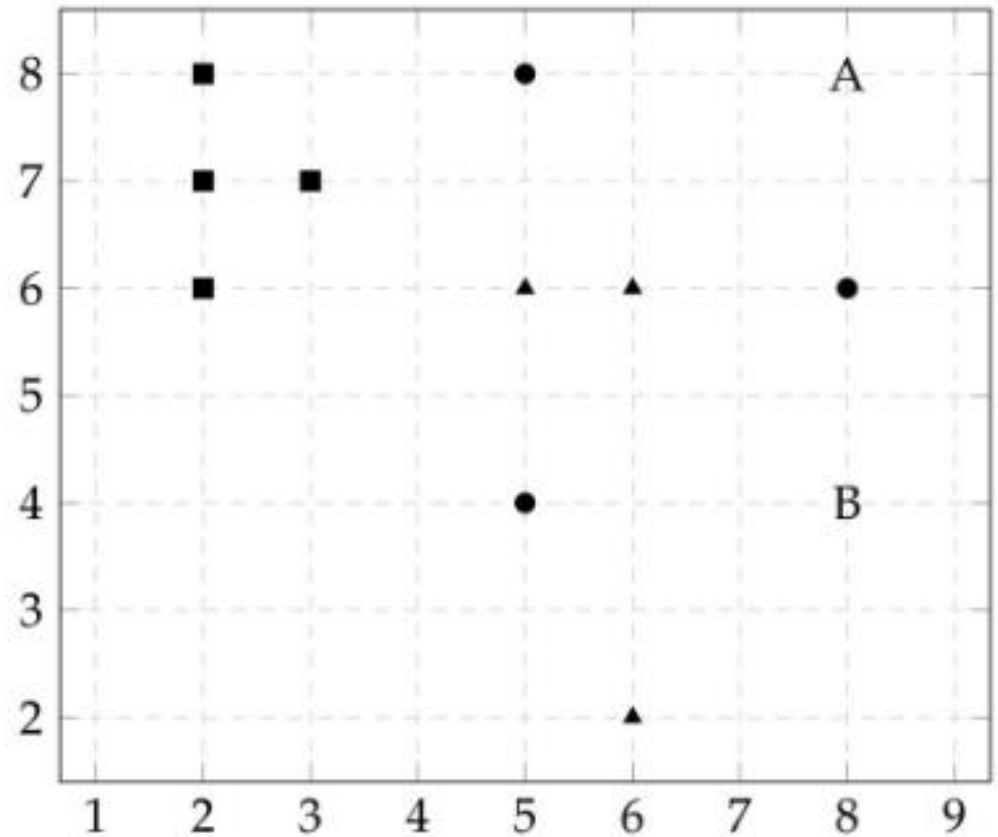
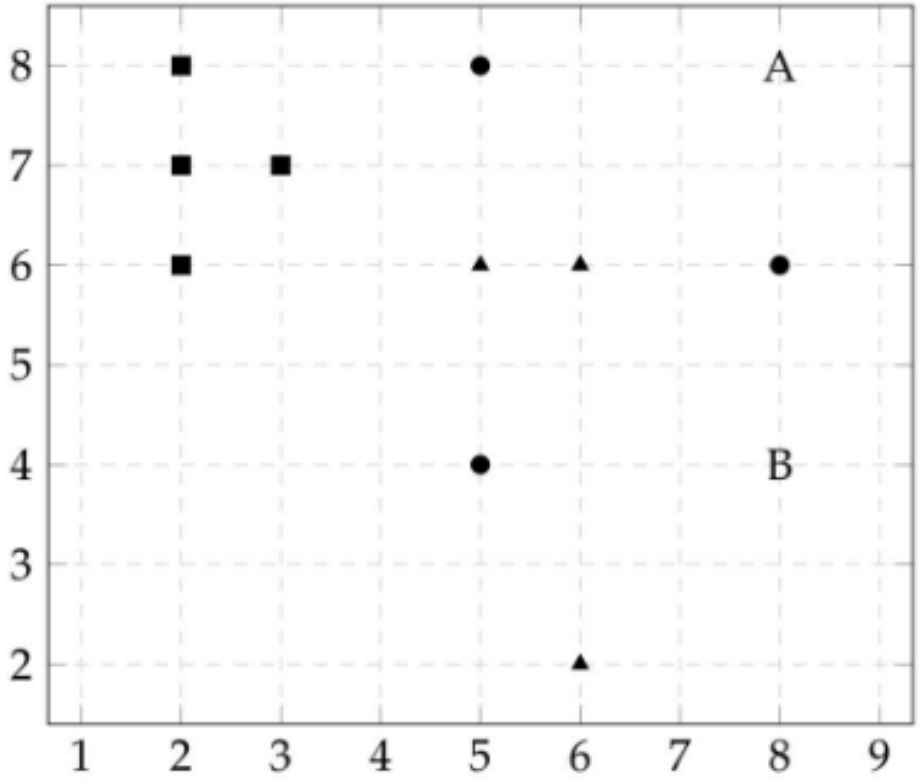
CSCI 567 Discussion
Week 4 - HW1 Review

Problem 1 Nearest Neighbor Classification (10 points)

For the data given below, squares, triangles, and circles are three different classes in the training set, and A and B are two test points with an unknown class. We denote the total number of training points as N (which equals 10) and consider K-nearest-neighbor (KNN) classifier with **L1** distance.





- Three classes:

  ■ Squares ▲ Triangles ● Circles

- Two test points:

  A and B

- K-nearest neighbor (**KNN**) with **L1** distance

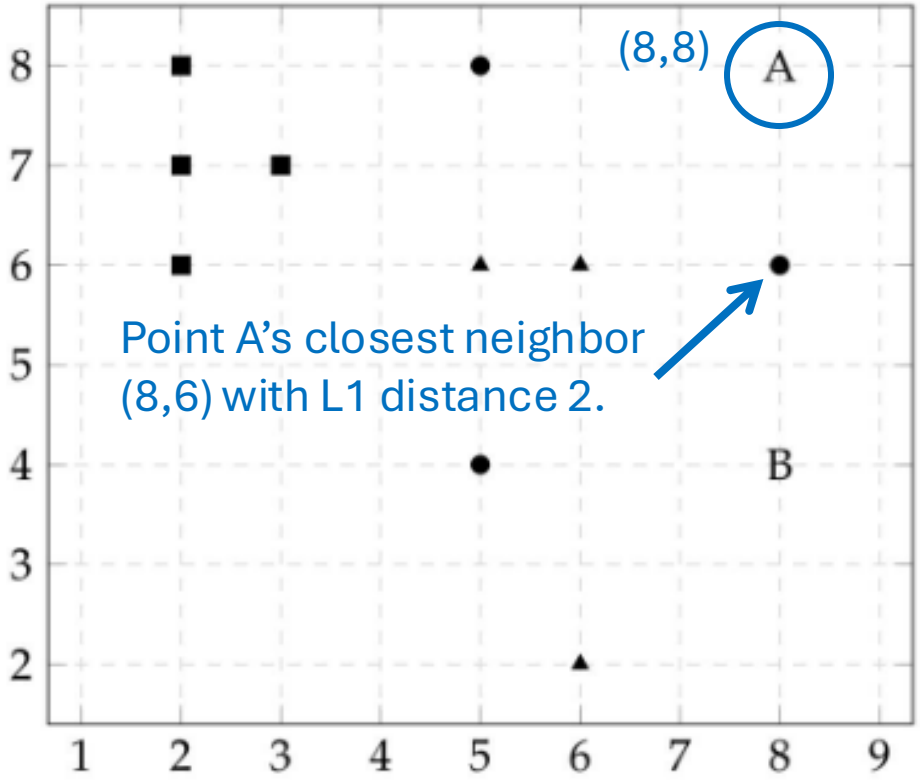
$$d((x1, y1), (x2, y2)) = | x1 - x2 | + | y1 - y2 |$$

1. What is the test point A classified as for $K = 1$? Explain briefly. (2 points)

Recall KNN with K=1:

- We only look at the single closest neighbor.
- Whatever class that neighbor belongs to → the predicted class.

