $z = \mathbf{w}^\mathsf{T}\mathbf{x}$.

(c) What is the inner product $z = \mathbf{w}^\mathsf{T}\mathbf{x}_{\text{right}}$ where $\mathbf{x}_{\text{right}}$ is the vector corresponding to the matrix $\mathbf{X}$ right shifted by one pixel with the left column filled with zeros.

(d) What is the inner product $z = \mathbf{w}^\mathsf{T}\mathbf{x}_{\text{left}}$ where $\mathbf{x}_{\text{left}}$ is the vector corresponding to the matrix $\mathbf{X}$ left shifted by one pixel with the right column filled with zeros.

(e) Write the python command that can covert a 4×4 image matrix, `Xmat` to the 16-dimensional vector, `x`. What is the python command to go from `x` to `Xmat`.

11. Consider the data set with scalar features x_i and binary class labels $y_i = \pm 1$.

x_i	0	1	2	3
y_i	1	-1	1	-1

A support vector classifier is of the form

$$\hat{y} = \begin{cases} 1 & z > 0 \\ -1 & z < 0, \end{cases} \quad z = \sum_i \alpha_i y_i K(x_i, x),$$

where $K(x, x')$ is the radial basis function, $K(x, x') = e^{-\gamma(x-x')^2}$, and $\gamma > 0$ and $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_4]$ are parameters of the classifier.

(a) Use python to plot z vs. x and $\hat{y}$ vs. x when $\gamma = 3$ and $\boldsymbol{\alpha} = [0, 0, 1, 1]$.

(b) Repeat (a) with $\gamma = 0.3$ and $\boldsymbol{\alpha} = [1, 1, 1, 1]$.

(c) Which classifier makes more errors on the training data.