hw4-Solution

Graded

22 Hours, 57 Minutes Late

Student

Chiang Yi Jie

Total Points

234 / 260 pts

Question 1

Problem 1

20 / 20 pts

✔ **+ 20 pts** Correct.

Partially correct.

- **+ 5 pts** Correct data size for each one-versus-one classifier.
- **+ 5 pts** Correct relation between CPU time consumption and data size for each classifier.
- **+ 5 pts** Correct number of one-versus-one binary classifiers.
- **+ 5 pts** On the right path without any other arithmetic mistakes.

- **+ 0 pts** Incorrect or blank.

Question 2

Problem 2

20 / 20 pts

✔ **+ 20 pts** Correct.

- **+ 15 pts** Generally correct, but with minor logical mistakes, unclear explanation, or notation abuses leading to misunderstandings.
- **+ 10 pts** Generally correct, but with some serious logical mistakes.
- **+ 0 pts** Incorrect or blank.
- **− 2 pts** Minor notation mistakes / abuse.

Question 3

Problem 3

20 / 20 pts

For calculating $E_{\text{in}}(g)$:

- ✔ **+ 8 pts** Correct.
- **+ 5 pts** Generally correct with minor mistakes/ambiguity.
- **+ 2 pts** On the right path, but with serious problems.
- **+ 0 pts** Incorrect or blank.

For calculating $E_{\text{out}}(g)$:

- ✔ **+ 12 pts** Correct.
- **+ 8 pts** Generally correct with minor mistakes/ambiguity.
- **+ 4 pts** On the right path, but with serious problems.
- **+ 0 pts** Incorrect or blank.

- **− 2 pts** Minor notation mistakes / abuse.

Question 4

Problem 4

20 / 20 pts

- **+ 17 pts** Correct results with some ambiguity, e.g. do not explain why non-diagonal elements will be 0
- ✔ **+ 20 pts** Correct results along with high quality clarity
- **+ 19 pts** Correct results with flaw
- **+ 15 pts** Clear process with minor mistake that leads to incorrect result
- **+ 0 pts** Totally nonsense/Empty answer
- **+ 10 pts** Correct direction for solving the problem but end up with wrong answer
  or
  Correct answer with nonsense/not robust reasoning
- **+ 5 pts** Wrong direction to solve the problem but preserve some reasonable explanation
- **+ 1 pt** Correct answer without any reasoning
- **+ 0 pts** Click here to replace this description.
- **+ 3 pts** Wrong answer but explain why non-diagonal elements is 0 clearly

Question 5

Problem 5

14 / 20 pts

- ✔ **+ 7 pts** correct formulation
- ✔ **+ 7 pts** correct derivation
- **+ 6 pts** write down correct result clearly with known value
- **− 2 pts** ambiguous result
- **+ 0 pts** totally nonsense/empty answer

Question 6

Problem 6

18 / 20 pts

- ✔ **+ 7 pts** correctly derive lambda with self-contained explanation
- ✔ **+ 7 pts** correctly formulate the relation with given condition
- ✔ **+ 6 pts** write down what is alpha and beta with known value explicitly, and they are correct
- ✔ **− 2 pts** unclear derivation/ambiguous notation/did not clearly specify the sign of alpha
- **+ 0 pts** totally nonsense
- **− 2 pts** Ambiguity

Question 7

Problem 7

20 / 20 pts

- ✔ **+ 7 pts** Correct answer on omega(w).
- ✔ **+ 7 pts** Correct answer on new w.
- ✔ **+ 6 pts** Correct and clear process.
- **+ 3 pts** Partially unclear or wrong process.
- **+ 0 pts** Wrong answer.

Question 8

Problem 8

20 / 20 pts

- ✔ **+ 7 pts** Correct answer.
- ✔ **+ 4 pts** Consider positive points case.
- ✔ **+ 4 pts** Consider negative points case.
- ✔ **+ 5 pts** Correct and clear process.
- **+ 3 pts** Partially unclear or wrong process.
- **+ 0 pts** Wrong answer.

Question 9

Problem 9

17 / 20 pts

- **+ 10 pts** Correct answer.
- ✔ **+ 7 pts** Partially wrong process. (Most of all didn’t divide “Ein” by 5.)
- **+ 5 pts** More incorrect.
- ✔ **+ 10 pts** Consider all the cases.
- **+ 5 pts** Consider partial cases.
- **+ 0 pts** Wrong.

Question 10

Problem 10

20 / 20 pts

- ✔ **+ 20 pts** Correct Answer
- **+ 0 pts** Wrong Answer
- **– 2 pts** Select wrong page
- **– 5 pts** No clearly statement of the result.

Question 11

Problem 11

20 / 20 pts

- ✔ **+ 20 pts** Correct Answer
- **+ 15 pts** The trend of the graph is correct but missing some values in the figure.
- **+ 5 pts** The figure looks weird.
- **+ 0 pts** Wrong Answer
- **– 2 pts** Select Wrong Pages

Question 12

Problem 12

20 / 20 pts

✔ **+ 20 pts** Correct Answer

+ 15 pts Correct Answer but the description or comparison is not totally completed to some extent.