(KNN: 08/29 lecture slides, page 45) 45 / 56



- Three classes:

  ■ Squares ▲ Triangles ● Circles

- Two test points:

  A and B

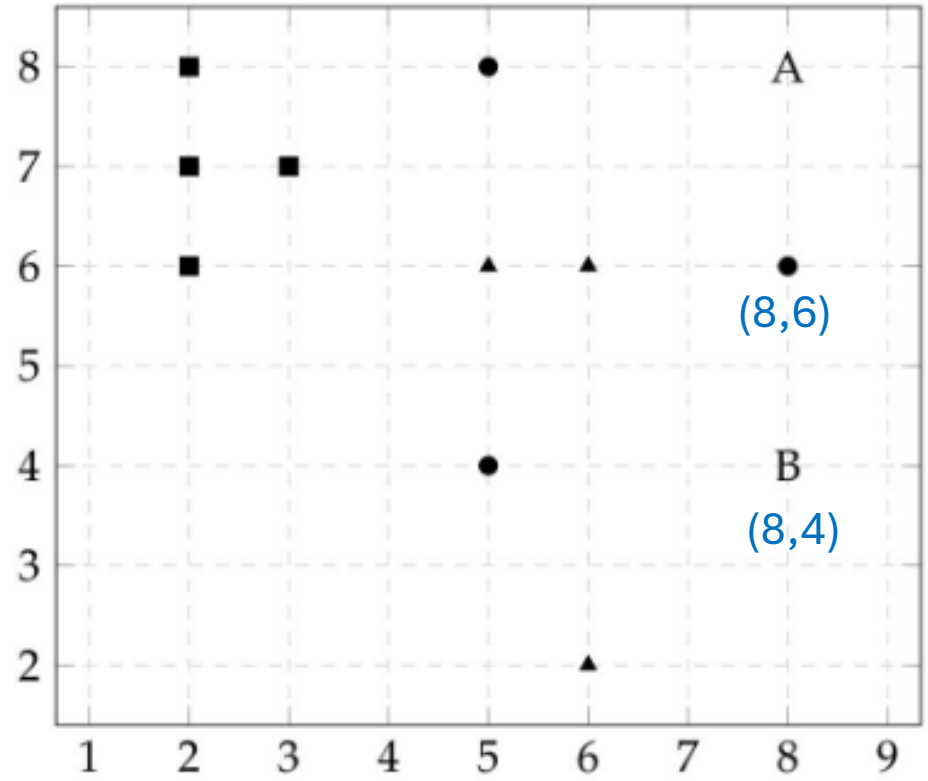
- K-nearest neighbor (**KNN**) with **L1** distance

$$d((x1, y1), (x2, y2)) = | x1 - x2 | + | y1 - y2 |$$

1. What is the test point A classified as for $K = 1$? Explain briefly. (2 points)

Answer:

- Circle.
- Explanation: The closest point to A is the circle at (8,6) (with L1 distance 2).



- Three classes:

  ■ Squares ▲ Triangles ● Circles

- Two test points:

  A and B

- K-nearest neighbor (**KNN**) with **L1** distance

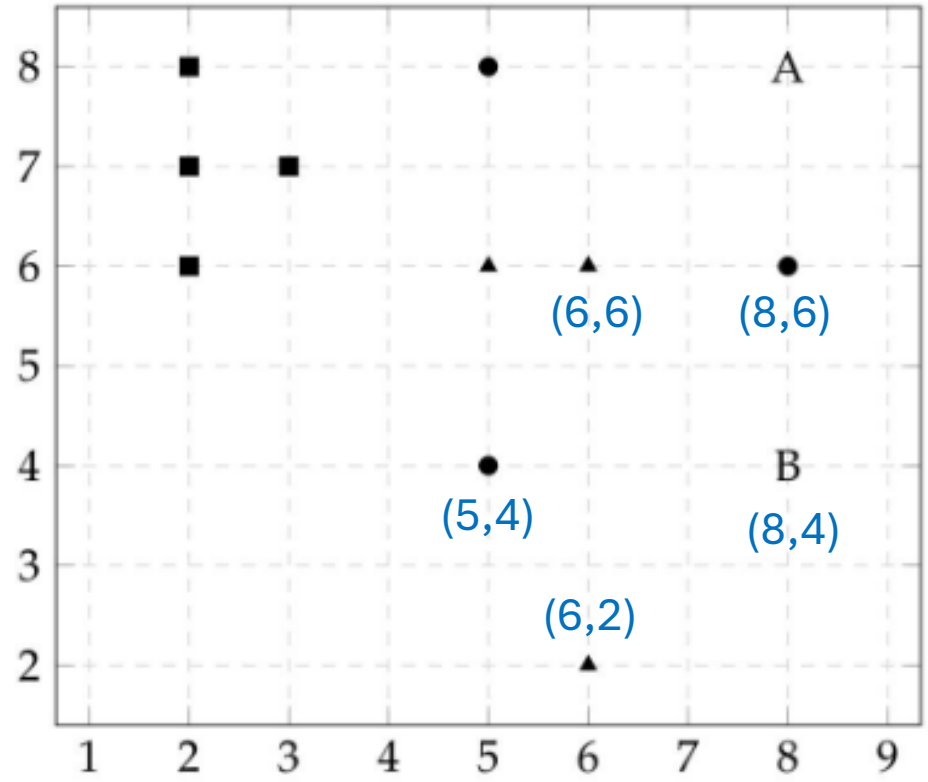
$$d((x1, y1), (x2, y2)) = | x1 - x2 | + | y1 - y2 |$$

2. What is the smallest odd value of K for KNN to predict triangle for the test point B? Explain briefly. (4 points)

K = 1:

- Neighbors: ●
  ● at (8,6) with dist = 2.

- The prediction of B is ●, so K = 1 does not work.



