Homework #4

RELEASE DATE: 11/13/2020 FOR PROBLEMS 1–15 FIRST

RELEASE DATE: 11/16/2020 FOR PROBLEMS 16–20

RED BUG FIX: 11/16/2020 23:50

BLUE BUG FIX: 11/25/2020 06:00

GREEN BUG FIX: 11/29/2020 07:15

DUE DATE: 12/04 (THREE WEEKS, YEAH!!), BEFORE 13:00 on Gradescope

RANGE: LECTURES 13-16 + ANY EARLIER KNOWLEDGE

QUESTIONS ARE WELCOMED ON THE NTU COOL FORUM.

We will instruct you on how to use Gradescope to upload your choices and your scanned/printed solutions. For problems marked with (*), please follow the guidelines on the course website and upload your source code to Gradescope as well. You are encouraged to (but not required to) include a README to help the TAs check your source code. Any programming language/platform is allowed.

Any form of cheating, lying, or plagiarism will not be tolerated. Students can get zero scores and/or fail the class and/or be kicked out of school and/or receive other punishments for those kinds of misconducts.

Discussions on course materials and homework solutions are encouraged. But you should write the final solutions alone and understand them fully. Books, notes, and Internet resources can be consulted, but not copied from.

Since everyone needs to write the final solutions alone, there is absolutely no need to lend your homework solutions and/or source codes to your classmates at any time. In order to maximize the level of fairness in this class, lending and borrowing homework solutions are both regarded as dishonest behaviors and will be punished according to the honesty policy.

You should write your solutions in English or Chinese with the common math notations introduced in class or in the problems. We do not accept solutions written in any other languages.

This homework set comes with 400 points. For each problem, there is one correct choice. For most of the problems, if you choose the correct answer, you get 20 points; if you choose an incorrect answer, you get −10 points. That is, the expected value of random guessing is −20 per problem, and if you can eliminate two of the choices accurately, the expected value of random guessing on the remaining three choices would be 0 per problem. For other problems, the TAs will check your solution in terms of the written explanations and/or code. The solution will be given points between $[-20, 20]$ based on how logical your solution is.

Deterministic Noise

1. (Lecture 13) Consider the target function $f(x) = e^x$. When x is uniformly sampled from $[0, 2]$, and we use all linear hypotheses $h(x) = w \cdot x$ to approximate the target function with respect to the squared error, what is the magnitude of deterministic noise for each x? Choose the correct answer; explain your answer.

[a] $|e^x|$

[b] $|e^x - (\frac{3+e^2}{8})x|$

[c] $|e^x - (\frac{3+3e^2}{8})x|$

[d] $|e^x - \frac{e^2}{8}x|$

[e] $|e^x - \frac{3e^2}{8}x|$

(*Hint: If you want to take page 17 of Lecture 13 for inspiration, please note that the answer on page 17 is* ***not exact****. Here, however, we are asking you for an exact answer.*)

Learning Curve

2. (Lecture 13) Learning curves are important for us to understand the behavior of learning algorithms. The learning curves that we have plotted in lecture 13 come from polynomial regression with squared error, and we see that the expected E_{in} curve is always below the expected E_{out} curve. Next, we think about whether this behavior is also true in general. Consider the 0/1 error, an arbitrary non-empty hypothesis set $\mathcal{H}$, and a learning algorithm $\mathcal{A}$ that returns one $h \in \mathcal{H}$ with the minimum E_{in} on any non-empty data set $\mathcal{D}$. That is,

$$\mathcal{A}(\mathcal{D}) = \underset{h \in \mathcal{H}}{\operatorname{argmin}} E_{\text{in}}(h).$$

Assume that each example in $\mathcal{D}$ is generated i.i.d. from a distribution $\mathcal{P}$, and define $E_{\text{out}}(h)$ with respect to the distribution. How many of the following statements are ***always false***?

- $\mathbb{E}_{\mathcal{D}}[E_{\text{in}}(\mathcal{A}(\mathcal{D}))] < \mathbb{E}_{\mathcal{D}}[E_{\text{out}}(\mathcal{A}(\mathcal{D}))]$
- $\mathbb{E}_{\mathcal{D}}[E_{\text{in}}(\mathcal{A}(\mathcal{D}))] = \mathbb{E}_{\mathcal{D}}[E_{\text{out}}(\mathcal{A}(\mathcal{D}))]$
- $\mathbb{E}_{\mathcal{D}}[E_{\text{in}}(\mathcal{A}(\mathcal{D}))] > \mathbb{E}_{\mathcal{D}}[E_{\text{out}}(\mathcal{A}(\mathcal{D}))]$

Choose the correct answer; explain your answer.

[a] 0

[b] 1

[c] 2

[d] 3

[e] 1126 (seriously?)

(*Hint: Think about the optimal hypothesis* $h^* = \operatorname{argmin}_{h \in \mathcal{H}} E_{\text{out}}(h)$.)

Noisy Virtual Examples

3. (Lecture 13) 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 the noise vector $\boldsymbol{\epsilon} \in \mathbb{R}^{d+1}$ is generated i.i.d. from a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \sigma^2 \cdot \mathrm{I}_{d+1})$. Here $\mathbf{0} \in \mathbb{R}^{d+1}$ denotes the all-zero vector and I_{d+1} is an identity matrix of size $d+1$.

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 $\mathbb{E}(\mathrm{X}_h^T \mathrm{X}_h)$, where the expectation is taken over the (Gaussian)-noise generating process above? Choose the correct answer; explain your answer.