+ 5 pts Correct Answer but lack of description or comparison.

+ 5 pts The trend of the figure is not correct.

+ 0 pts Wrong Answer

Question 13

Problem 13

5 / 20 pts

+ 0 pts Wrong answer

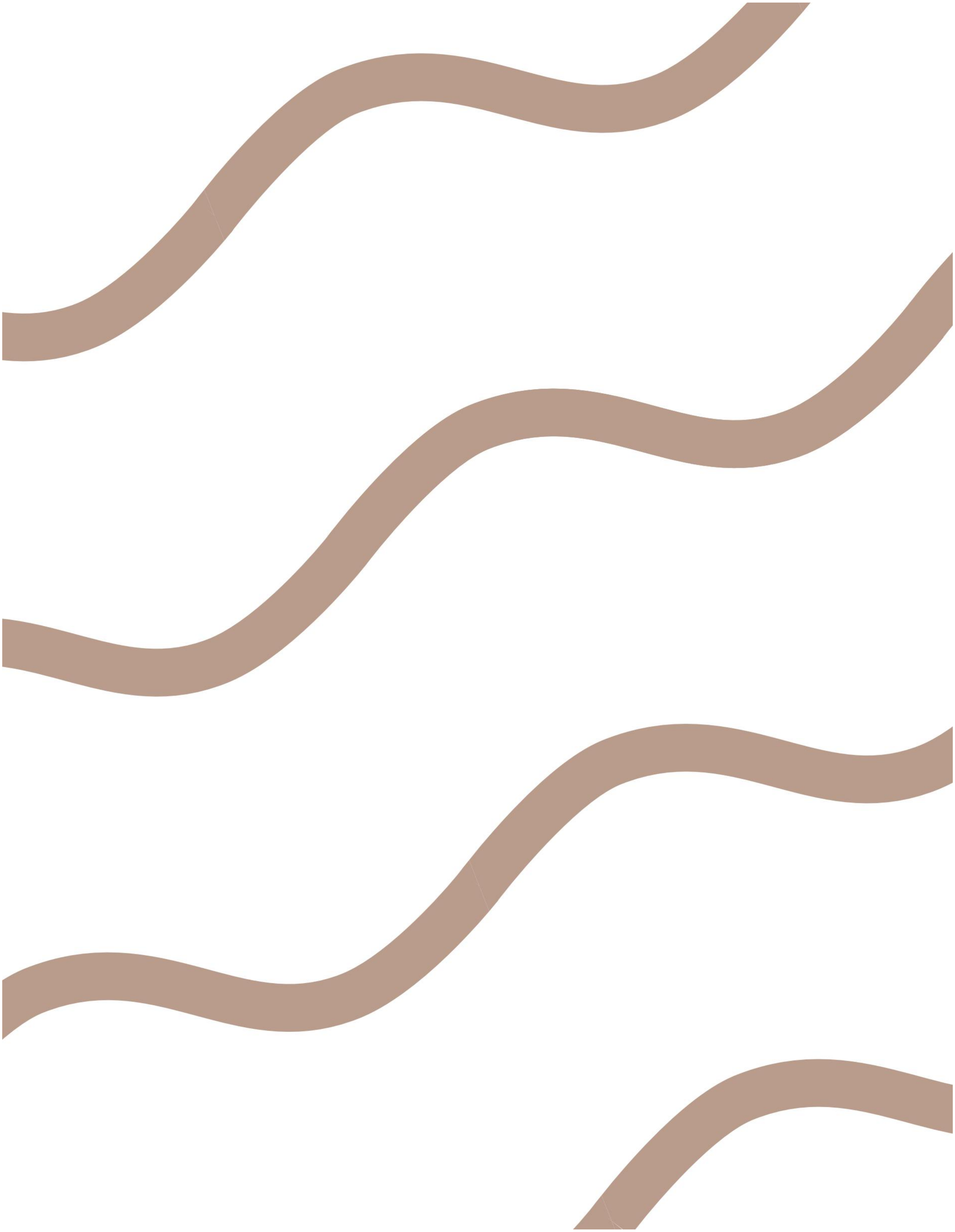
✔ **+ 5 pts** With some reasonable efforts.

+ 15 pts Only correct on deriving the sufficient condition: "If $X^T X = \alpha I$, then $\mathbf{w}_C$ solves the C-constrained linear regression problem." (Common errors are directly deriving A = B from AC = BC or deriving A = 0 if AB = 0).

+ 18 pts Solves both sides but has minor errors in proving the sufficiency.

+ 20 pts Totally correct (proving the necessary condition with some assumptions).

No questions assigned to the following page.



Question assigned to the following page: 1

1. (20 points) If some algorithm always takes a total CPU time of aN^3 for training a binary classifier on a size-N binary classification data set. Consider a size-N K-class classification data set where each class is of size N/K. What is the total CPU time needed for training a K-class classifier via one-versus-one decomposition on the data set (ignoring the minor time needed for re-labeling the data set for the sub-problems)? List your derivation steps.

(Note: This result tells you that one-versus-one may actually be computationally "cheap" because each sub-problem has fewer data.)