- Three classes:

  ■ Squares ▲ Triangles ● Circles

- Two test points:

  A and B

- K-nearest neighbor (**KNN**) with **L1** distance

$$d((x1, y1), (x2, y2)) = | x1 - x2 | + | y1 - y2 |$$

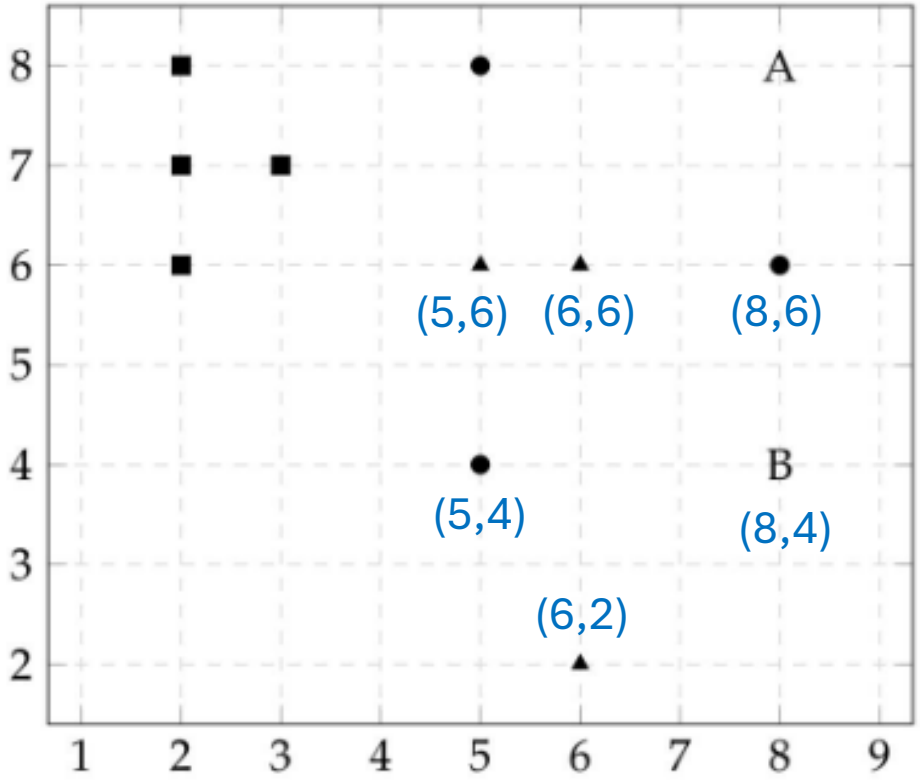
2. What is the smallest odd value of K for KNN to predict triangle for the test point B? Explain briefly. (4 points)

K = 3:

- Neighbors: ●, ●, ▲

  ● at (8,6) with dist = 2; ● at (5,4) with dist = 3; either ▲ at (6,2) with dist = 4 or ▲ at (6,6) with dist = 4.

- The prediction of B is ●, so K = 3 does not work.



- Three classes:

  ■ Squares ▲ Triangles ● Circles

- Two test points:

  A and B

- K-nearest neighbor (**KNN**) with **L1** distance

$$d((x1, y1), (x2, y2)) = | x1 - x2 | + | y1 - y2 |$$

2. What is the smallest odd value of K for KNN to predict triangle for the test point B? Explain briefly. (4 points)

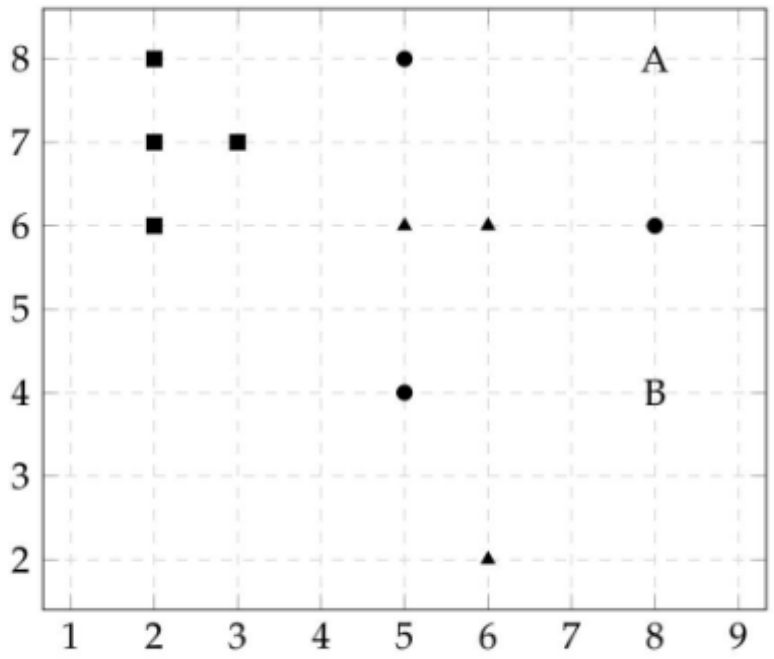
K = 5:

- Neighbors: ●, ●, ▲, ▲, ▲

  ● at (8,6) with dist = 2; ● at (5,4) with dist = 3; ▲ at (6,2) with dist = 4; ▲ at (6,6) with dist = 4; ▲ at (5,6) with dist = 5.

- The prediction of B is ▲, so K = 5 is the answer.

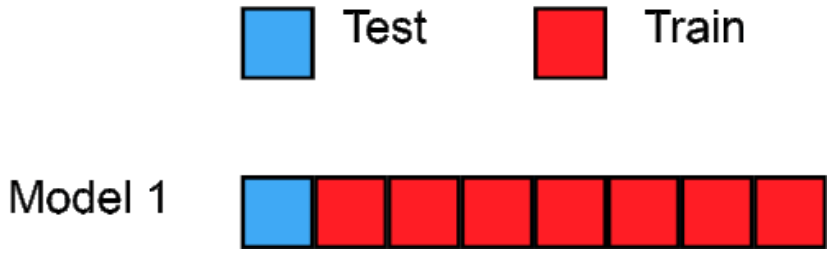
3. Suppose one performs leave-one-out validation (that is, N-fold cross validation) to choose the best hyper-parameter K. List all the points that are misclassified during the N runs when testing the hyper-parameter value $K = 1$, and report the averaged error rate for this hyper-parameter. (4 points)



Leave-one-out validation (N-fold cross validation, where N = 10):

- We have N = 10 training points.

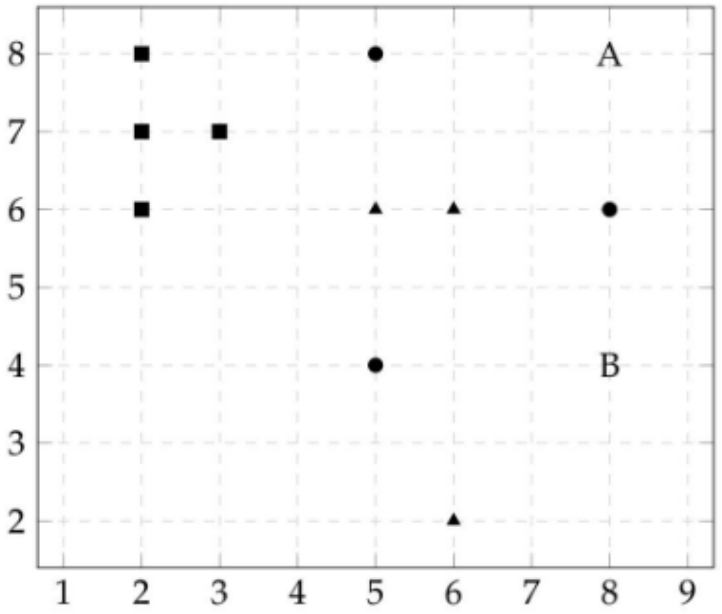
- For each **run** $i = 1 \ldots N$:
  - **Leave out 1 point**.
  - **Classify** the point with KNN (K = 1) using the other 9 points.
  - **Check** if it is correct.