$C^k_2 = \frac{k(k-1)}{2}$ (classifier)

Every data set $\frac{N}{K}$.

Every classified try need ① data set: $\frac{N}{K} \times 2$

need $\frac{k(k-1)}{2}$ times classfied

$\rightarrow a\left(\frac{2N}{k}\right)^3 \times \frac{k(k-1)}{2} = 4a\,\frac{N^3(k-1)}{k^2}$ #

Question assigned to the following page: 2

2. (20 points) Consider the following matrix, which is called the Vandermonde matrix.

$$\mathrm{V} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{N-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{N-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & x_N & x_N^2 & \dots & x_N^{N-1} \end{bmatrix}$$

An N by N Vandermonde matrix has a determinant of

$$\det(\mathrm{V}) = \prod_{1 \le n < m \le N} (x_m - x_n)$$

and is thus invertible if all $\{x_n\}_{n=1}^N$ are different.

Consider some one-dimensional data $\{(x_n, y_n)\}_{n=1}^N$ where $x_n \in \mathbb{R}$ and $y_n \in \mathbb{R}$. Assume that all $\{x_n\}_{n=1}^N$ are different. Obtain a hypothesis $g(x) = \tilde{\mathbf{w}}^T \mathbf{\Phi}_Q(x)$ by applying a Q-dimensional polynomial transform $\mathbf{z}_n = \mathbf{\Phi}_Q(x_n)$, and running linear regression on $\{(\mathbf{z}_n, y_n)\}_{n=1}^N$ to get some $\tilde{\mathbf{w}}$. Use the property of the Vandermonde matrix above to prove that there exists some Q such that $E_{\text{in}}(g) = 0$ when E_{in} is measured by the squared error.

$$E_{in} = \frac{1}{N} \sum_{n=1}^{n} \| y_n - g(x_n) \|^2 = \frac{1}{N} \sum_{i=1}^{n} \| y_n - \tilde{w}^T \Phi_Q(x_n) \|^2 = 0$$

$$\sum_{i=1}^{n} \| \underbrace{y_n - \tilde{w} \Phi_Q (x_n)}_{\geq 0} \|^2 = 0$$

$$\rightarrow \quad y_n - \tilde{w} \Phi_Q (x_n) = 0$$

$$y_n = \tilde{w}^T \Phi_Q (x_n)$$

$$y = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^Q \\ 1 & x_2 & x_2^2 & \cdots & x_2^Q \\ \vdots & & & & \vdots \\ & & & & x_n^Q \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ \vdots \end{bmatrix}$$

$$\Rightarrow y_n = [1 . x_n , x_n^2 \cdots x_n^Q] \begin{bmatrix} a_0 \\ a_1 \\ \vdots \end{bmatrix} = [a_0 \, a_1 \cdots] \begin{bmatrix} 1 \\ x_n \\ x_n^2 \\ \vdots \end{bmatrix}$$

$$\Rightarrow y = \begin{bmatrix} 1 & x_1 & \cdots & x_1^Q \\ 1 & & & \\ \vdots & & & \end{bmatrix} \begin{bmatrix} w_0 \\ \vdots \end{bmatrix} \quad (a = w)$$

$$\begin{bmatrix} w_0 \\ \vdots \end{bmatrix} = \begin{bmatrix} 1 & x_1 & \cdots & x_1^Q \\ 1 & & & x_2^Q \\ \vdots & & & \end{bmatrix}^{-1} y$$

, V^{-1} must exists if w exists

There exists $Q = N-1$ such that $y_n = w^T \Phi w_Q(x_n)$. $E_{in}(g) = 0$ #

Question assigned to the following page: 3

3. (20 points) Assume that a transformer (no, not chat-Generative-Pretrained-Transformer!) peeks some one-dimensional examples and decides the following transform $\mathbf{\Phi}$ "intelligently" from the data of size N. The transform maps $x \in \mathbb{R}$ to $\mathbf{z} = (z_1, z_2, \ldots, z_N) \in \mathbb{R}^N$, where

$$(\mathbf{\Phi}(x))_n = z_n = [\![x = x_n]\!] .$$

Assume that each training and testing example is generated i.i.d. from a joint distribution $p(x, y)$ where x is sampled uniformly from $[-1, 1]$ $\boxed{\text{and } y = x + \epsilon,}$ where ϵ is independently sampled from a Gaussian distribution with mean 0 and variance 1. For simplicity, you can assume that all x_n are different in the training data set. Consider a learning algorithm that performs linear regression after the feature transform (for simplicity, please exclude $z_0 = 1$) to get a $g(x) = \tilde{\mathbf{w}}^T \mathbf{\Phi}(x)$. Consider the squared error. What is $E_{\text{in}}(g)$? What is $E_{\text{out}}(g)$? List your derivation steps.

(Note: This result tells you that "snooping" your data too much can be a bad idea.)

$$(\Phi(x))_n = Z_n = [\![x = x_n]\!]$$

$$\mathbb{X} = [x_1 . x_2 \cdots x_N]$$

$$\mathbb{Z}_1 = \begin{bmatrix} [\![x_1 = x_1]\!] \\ [\![x_1 = x_2]\!] \\ \vdots \\ [\![x_1 = x_N]\!] \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix} \rightarrow \mathbb{Z} = \begin{bmatrix} (\Phi(x))_1 & (\Phi(x))_2 \cdots \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & & 0 \\ 0 & 1 & 0 & & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & & 1 \end{bmatrix}$$

Squared Error $E_{in} = \frac{1}{N} \sum_{n=1}^{N} (y_n - \tilde{w}^T (\Phi(x))_n)^2$

$$[w_1 \cdots w_n] \begin{bmatrix} 1 & 0 & 0 & & 0 \\ 0 & 1 & 0 & & 0 \\ \vdots & 0 & 1 & \cdots & 0 \\ 0 & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & & 1 \end{bmatrix} = [y_1 . y_2 \cdots y_n]$$

There exists $\tilde{w}$ such that $y_n = \tilde{w}^T (\Phi(x))_n$

(# numbers of variables = # number of eq) $\hookrightarrow E_{in} = 0$

$$Y_{test} = [x_1 + \varepsilon_1 . x_2 + \varepsilon_2 \cdots x_n + \varepsilon_n]$$

$$\mathbb{Z}_{test} = \begin{bmatrix} [\![x_1 = x_{test1}]\!] \\ [\![x_2 = x_{test2}]\!] \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \mathbb{Z}_{test} = \underbrace{\begin{bmatrix} 0 & 0 & & 0 \\ 0 & 0 & \cdots & 0 \\ 0 & 0 & & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & 0 \end{bmatrix}}_{\#\text{ number of } x_{test}} \Big\} \#\text{ number of } x_{test}$$

$$E_{out} = E(y - \tilde{w}(\Phi(x_{test}))_n)^2$$

$$= E(y^2)$$

$$= E(x^2) + E(2x\varepsilon) + E(\varepsilon^2)$$

$$= \underbrace{\frac{1}{2}\int_{-1}^{1} x \, dx}_{\text{sampled uniformly from } [-1,1]} + 0 + Var(\varepsilon) + (E(\varepsilon))^2 = \frac{1}{2} \cdot \frac{1}{3} x^3 \Big|_{-1}^{1} + 1 + 0^2 = \frac{4}{3} \#$$

Question assigned to the following page: 4

4. (20 points) On page 20 of Lecture 13, we discussed about adding "virtual examples" (hints) to help combat overfitting. One way of generating virtual examples is to add a small noise to the input vector $\mathbf{x} \in \mathbb{R}^{d+1}$ (including the 0-th component x_0) For each $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)$ in our training data set, assume that we generate virtual examples $(\tilde{\mathbf{x}}_1, y_1), (\tilde{\mathbf{x}}_2, y_2), \ldots, (\tilde{\mathbf{x}}_N, y_N)$ where $\tilde{\mathbf{x}}_n$ is simply $\mathbf{x}_n + \boldsymbol{\epsilon}$ and each component of the noise vector $\boldsymbol{\epsilon} \in \mathbb{R}^{d+1}$ is generated i.i.d. from a uniform distribution within $[-\delta, \delta]$. The vector $\boldsymbol{\epsilon}$ is a random vector that varies for each virtual example.

Recall that when training the linear regression model, we need to calculate $\mathrm{X}^T\mathrm{X}$ first. Define the hinted input matrix

$$\mathrm{X}_h = \begin{bmatrix} | & \cdots & | & | & \cdots & | \\ \mathbf{x}_1 & \cdots & \mathbf{x}_N & \tilde{\mathbf{x}}_1 & \cdots & \tilde{\mathbf{x}}_N \\ | & \cdots & | & | & \cdots & | \end{bmatrix}^T.$$

What is the expected value $\mathbb{E}(\mathrm{X}_h^T\mathrm{X}_h)$ as a function of X and δ, where the expectation is taken over the (uniform)-noise generating process above? Prove your result.

(Note: This result may ring a bell on how such virtual examples can act like regularizers.)

$$X_h^T X_h = \begin{bmatrix} | & | & & | & & | \\ x_1 & x_2 & \cdots & \tilde{x}_1 & \cdots & \tilde{x}_N \\ | & | & & | & & | \end{bmatrix} \cdot \begin{bmatrix} - x_1 - \\ - x_2 - \\ \vdots \\ - \tilde{x}_1 - \\ \vdots \\ - \tilde{x}_N - \end{bmatrix}$$

$$\tilde{x}_n = \begin{bmatrix} x_{n1} \\ x_{n2} \\ \vdots \\ x_{nN} \end{bmatrix} + \begin{bmatrix} \varepsilon_{n1} \\ \varepsilon_{n2} \\ \vdots \\ \varepsilon_{nN} \end{bmatrix}$$

$$(x_h^T x_h)_{ij} = \sum_{n=1}^{N} x_{in}^T * x_{nj} + \sum_{n=1}^{N} (x_{in}^T + \varepsilon_{i,n})(x_{nj} + \varepsilon_{nj})$$