From: https://en.wikipedia.org/wiki/Cross-validation_(statistics)#/media/File:LOOCV.gif

(S-fold cross-validation: 08/29 lecture slides, page 53)

3. Suppose one performs leave-one-out validation (that is, N-fold cross validation) to choose the best hyper-parameter K. List all the points that are misclassified during the N runs when testing the hyper-parameter value $K = 1$, and report the averaged error rate for this hyper-parameter. (4 points)



Run	Leave Out	Closest Neighbor (K = 1)	Classification
1	■ at (2,8)	■ at (2,7)	■ ✓
2	● at (5,8)	▲ at (5,6)	▲ X
...	...	...	...
...	● at (8,6)	▲ at (6,6)	▲ X
...	● at (5,4)	▲ at (5,6)	▲ X
...	▲ at (6,2)	● at (5,4)	● X
10	...	...	...

Answer:

The misclassified points: ▲ at (6,2) and all three ● (at (5,8), (8,6), and (5,4)).
The error rate: 4/10 = 0.4

Problem 2 Linear Regression (24 points)

2.1 (10 points) In the class, we discussed L2 regularized least square solution defined as

$$\boldsymbol{w}_* = \arg\min_{\boldsymbol{w}\in\mathbb{R}^D} \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{w}\|_2^2 \tag{1}$$

where $\boldsymbol{X} \in \mathbb{R}^{N\times D}$ is the data matrix with each row corresponding to the feature of an example, $\boldsymbol{y} \in \mathbb{R}^N$ is a vector of all the outcomes, $\|\cdot\|_2$ stands for the L2 norm, and λ is the regularization coefficient. In this problem, we consider a different regularization method:

$$\boldsymbol{w}'_* = \arg\min_{\boldsymbol{w}\in\mathbb{R}^D} \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2 + \boldsymbol{w}^T\boldsymbol{M}\boldsymbol{w} \tag{2}$$

where $\boldsymbol{M} \in \mathbb{R}^{D\times D}$ is a sysmetric positive definite matrix.

(Regularized linear regression : 09/05 lecture slides, page 49)

$$w_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + \lambda \|w\|_2^2 \quad (1)$$

$$w'_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + w^T M w \quad (2)$$

Where,

$X \in \mathbb{R}^{N \times D}$ is the data matrix

$y \in \mathbb{R}^N$ is a vector of all the outcomes

λ is the regularization coefficient

$M \in \mathbb{R}^{D \times D}$ is a sysmetric positive definite matrix.

1. Show that the new method is a generalization of the standard L2 regularization by picking a matrix M such that w'_* in Eq. (2) equals w_* in Eq. (1). (2 points)

$$\|w\|_2^2 = \sum_{j=1}^{D} w_j^2 = w^\top w$$

For example,

$$w = [2\ 3\ 4], w^T = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$$

$$w^T w = [2\ 3\ 4] \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix} = 2^2 + 3^2 + 4^2 = \|w\|_2^2$$

$$w_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + \lambda \|w\|_2^2 \qquad (1)$$

$$w'_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + w^T M w \qquad (2)$$

Where,

$X \in \mathbb{R}^{N \times D}$ is the data matrix

$y \in \mathbb{R}^N$ is a vector of all the outcomes

λ is the regularization coefficient

$M \in \mathbb{R}^{D \times D}$ is a sysmetric positive definite matrix.

1. Show that the new method is a generalization of the standard L2 regularization by picking a matrix M such that w'_* in Eq. (2) equals w_* in Eq. (1). (2 points)

Answer:

To make Eq. (2) equals to Eq. (1), we pick the matrix:

$$M = \lambda I$$

where I is the D by D identify matrix clearly makes $w^T M w$ equal to $\lambda \|w\|_2^2$.

$$w^\top M w = w^\top (\lambda I) w = \lambda w^\top w = \lambda \|w\|_2^2 \qquad \|w\|_2^2 = \sum_{j=1}^{D} w_j^2 = w^\top w$$

$$w_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + \lambda \|w\|_2^2 \quad (1)$$

$$w'_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + w^T M w \quad (2)$$

Where,

$X \in \mathbb{R}^{N \times D}$ is the data matrix

$y \in \mathbb{R}^N$ is a vector of all the outcomes

λ is the regularization coefficient

$M \in \mathbb{R}^{D \times D}$ is a sysmetric positive definite matrix.

2. Find the closed form of w'_* by writing down the gradient of $F(w) = \|Xw - y\|_2^2 + w^T M w$ and setting it to **0**. (4 points)

$$F(w) = \|Xw - y\|_2^2 + w^T M w$$

$$\|Xw - y\|_2^2 = (Xw - y)^\top (Xw - y)$$

$$= \left((Xw)^\top - y^\top\right)(Xw - y)$$

using $(AB)^\top = B^\top A^\top$

$$= (Xw)^\top (Xw) - (Xw)^\top y - y^\top (Xw) + y^\top y$$

$(Xw)^\top y$ is a scalar: $(N \times D\ D \times 1)^T\ N \times 1 \rightarrow (N \times 1)^T\ N \times 1 \rightarrow 1 \times N\ N \times 1 \rightarrow 1 \times 1$

So we have, $(Xw)^\top y = \left((Xw)^\top y\right)^\top = y^\top (Xw).$

(09/05 lecture slides, page 27)

$$w_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + \lambda \|w\|_2^2 \quad (1)$$

$$w'_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + w^T M w \quad (2)$$

Where,

$X \in \mathbb{R}^{N \times D}$ is the data matrix

$y \in \mathbb{R}^N$ is a vector of all the outcomes

λ is the regularization coefficient

$M \in \mathbb{R}^{D \times D}$ is a sysmetric positive definite matrix.