$$= 2\sum_{n=1}^{N} x_{i,n}^T x_{n,j} + \sum_{n=1}^{N} x_{i,n}\varepsilon_{n,j} + \varepsilon_{i,n}x_{n,j}^T + \varepsilon_{i,n}\varepsilon_{n,j}$$

$$x_h^T x_h = 2x^T x + x^T\varepsilon + \varepsilon^T x + N\varepsilon^T\varepsilon$$

$$E(x_h^T x_h) = 2x^T x + E(\varepsilon)\cdot(x^T + x) * \begin{bmatrix} | & & | \\ \vdots & \cdots & \vdots \\ | & & | \end{bmatrix} + N\begin{bmatrix} \varepsilon_1^2 & \cdots & \\ \varepsilon_1\varepsilon_2 & \varepsilon_2^2 & \cdots \\ \vdots & & \\ & & \varepsilon_N^2 \end{bmatrix}$$

$$E(\varepsilon) = \int_{-\delta}^{\delta} \frac{x}{2\delta}\,dx = 0$$

$$E(\varepsilon^2) = \int_{-\delta}^{\delta} \frac{x^2}{2\delta}\,dx = \frac{\delta^3}{3}$$

$$E(x_h^T x_h) = 2x^T x + N\frac{\delta^3}{3}\cdot I \quad \#$$

Question assigned to the following page: 5

5. (20 points) Consider the augmented error

$$E_{\text{aug}}(\mathbf{w}) = E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$$

with some $\lambda > 0$. When minimizing E_{aug} with the fixed-learning rate gradient descent algorithm with a learning rate $\eta > 0$, the update rule is

$$\mathbf{w}_{t+1} \leftarrow \alpha(\mathbf{w}_t - \beta\nabla E_{\text{in}}(\mathbf{w}_t)).$$

What are α and β? Prove your result.

(Note: You should get some $\alpha < 1$, which means that the weight vector is decayed (decreased). This is why L2 regularizer is often also called the weight-decay regularizer.)

$$E_{in} = \frac{1}{N}(Zw - y)^T(Zw - y)$$

$$\nabla E_{aug}(w) = 0 = \nabla E_{in}(w) + \frac{2\lambda}{N}w$$

$$\begin{aligned}
w_{t+1} &\leftarrow w - \eta\nabla E_{aug} \\
&= w - \eta\cdot\left(\nabla E_{in}(w) + \frac{2\lambda}{N}w\right) \\
&= \left(1 - \frac{2\lambda}{N}\right)w - \eta\cdot\nabla E_{in} \\
&= \frac{N-2\lambda}{N}\left(w_t - \eta\cdot\frac{N}{N-2\lambda}\nabla E_{in}\right)
\end{aligned}$$

$$\alpha = \frac{N-2\lambda}{N}, \quad \beta = \frac{N\eta}{N-2\lambda\eta} \#$$

Question assigned to the following page: 6

6. (20 points) Consider a one-dimensional data set $\{(x_n, y_n)\}_{n=1}^{N}$ where each $x_n \in \mathbb{R}$ and $y_n \in \mathbb{R}$. Then, solve the following one-variable regularized linear regression problem:

$$\min_{w\in\mathbb{R}} \frac{1}{N}\sum_{n=1}^{N}(w \cdot x_n - y_n)^2 + \frac{\lambda}{N}w^2.$$

If the optimal solution to the problem above is w^*, it can be shown that w^* is also the optimal solution of

$$\min_{w\in\mathbb{R}} \frac{1}{N}\sum_{n=1}^{N}(w \cdot x_n - y_n)^2 \text{ subject to } w^2 \leq C$$

with $C = (w^*)^2$. This allows us to express the relationship between C in the constrained optimization problem and λ in the augmented optimization problem for any $\lambda > 0$. In particular,

$$\lambda = \frac{\alpha}{\sqrt{C}} + \beta$$

What are α and β? Prove your result.

(Note: This should allow you to see how λ decreases [when $\lambda > 0$] as C increases [until some upper bound].)

$$\min \frac{1}{N}\sum_{n=1}^{N}(w\cdot x_n - y_n)^2 + \frac{\lambda}{N}w^2$$

$$= \min \frac{1}{N}\sum_{n=1}^{N}(w^2x_n^2 + y_n^2 - 2wx_ny_n) + \frac{\lambda}{N}w^2$$

$$\frac{\partial}{\partial w}\left(\frac{1}{N}\sum_{n=1}^{N}(w^2x_n^2 + y_n^2 - 2w\,x_ny_n) + \frac{\lambda}{N}w^2\right) = 0$$

$$\frac{1}{N}\sum_{n=1}^{N}(2wx_n^2 + 2x_ny_n) + \frac{\lambda}{N}\cdot 2w = 0$$

$$2w\sum_{n=1}^{N}x_n^2 + 2\sum_{n=1}^{N}x_ny_n + 2\lambda w = 0$$

$$w\left(\sum_{n=1}^{N}x_n^2 + \lambda\right) - \sum_{n=1}^{N}x_ny_n = 0 \quad , \quad w^* = \frac{\sum_{n=1}^{N}x_ny_n}{\sum_{n=1}^{N}x_n^2 + \lambda} \quad , \quad C = w^{*2} = \left(\frac{\sum_{n=1}^{N}x_ny_n}{\sum_{n=1}^{N}x_n^2 + \lambda}\right)^2$$