2. Find the closed form of w'_* by writing down the gradient of $F(w) = \|Xw - y\|_2^2 + w^T M w$ and setting it to $\mathbf{0}$. (4 points)

$$F(w) = \|Xw - y\|_2^2 + w^T M w$$

$$\|Xw - y\|_2^2 = (Xw - y)^\top (Xw - y)$$

$$= \left((Xw)^\top - y^\top\right)(Xw - y) \qquad \text{using } (AB)^\top = B^\top A^\top$$

$$= (Xw)^\top (Xw) - (Xw)^\top y - y^\top (Xw) + y^\top y$$

$$= (Xw)^\top (Xw) - 2y^\top (Xw) + y^\top y \qquad \text{using } (Xw)^\top y = y^\top (Xw)$$

$$= w^\top X^\top X w - 2y^\top X w + y^\top y \qquad \text{using } (AB)^\top = B^\top A^\top$$

$$w_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + \lambda \|w\|_2^2 \quad (1)$$

$$w'_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + w^T M w \quad (2)$$

Where,

$X \in \mathbb{R}^{N \times D}$ is the data matrix

$y \in \mathbb{R}^N$ is a vector of all the outcomes

λ is the regularization coefficient

$M \in \mathbb{R}^{D \times D}$ is a sysmetric positive definite matrix.

2. Find the closed form of w'_* by writing down the gradient of $F(w) = \|Xw - y\|_2^2 + w^T M w$ and setting it to **0**. (4 points)

$$F(w) = \|Xw - y\|_2^2 + w^T M w$$

$$F(w) = w^\top X^\top X w \; - \; 2 y^\top X w \; + \; y^\top y \; + \; w^\top M w.$$

The gradient:

$$\nabla_w [y^\top y] = 0$$

$$\nabla_w \left[w^\top X^\top X w\right] = 2X^\top X\, w$$

$$\nabla_w \left[-2\, y^\top X w\right] = -2\, X^\top y$$

$$\nabla_w \left[w^\top M w\right] = 2Mw$$

$$\frac{\partial \mathbf{x}^\top \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = 2\mathbf{A}\mathbf{x}$$

$$\frac{\partial \mathbf{a}^\top \mathbf{x}}{\partial \mathbf{x}} = \mathbf{a} \qquad y^\top X w = (X^\top y)^\top w$$

A is not a function of **x** and **A** is symmetric

a is not a function of **x**

(09/05 lecture slides, page 52)

$$w_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + \lambda \|w\|_2^2 \quad (1)$$

$$w'_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + w^T M w \quad (2)$$

Where,

$X \in \mathbb{R}^{N \times D}$ is the data matrix

$y \in \mathbb{R}^N$ is a vector of all the outcomes

λ is the regularization coefficient

$M \in \mathbb{R}^{D \times D}$ is a sysmetric positive definite matrix.

2. Find the closed form of w'_* by writing down the gradient of $F(w) = \|Xw - y\|_2^2 + w^T M w$ and setting it to $\mathbf{0}$. (4 points)

$$F(w) = \|Xw - y\|_2^2 + w^T M w$$

$$F(w) = w^\top X^\top X w \; - \; 2 y^\top X w \; + \; y^\top y \; + \; w^\top M w.$$

The gradient:

$$\nabla F(w) \; = \; 2X^\top X \, w \; - \; 2X^\top y \; + \; 2Mw$$

$$2X^T(Xw - y) + 2Mw.$$

$$w_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + \lambda \|w\|_2^2 \quad (1)$$

$$w'_* = \arg\min_{w \in \mathbb{R}^D} \|Xw - y\|_2^2 + w^T M w \quad (2)$$

Where,

$X \in \mathbb{R}^{N \times D}$ is the data matrix

$y \in \mathbb{R}^N$ is a vector of all the outcomes

λ is the regularization coefficient

$M \in \mathbb{R}^{D \times D}$ is a sysmetric positive definite matrix.

2. Find the closed form of w'_* by writing down the gradient of $F(w) = \|Xw - y\|_2^2 + w^T M w$ and setting it to **0**. (4 points)

Set it to **0**:

$$2X^T(Xw - y) + 2Mw.$$

$$2X^\top X w - 2X^\top y + 2Mw = 0$$

$$(X^\top X + M)w = X^\top y$$

$$Aw = b$$

Let $A := X^\top X + M$ and $b := X^\top y$

$$A^{-1}Aw = A^{-1}b$$

$w = A^{-1}b$, that is, $w'_* = \left(X^\top X + M\right)^{-1} X^\top y.$

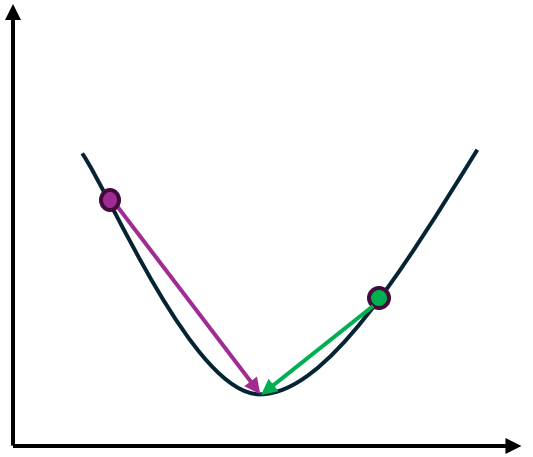
3. Recall the Newton method: $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \boldsymbol{H}_t^{-1}\nabla F(\boldsymbol{w}^{(t)})$ where $H_t = \nabla^2 F(\boldsymbol{w}^{(t)})$. Show that no matter what the initialization $\boldsymbol{w}^{(0)}$ is, Newton method always takes one step only to find the minimizer $\boldsymbol{w}'_*$ of $F(\boldsymbol{w}) = \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2 + \boldsymbol{w}^T\boldsymbol{M}\boldsymbol{w}$. (4 points)

$$\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \boldsymbol{H}_t^{-1}\nabla F(\boldsymbol{w}^{(t)})$$

$$\nabla F(w) = 2\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}) + 2\boldsymbol{M}\boldsymbol{w}$$

At **any point**, the Hessian is always $H := \nabla^2 F(w) = 2(\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{M})$.

$$H^{-1} = \left[2\left(X^\top X + M\right)\right]^{-1} = \tfrac{1}{2}\left(X^\top X + M\right)^{-1}$$



Example: Newton method jumps directly to its minimizer in one step from any start.

So applying Newton method to any initialization of $w^{(0)}$ gives,

$$\begin{aligned}
\boldsymbol{w}^{(1)} &= \boldsymbol{w}^{(0)} - (\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{M})^{-1}(\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{w}^{(0)} - \boldsymbol{y}) + \boldsymbol{M}\boldsymbol{w}^{(0)}) \\
&= \boldsymbol{w}^{(0)} - (\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{M})^{-1}\left((\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{M})\boldsymbol{w}^{(0)} - \boldsymbol{X}^T\boldsymbol{y}\right) \\
&= \left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} + \boldsymbol{M}\right)^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} = \boldsymbol{w}'_*.
\end{aligned}$$

Hence Newton method needs only **one step**, regardless of initialization.