$$\sqrt{C} \times \left(\sum_{n=1}^{N}x_n^2 + \lambda\right) = \sum_{n=1}^{N}x_ny_n$$

$$\lambda = \frac{1}{\sqrt{C}}\sum_{n=1}^{N}x_ny_n - \sum_{n=1}^{N}x_n^2) = \frac{1}{\sqrt{C}}\sum_{n=1}^{N}x_ny_n - \sum_{n=1}^{N}x_n^2$$

$$\alpha = \sum_{n=1}^{N}x_ny_n \quad , \quad \beta = -\sum_{n=1}^{N}x_n^2 \quad \#$$

Question assigned to the following page: 7

7. (20 points) Scaling can affect regularization. Consider a data set $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$. Define $\Phi(\mathbf{x}) = \mathrm{V}\mathbf{x}$ where V is a diagonal matrix with the i-th diagonal component storing a *positive* value to scale the i-th feature. Now, conduct L1-regularized linear regression with the transformed data $\{(\Phi(\mathbf{x}_n), y_n)\}_{n=1}^N$.

$$\min_{\tilde{\mathbf{w}} \in \mathbb{R}^{d+1}} \frac{1}{N} \sum_{n=1}^N (\tilde{\mathbf{w}}^T \Phi(\mathbf{x}_n) - y_n)^2 + \frac{\lambda}{N} \|\tilde{\mathbf{w}}\|_1$$

The problem is equivalent to the following regularized linear regression problem on the original data with a different regularizer.

$$\min_{\mathbf{w} \in \mathbb{R}^{d+1}} \frac{1}{N} \sum_{n=1}^N (\mathbf{w}^T \mathbf{x}_n - y_n)^2 + \frac{\lambda}{N} \Omega(\mathbf{w})$$

What is $\Omega(\mathbf{w})$? How do the optimal $\tilde{\mathbf{w}}$ and the optimal $\mathbf{w}$ correspond to each other? Prove your result.

(Note: The result shows you how scaling the data effectively changes the regularizer.)

$V = \begin{bmatrix} a & & & \\ & b & & \\ & & c & \\ & & & \ddots \end{bmatrix}$ diagonal matric with i component

$\min_{\tilde{w} \in \mathbb{R}^{d+1}} \frac{1}{N} \sum_{n=1}^{N} (\tilde{w}^T (Vx_n) - y_n)^2 + \frac{\lambda}{N} \|\tilde{w}\|_1$, $X = \begin{bmatrix} - x_1 - \\ - x_2 - \\ \vdots \end{bmatrix}$, $y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \end{bmatrix}$

equivelent to

$\min_{\tilde{w} \in \mathbb{R}^{d+1}} \frac{1}{N} \sum_{n=1}^{N} (w^T x_n - y_n)^2 + \frac{\lambda}{N} \Omega(w)$

$V^T = V$ (diagonal)

$\Rightarrow \tilde{w}^T V = w^T \Rightarrow V^T \tilde{w} = w \Rightarrow \tilde{w} = V^{-1} w$

$\Omega(w) = \|\tilde{w}\|_1 = \|V^{-1} w\|_1$ #

Question assigned to the following page: 8

8. (20 points) Consider a binary classification algorithm $\mathcal{A}_{\text{minority}}$, which returns a constant classifier that always predicts the minority class (i.e., the class with fewer instances in the data set that it sees). As you can imagine, the returned classifier is the worst-E_{in} one among all constant classifiers. Consider the 0/1 error. For a binary classification data set with N positive examples and N negative examples, what is $E_{\text{loocv}}(\mathcal{A}_{\text{minority}})$? Prove your result.

(Note: This result may tell you that in some special situations, leave-one-out cross-validation is not always trustworthy.)

(1)

Choose +1 as D_{test} :

\# number of $D_{positive}$ would be $(N-1)$ and it would become minority class. $err^{+} = 0$ #

(2)

Choose −1 as D_{test} :

\# number of $D_{negative}$ would be $(N-1)$ and it would become minority class. $err^{-} = 0$

$\Rightarrow E_{loocv}(A_{minority}) = \frac{0}{2N} = 0$

It indicates that although $E_{loocv} = 0$, Loocv gives a misleading estimation since it evalnate the ability of handling specific data.

Although $E_{loocv} = 0$, it doesn't truely indicate the power of $A_{minority}$.

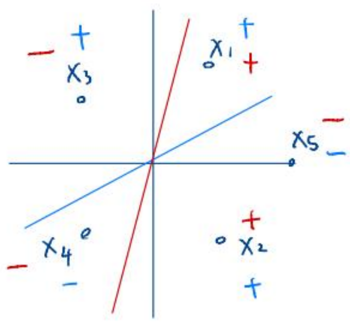
Question assigned to the following page: 9

9. In Lecture 16, we talked about the probability to fit data perfectly when the labels are random. For instance, page 6 of Lecture 16 shows that the probability of fitting the data perfectly with decision stumps is $(2N)/2^N$. Consider five points in $\mathbb{R}^2$ as input vectors $\mathbf{x}_1 = (+1, +1)$, $\mathbf{x}_2 = (+1, -1)$, $\mathbf{x}_3 = (-1, +1)$, $\mathbf{x}_4 = (-1, -1)$, $\mathbf{x}_5 = (2, 0)$, and a 2D perceptron model that minimizes $E_{\text{in}}(\mathbf{w})$ to the lowest possible value. One way to measure the power of the model is to consider five random labels y_1, y_2, y_3, y_4, y_5, each in ± 1 and generated by i.i.d. fair coin flips, and then compute

$$\mathbb{E}_{y_1,y_2,y_3,y_4,y_5}\left(\min_{\mathbf{w}\in\mathbb{R}^{2+1}} E_{\text{in}}(\mathbf{w})\right)$$

in terms of the 0/1 error. For a perfect fitting, $\min_{\mathbf{w}} E_{\text{in}}(\mathbf{w})$ will be 0; for a less perfect fitting (when the data is not linearly separable), $\min_{\mathbf{w}} E_{\text{in}}(\mathbf{w})$ will be some non-zero value. The expectation above averages over all 32 possible combinations of y_1, y_2, y_3, y_4, y_5. What is the value of the expectation? Prove your result.

(Note: It can be shown that 1 minus twice the expected value above is the same as the so-called empirical Rademacher complexity of 2D perceptrons. Rademacher complexity, similar to the VC dimension, is another tool to measure the complexity of a hypothesis set. If a hypothesis set shatters some data points, zero E_{in} can always be achieved and thus Rademacher complexity is 1; if a hypothesis set cannot shatter some data points, Rademacher complexity provides a soft measure of how "perfect" the hypothesis set is.)



x_1	-1	1	1	1	-1	1	-1	-1	1
x_2	-1	-1	1	1	1	1	1	-1	-1
x_3	1	-1	-1	-1	1	-1	1	1	-1
x_4	1	1	-1	1	-1	-1	-1	-1	1
x_5	1	1	1	-1	1	-1	-1	1	-1

error occurs at perceptron has no ability to shatter all five points.

$\Rightarrow$ when $x_1 \to x_3 \to x_4 \to x_2 \to x_5$ have more than 2 sign changes.

$\downarrow$

there exists 10 possible combination such that error = 1

$E_{y_1 \sim y_5}(\min E_{in}(w)) = \frac{10}{32}$

Question assigned to the following page: 10

10. (20 points, *) Select the best λ^* as

$$\underset{\log_{10}\lambda \in \{-6,-4,-2,0,2\}}{\operatorname{argmin}} E_{\text{in}}(\mathbf{w}_\lambda).$$

Break the tie, if any, by selecting the largest λ. What is $\log_{10}(\lambda^*)$?

```
lamda-6
Accuracy = 96% (192/200) (classification)
5000.0
lamda-4
Accuracy = 92% (184/200) (classification)
50.0
lamda-2
Accuracy = 91% (182/200) (classification)
0.5
lamda0
Accuracy = 87.5% (175/200) (classification)
0.005
lamda2
Accuracy = 80.5% (161/200) (classification)
Best Ein: 0.04
log10(λ*): -6
```

$\arg\min E_{in}(w_\lambda)$ happens at $\lambda^* = -6$ #

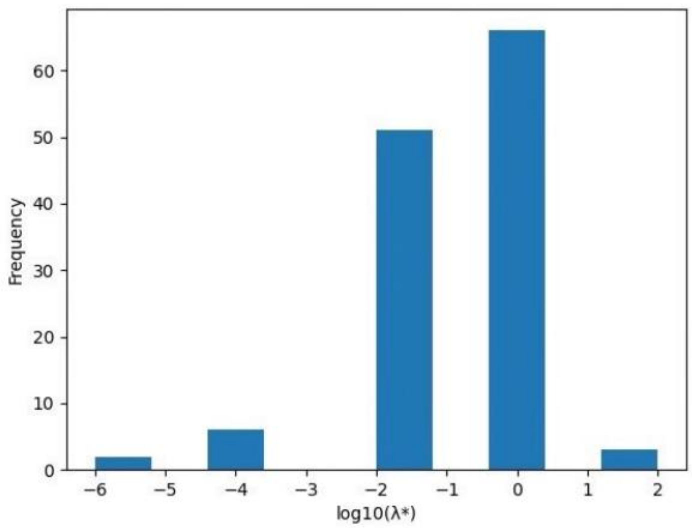
$\log\lambda \in \{-6, -4, 0, 2, -2\}$

Question assigned to the following page: 11

11. (20 points, *) Now randomly split the given training examples in $\mathcal{D}$ to two sets: 120 examples as $\mathcal{D}_{\text{train}}$ and 80 as $\mathcal{D}_{\text{val}}$. Run $\mathcal{A}_\lambda$ on *only* $\mathcal{D}_{\text{train}}$ to get $\mathbf{w}_\lambda^-$ (the weight vector within the g^- returned), and validate $\mathbf{w}_\lambda^-$ with $\mathcal{D}_{\text{val}}$ to get $E_{\text{val}}(\mathbf{w}_\lambda^-)$. Select the best λ^* as

$$\operatorname*{argmin}_{\log_{10}\lambda\in\{-6,-4,-2,0,2\}} E_{\text{val}}(\mathbf{w}_\lambda^-).$$

Break the tie, if any, by selecting the largest λ. Repeat the experiment above for 128 times, each with a different random split. Plot a histogram on the distribution of $\log_{10}(\lambda^*)$ selected from the 128 experiments.