(Newton method: 09/12 lecture slides, page 34)

2.2 (14 points) Assume we have a training set $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \mathbb{R}^D \times \mathbb{R}$, where each outcome y_n is generated by a probabilistic model $\boldsymbol{w}_*^{\mathrm{T}} \boldsymbol{x}_n + \epsilon_n$ with ϵ_n being an independent Gaussian noise with zero-mean and variance σ^2 for some $\sigma > 0$. In other words, the probability density of any outcome $y \in \mathbb{R}$ given $\boldsymbol{x}_n$ is

$$\Pr(y \mid \boldsymbol{x}_n; \boldsymbol{w}_*, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(y - \boldsymbol{w}_*^{\mathrm{T}} \boldsymbol{x}_n)^2}{2\sigma^2}\right).$$

1. Assume σ is fixed and given, find the maximum likelihood estimation for $\boldsymbol{w}_*$. In other words, first write down the joint density of the outcomes $y_1, \ldots, y_N$ given $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ as a function of the value of $\boldsymbol{w}_*$; then find the value of $\boldsymbol{w}_*$ that maximizes this density. You can assume $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ is invertible where $\boldsymbol{X}$ is the data matrix as used in Problem 2.1. (6 points)

The probability of seeing label $y_1, \ldots, y_n$ given $x_1, \ldots, x_n$ is that

$$\mathcal{P}(\boldsymbol{w}) = \prod_{n=1}^{N} \Pr(y_n \mid \boldsymbol{x}_n; \boldsymbol{w}, \sigma) = \prod_{n=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(y_n - \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n)^2}{2\sigma^2}\right).$$

(The joint density for a linear model **w**)

Now, we need to find a $\boldsymbol{w}_*$ that maximizes $\mathcal{P}(\boldsymbol{w})$

(MLE: 09/12 lecture slides, page 49)

1. Assume σ is fixed and given, find the maximum likelihood estimation for $\boldsymbol{w}_*$. In other words, first write down the joint density of the outcomes $y_1, \ldots, y_N$ given $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ as a function of the value of $\boldsymbol{w}_*$; then find the value of $\boldsymbol{w}_*$ that maximizes this density. You can assume $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ is invertible where $\boldsymbol{X}$ is the data matrix as used in Problem 2.1. (6 points)

$$\mathcal{P}(\boldsymbol{w}) = \prod_{n=1}^{N} \Pr(y_n \mid \boldsymbol{x}_n; \boldsymbol{w}, \sigma) = \prod_{n=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(y_n - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n)^2}{2\sigma^2}\right).$$

Taking the negative log, this becomes

$$F(\boldsymbol{w}) = -\ln P(w) = -\sum_{n=1}^{N} \ln\left[\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_n - w^{\top}x_n)^2}{2\sigma^2}\right)\right]$$

$$= -\sum_{n=1}^{N} \left[\ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{(y_n - w^{\top}x_n)^2}{2\sigma^2}\right]$$

$$= -\sum_{n=1}^{N} \left[-\ln\sigma - \ln\sqrt{2\pi} - \frac{(y_n - w^{\top}x_n)^2}{2\sigma^2}\right]$$

$$= \sum_{n=1}^{N} \left[\ln\sigma + \ln\sqrt{2\pi}\right] + \frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - w^{\top}x_n)^2$$

$$\log_b(xy) = \log_b x + \log_b y$$

$$\ln e^t = t$$

$$\log_b \sqrt[p]{x} = \frac{\log_b x}{p}$$

$$\log_b \frac{x}{y} = \log_b x - \log_b y$$

$$F(\boldsymbol{w}) = N\ln\sqrt{2\pi} + N\ln\sigma + \frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n)^2 = N\ln\sqrt{2\pi} + N\ln\sigma + \frac{1}{2\sigma^2}\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2$$

(MLE: 09/12 lecture slides, page 50)

1. Assume σ is fixed and given, find the maximum likelihood estimation for $\boldsymbol{w}_*$. In other words, first write down the joint density of the outcomes $y_1, \ldots, y_N$ given $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ as a function of the value of $\boldsymbol{w}_*$; then find the value of $\boldsymbol{w}_*$ that maximizes this density. You can assume $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ is invertible where $\boldsymbol{X}$ is the data matrix as used in Problem 2.1. (6 points)

$$\mathcal{P}(\boldsymbol{w}) = \prod_{n=1}^{N} \Pr(y_n \mid \boldsymbol{x}_n; \boldsymbol{w}, \sigma) = \prod_{n=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(y_n - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n)^2}{2\sigma^2}\right).$$

$$F(\boldsymbol{w}) = N \ln\sqrt{2\pi} + N \ln\sigma + \frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n)^2 = N \ln\sqrt{2\pi} + N \ln\sigma + \frac{1}{2\sigma^2}\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2$$

Maximizing P(**w**) is the same as minimizing F(**w**), the negative log of P(**w**):

- σ is fixed and given, so it becomes minimizing the last part of F(**w**): $\sum_{n=1}^{N}(y_n - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n)^2$

(the same objective as for least square regression)

Similarly,

$$\nabla_w \|y - Xw\|_2^2 = 2X^\top(Xw - y) = 0$$

$$X^\top X\, w = X^\top y$$

$$\boldsymbol{w}_* = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}.$$

2. Now consider σ as a parameter of the probabilistic model too, that is, the model is specified by both w_* and σ. Find the maximum likelihood estimation for w_* and σ. (8 points)

Follow the same steps from the previous question, we can get F(**w,** σ)**:**

$$F(\boldsymbol{w},\sigma) = N\ln\sqrt{2\pi} + N\ln\sigma + \frac{1}{2\sigma^2}\|\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y}\|_2^2.$$

MLE for w∗ and σ is the minimizer of the function F(**w,** σ).

Take the derivative over $\boldsymbol{w}$ and set it to 0 to find $\boldsymbol{w}_*$:

$$\frac{\partial F(w,\sigma)}{\partial w} = \frac{1}{\sigma^2}X^\top(Xw-y) = \frac{1}{\sigma^2}(X^\top X\,w - X^\top y)$$

$$\boldsymbol{w}_* = (\mathbf{X}^\mathrm{T}\mathbf{X})^{-1}\mathbf{X}^\mathrm{T}\boldsymbol{y}.$$

(Does not depend on σ)

Take the derivative over σ:

$$\frac{\partial F(w,\sigma)}{\partial \sigma} = \frac{N}{\sigma} - \frac{\|Xw-y\|^2}{\sigma^3}.$$

$$N\sigma^2 - \|Xw-y\|^2 = 0 \qquad \sigma > 0.$$

$$\sigma = \sqrt{\frac{\|Xw-y\|^2}{N}}$$

This depends on $\boldsymbol{w}$, so we need to find $\boldsymbol{w}_*$ first.

2. Now consider σ as a parameter of the probabilistic model too, that is, the model is specified by both $\boldsymbol{w}_*$ and σ. Find the maximum likelihood estimation for $\boldsymbol{w}_*$ and σ. (8 points)

$$F(\boldsymbol{w},\sigma) = N\ln\sqrt{2\pi} + N\ln\sigma + \frac{1}{2\sigma^2}\|\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y}\|_2^2. \qquad \sigma > 0.$$

We first fix σ and minimize over $\boldsymbol{w}$ (the same MLE from the previous question):

$$\boldsymbol{w}_* = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}.$$

For any σ > 0,

$$N\ln\sqrt{2\pi} + N\ln\sigma + \frac{1}{2\sigma^2}|Xw-y|^2 \;\geq\; N\ln\sqrt{2\pi} + N\ln\sigma + \frac{1}{2\sigma^2}|Xw_*-y|^2$$

Then we plug in $\boldsymbol{w}_*$ and minimize F($\boldsymbol{w}_*$, σ) to find σ :

$$\frac{\partial F(\boldsymbol{w}_*,\sigma)}{\partial\sigma} = \frac{N}{\sigma} - \frac{1}{\sigma^3}\|\boldsymbol{X}\boldsymbol{w}_*-\boldsymbol{y}\|_2^2 = 0.$$

$$\sigma = \frac{1}{\sqrt{N}}\|\boldsymbol{X}\boldsymbol{w}_*-\boldsymbol{y}\|_2 = \frac{1}{\sqrt{N}}\|\boldsymbol{X}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}-\boldsymbol{y}\|_2.$$

Problem 3 Linear Classifiers (16 points)

In Lecture 3 we have seen the hinge loss $\ell(z) = \max\{0, 1-z\}$, which is non-differentiable at $z = 1$. To avoid this issue, we can consider the square of hinge loss $\ell(z)^2$, which is differentiable everywhere. More specifically, given a binary dataset $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \mathbb{R}^D \times \{-1, 1\}$, we define the following new loss function for a linear model $\boldsymbol{w} \in \mathbb{R}^D$:

$$F(\boldsymbol{w}) = \frac{1}{N} \sum_{n=1}^{N} F_n(\boldsymbol{w}), \quad \text{where } F_n(\boldsymbol{w}) = \left(\max\left\{0, 1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n\right\}\right)^2. \tag{3}$$

1. For a fixed n, write down the gradient $\nabla F_n(\boldsymbol{w})$ (show your derivation), then fill in the missing details in the repeat-loop of the algorithm below which applies SGD to minimize F. (6 points)

When $1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n < 0$, $F_n(\boldsymbol{w}) = \left(\max\left\{0, 1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n\right\}\right)^2 = 0$ The gradient of it is 0.

When $1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n > 0$, $F_n(\boldsymbol{w}) = (1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n)^2$

Take the derivative by using the chain rule:

$$\frac{d}{dz} z^2 = 2z \qquad \nabla_w F_n(w) = 2(1 - y_n w^\top x_n) \cdot \nabla_w (1 - y_n w^\top x_n) = 2(1 - y_n w^\top x_n)(-y_n x_n)$$

$$\nabla F_n(\boldsymbol{w}) = -2y_n(1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n)\boldsymbol{x}_n$$

Problem 3 Linear Classifiers (16 points)

In Lecture 3 we have seen the hinge loss $\ell(z) = \max\{0, 1-z\}$, which is non-differentiable at $z = 1$. To avoid this issue, we can consider the square of hinge loss $\ell(z)^2$, which is differentiable everywhere. More specifically, given a binary dataset $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \mathbb{R}^D \times \{-1, 1\}$, we define the following new loss function for a linear model $\boldsymbol{w} \in \mathbb{R}^D$:

$$F(\boldsymbol{w}) = \frac{1}{N} \sum_{n=1}^{N} F_n(\boldsymbol{w}), \quad \text{where } F_n(\boldsymbol{w}) = \left(\max\left\{0, 1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n\right\}\right)^2. \tag{3}$$

1. For a fixed n, write down the gradient $\nabla F_n(\boldsymbol{w})$ (show your derivation), then fill in the missing details in the repeat-loop of the algorithm below which applies SGD to minimize F. (6 points)

When $1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n < 0$, $F_n(\boldsymbol{w}) = \left(\max\left\{0, 1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n\right\}\right)^2 = 0$ The gradient of it is 0.

When $1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n > 0$, $F_n(\boldsymbol{w}) = (1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n)^2$

$$\max\{0, 1 - y_n w^T x_n\} = \begin{cases} 0 & \text{if } 1 - y_n w^T x_n \leq 0, \\ 1 - y_n w^T x_n & \text{if } 1 - y_n w^T x_n > 0. \end{cases}$$

$$\nabla F_n(\boldsymbol{w}) = -2 y_n (1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n) \boldsymbol{x}_n.$$

which is also **0** when $y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n$ approaches 1

$$\nabla F_n(\boldsymbol{w}) = -2 y_n \max\{0, 1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n\} \boldsymbol{x}_n.$$

Problem 3 Linear Classifiers (16 points)

In Lecture 3 we have seen the hinge loss $\ell(z) = \max\{0, 1-z\}$, which is non-differentiable at $z = 1$. To avoid this issue, we can consider the square of hinge loss $\ell(z)^2$, which is differentiable everywhere. More specifically, given a binary dataset $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \mathbb{R}^D \times \{-1, 1\}$, we define the following new loss function for a linear model $\boldsymbol{w} \in \mathbb{R}^D$:

$$F(\boldsymbol{w}) = \frac{1}{N} \sum_{n=1}^{N} F_n(\boldsymbol{w}), \quad \text{where } F_n(\boldsymbol{w}) = \left(\max\left\{0, 1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n\right\}\right)^2. \tag{3}$$

1. For a fixed n, write down the gradient $\nabla F_n(\boldsymbol{w})$ (show your derivation), then fill in the missing details in the repeat-loop of the algorithm below which applies SGD to minimize F. (6 points)

Algorithm 1: SGD for minimizing Eq. (3)

Input: A training set $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \mathbb{R}^D \times \{-1, 1\}$, learning rate $\eta > 0$
Initialization: $\boldsymbol{w} = \boldsymbol{0}$
Repeat:
  randomly pick an example $(\boldsymbol{x}_n, y_n)$
  update $\boldsymbol{w} \leftarrow \boldsymbol{w} + 2\eta y_n \max\{0, 1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n\} \boldsymbol{x}_n$

$$\nabla F_n(\boldsymbol{w}) = -2 y_n \max\{0, 1 - y_n \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n\} \boldsymbol{x}_n.$$

(SGD: 09/12 lecture slides, page 26)

2. Next, consider modifying $F_n(\boldsymbol{w})$ as

$$F_n(\boldsymbol{w}) = \begin{cases} \left(\max\left\{0, 1 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{if } y_n = 1, \\ 0.1\left(\max\left\{0, 1 + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{else.} \end{cases} \tag{4}$$

(a) Consider a binary classification dataset of points in two dimensions as shown in Figure 1, where the red, plus signs denote samples with label $+1$, and the green, minus signs denote samples with label -1. When training a linear classifier with the modified loss in Eq. (4), which of $\boldsymbol{w}_1$ or $\boldsymbol{w}_2$ in Figure 1 do you think is more likely the resulting decision boundary? Explain briefly. (2 points)