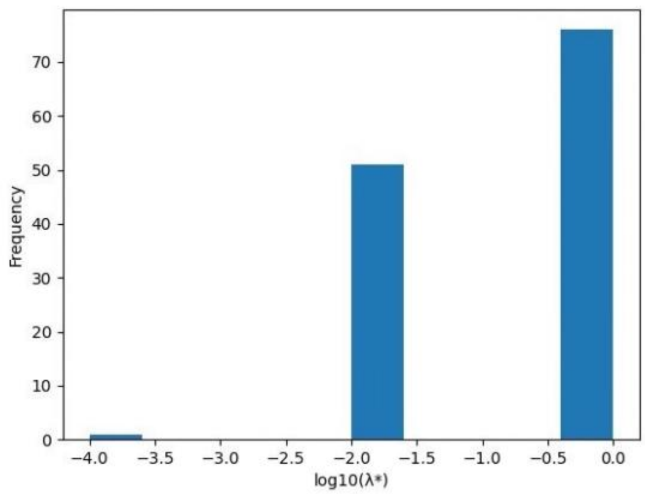


Question assigned to the following page: 12

12. (20 points, *) Now randomly split the given training examples in $\mathcal{D}$ to five folds, 40 being fold 1, another 40 being fold 2, and so on. Select the best λ^* as

$$\underset{\log_{10}\lambda\in\{-6,-4,-2,0,2\}}{\operatorname{argmin}} E_{\text{CV}}(\mathcal{A}_\lambda).$$

Break the tie, if any, by selecting the largest λ. Repeat the experiment above for 128 times, each with a different random split. Plot a histogram on the distribution of $\log_{10}(\lambda^*)$ selected from the 128 experiments. Compare your result with the $\log_{10}(\lambda^*)$ selected for the two problems above. Describe your findings.



If we seperate 200 datas into 5 pieces for validation and training, $\min(E_{cv})$ happens when $\log_{10}(\lambda) = 0$. $\log_{10}(\lambda) = -2$.

which is similiar to problem 11, when we seperate data into 2 segments, 120 for training and 80 for testing, $\min(E_{in})$ happens when $\log_{10}(\lambda) = 0, -2$, seldom did $\min(E_{in})$ happens when $\log_{10}(\lambda) = -6, -4, 2$

But different from problem 10 (without seperate data into D_{train} and D_{val}), which $\min(E_{in})$ happens at $\log_{10}(\lambda) = -6$.

Question assigned to the following page: 13

13. (Bonus 20 points) Dr. Regularize recently learned regularization and thought that its basic goal is to restrict the length of the weight vector $\mathbf{w}$ to be less than $\sqrt{C}$. Ze then designed a "new" regularization algorithm—simply performing linear regression first to get some $\mathbf{w}_{\text{LIN}}$, and then get $\mathbf{w}_C = \frac{\mathbf{w}_{\text{LIN}}}{\|\mathbf{w}_{\text{LIN}}\|} \cdot \sqrt{C}$. Then, $\mathbf{w}_C$ would be of length $\sqrt{C}$ only. Ze then asks chatGPT whether this is equivalent to the C-constrained regularization (and hence equivalent to λ-penalized L2 regularization) that ze learned in class, and got the following answer.

if $X^T X = \alpha I = \begin{bmatrix} \alpha & \\ & \ddots \end{bmatrix}$, $X = \begin{bmatrix} -\sqrt{\alpha} & \\ & \ddots \end{bmatrix}$

$W_{LIN} = (X^T X)^{-1} X^T y$

$\min_w E_{in}(w) = \frac{1}{N} \sum_{n=1}^{N} (w^T x_n - y_n)$, $\sum_{q=0}^{Q} w_q^2 \le C$

$\rightarrow W_{LIN} = \frac{1}{\alpha} X^T y = \frac{1}{\alpha} X y = \frac{1}{\sqrt{\alpha}} y$, $W_c = \sqrt{\frac{C}{N}}\, y \rightarrow W_c \parallel y$

in constrained regularization , w is updated until $W_{REG} \parallel -\nabla E_{in}$

$W_{REG}^T x_n = y^T$

$\rightarrow W_{REG}^T \alpha I = y^T$, $W_{REG} = \frac{1}{\sqrt{\alpha}} y$, $w^T w = C \Rightarrow W_{REG} = \sqrt{\frac{C}{N}}\, y$

when $X^T X = \alpha I$, $W_c = W_{REG} = \sqrt{\frac{C}{N}}\, y$ *