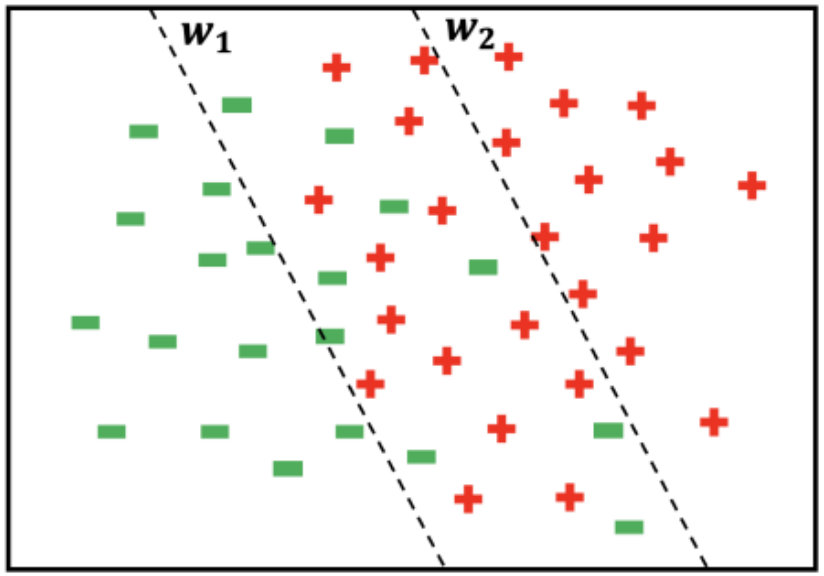


Figure 1: A binary classification task

- The loss function now does not penalize misclassifying negative points as heavily as it penalizes misclassifying positive points.

=> The model does not want to misclassify ✚ points.

2. Next, consider modifying $F_n(\boldsymbol{w})$ as

$$F_n(\boldsymbol{w}) = \begin{cases} \left(\max\left\{0, 1 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{if } y_n = 1, \\ 0.1\left(\max\left\{0, 1 + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{else.} \end{cases} \tag{4}$$

(a) Consider a binary classification dataset of points in two dimensions as shown in Figure 1, where the red, plus signs denote samples with label $+1$, and the green, minus signs denote samples with label -1. When training a linear classifier with the modified loss in Eq. (4), which of $\boldsymbol{w}_1$ or $\boldsymbol{w}_2$ in Figure 1 do you think is more likely the resulting decision boundary? Explain briefly. (2 points)

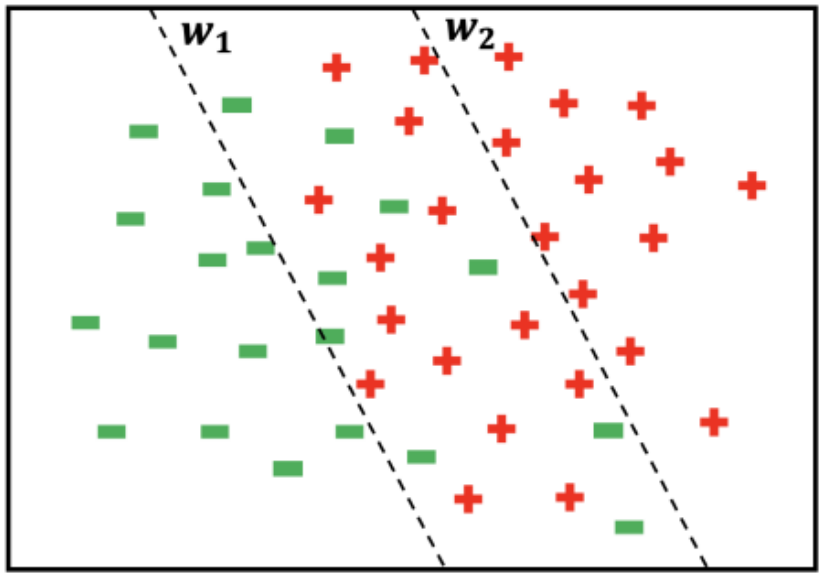


Figure 1: A binary classification task

- The dataset is not linearly separable.
  - Any linear classifier will make mistakes.
- Therefore, **w_1 is more likely.**

2. Next, consider modifying $F_n(\boldsymbol{w})$ as

$$F_n(\boldsymbol{w}) = \begin{cases} \left(\max\left\{0, 1 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{if } y_n = 1, \\ 0.1\left(\max\left\{0, 1 + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{else.} \end{cases} \tag{4}$$

(b) Based on your answer from the last question, give an example where one would want to modify the loss function in such a way. (2 points)

For example, in fraud detection (+1 represents a fraud), it is far more important to make sure that an actual fraud is not missed, and thus the loss function should give more weights to positive labels.

We accept any reasonable example where the one label represents a critical or high-stakes outcome and is far more important than the other label.

2. Next, consider modifying $F_n(\boldsymbol{w})$ as

$$F_n(\boldsymbol{w}) = \begin{cases} \left(\max\left\{0, 1 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{if } y_n = 1, \\ 0.1\left(\max\left\{0, 1 + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{else.} \end{cases} \tag{4}$$

(c) Similarly to Question 3.1, write down the gradient of this modified loss F_n, then fill in the missing details in the repeat-loop of the algorithm below which applies SGD to minimize F. (6 points)

Similar steps as in Question 3.1,

When $y_n = 1$,

- If $1 - w^\top x_n \leq 0$, the gradient is 0.
- If $1 - w^\top x_n > 0$,

$$\nabla F_n(w) = 2(1 - w^\top x_n)\,\nabla(1 - w^\top x_n) = 2(1 - w^\top x_n)(-x_n) = -2\,\max\{0, 1 - w^\top x_n\}\,x_n$$

Else,

- If $1 + w^\top x_n \leq 0$, the gradient is 0.
- If $1 + w^\top x_n > 0$,

$$\nabla F_n(w) = 0.1 \cdot 2(1 + w^\top x_n)\,\nabla(1 + w^\top x_n) = 0.2\,\max\{0, 1 + w^\top x_n\}\,x_n.$$

2. Next, consider modifying $F_n(\boldsymbol{w})$ as

$$F_n(\boldsymbol{w}) = \begin{cases} \left(\max\left\{0, 1 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{if } y_n = 1, \\ 0.1\left(\max\left\{0, 1 + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\right\}\right)^2, & \text{else.} \end{cases} \tag{4}$$

(c) Similarly to Question 3.1, write down the gradient of this modified loss F_n, then fill in the missing details in the repeat-loop of the algorithm below which applies SGD to minimize F. (6 points)

$$\nabla F_n(\boldsymbol{w}) = \begin{cases} -2\max\{0, 1 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\}\boldsymbol{x}_n, & \text{if } y_n = 1, \\ 0.2\max\{0, 1 + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\}\boldsymbol{x}_n & \text{else.} \end{cases}$$

Algorithm 2: SGD for minimizing modified loss Eq. (4)

Input: A training set $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \mathbb{R}^D \times \{-1, 1\}$, learning rate $\eta > 0$
Initialization: $\boldsymbol{w} = \mathbf{0}$
Repeat:
  randomly pick an example $(\boldsymbol{x}_n, y_n)$
  update $\boldsymbol{w} \leftarrow \boldsymbol{w} - \begin{cases} -2\eta\max\{0, 1 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\}\boldsymbol{x}_n, & \text{if } y_n = 1, \\ 0.2\eta\max\{0, 1 + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n\}\boldsymbol{x}_n & \text{else.} \end{cases}$