[a] $\mathrm{X}^T\mathrm{X} + \sigma^2\mathrm{I}_{d+1}$

[b] $\mathrm{X}^T\mathrm{X} + 2\sigma^2\mathrm{I}_{d+1}$

[c] $2\mathrm{X}^T\mathrm{X} + \sigma^2\mathrm{I}_{d+1}$

[d] $2\mathrm{X}^T\mathrm{X} + N\sigma^2\mathrm{I}_{d+1}$

[e] $2\mathrm{X}^T\mathrm{X} + 2N\sigma^2\mathrm{I}_{d+1}$

(*Note: The choices here "hint" you that the expected value is related to the matrix being inverted for regularized linear regression—see page 10 of Lecture 14. That is, data hinting "by noise" is closely related to regularization. If* $\mathbf{x}$ *contains the pixels of an image, the virtual example is a Gaussian-noise-contaminated image with the same label, e.g.* `https://en.wikipedia.org/wiki/Gaussian_noise`. *Adding such noise is a very common technique to generate virtual examples for images.*)

4. (Lecture 13) Following the previous problem, when training the linear regression model, we also need to calculate $\mathrm{X}^T\mathbf{y}$. Define the hinted label vector $\mathbf{y}_h = \begin{bmatrix}\mathbf{y}\\ \mathbf{y}\end{bmatrix}$. What is the expected value $\mathbb{E}(\mathrm{X}_h^T\mathbf{y}_h)$, where the expectation is taken over the (Gaussian)-noise generating process above? Choose the correct answer; explain your answer.

[a] $2N\mathrm{X}^T\mathbf{y}$

[b] $N\mathrm{X}^T\mathbf{y}$

[c] $\mathbf{0}$

[d] $\mathrm{X}^T\mathbf{y}$

[e] $2\mathrm{X}^T\mathbf{y}$

Regularization

5. (Lecture 14) Consider the matrix of input vectors as X (as defined in Lecture 9), and assume $\mathrm{X}^T\mathrm{X}$ to be invertible. That is, $\mathrm{X}^T\mathrm{X}$ must be symmetric positive definite and can be decomposed to $\mathrm{Q}\Gamma\mathrm{Q}^T$, where Q is an orthogonal matrix ($\mathrm{Q}^T\mathrm{Q} = \mathrm{Q}\mathrm{Q}^T = \mathrm{I}_{d+1}$) and Γ is a diagonal matrix that contains the eigenvalues $\gamma_0, \gamma_1, \ldots, \gamma_d$ of $\mathrm{X}^T\mathrm{X}$. Note that the eigenvalues must be positive.

Now, consider a feature transform $\mathbf{\Phi}(\mathbf{x}) = \mathrm{Q}^T\mathbf{x}$. The feature transform "rotates" the original $\mathbf{x}$. After transforming each $\mathbf{x}_n$ to $\mathbf{z}_n = \mathbf{\Phi}(\mathbf{x}_n)$, denote the new matrix of transformed input vectors as Z. That is, $\mathrm{Z} = \mathrm{XQ}$. Then, apply regularized linear regression in the $\mathcal{Z}$-space (see Lecture 12). That is, solve

$$\min_{\mathbf{w}\in\mathbb{R}^{d+1}} \frac{1}{N}\|\mathrm{Z}\mathbf{w} - \mathbf{y}\|^2 + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w}.$$

Denote the optimal solution when $\lambda = 0$ as $\mathbf{v}$ (i.e. $\mathbf{w}_{\text{lin}}$), and the optimal solution when $\lambda > 0$ as $\mathbf{u}$ (i.e., $\mathbf{w}_{\text{reg}}$). What is the ratio u_i/v_i? Choose the correct answer; explain your answer.

[a] $\frac{1}{1+\lambda}$

[b] $\frac{1}{1+\lambda^2}$

[c] $\frac{\gamma_i^2}{\gamma_i^2+\lambda}$

[d] $\frac{\gamma_i}{\gamma_i+\lambda}$

[e] $\frac{\gamma_i}{\gamma_i+\lambda^2}$

(*Note: All the choices are of value* < 1 *if* $\lambda > 0$. *This is the behavior of weight "decay"—*$\mathbf{w}_{reg}$ *is shorter than* $\mathbf{w}_{lin}$. *That is why the L2-regularizer is also called the weight-decay regularizer*)

6. (Lecture 14) 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 $\lambda > 0$. What is the relationship? Choose the correct answer; explain your answer.

[a] $C = \left(\dfrac{\sum_{n=1}^{N} x_n y_n}{\sum_{n=1}^{N} x_n^2 + \lambda} \right)^2$

[b] $C = \left(\dfrac{\sum_{n=1}^{N} y_n^2}{\sum_{n=1}^{N} x_n^2 + \lambda} \right)^2$

[c] $C = \left(\dfrac{\sum_{n=1}^{N} x_n^2 y_n^2}{\sum_{n=1}^{N} x_n^2 + \lambda} \right)^2$

[d] $C = \left(\dfrac{\sum_{n=1}^{N} x_n y_n}{\sum_{n=1}^{N} y_n^2 + \lambda} \right)^2$

[e] $C = \left(\dfrac{\sum_{n=1}^{N} x_n^2}{\sum_{n=1}^{N} y_n^2 + \lambda} \right)^2$

(*Note: All the choices hint you that a smaller λ corresponds to a bigger C.*)

7. (Lecture 14) Additive smoothing (`https://en.wikipedia.org/wiki/Additive_smoothing`) is a simple yet useful technique in estimating discrete probabilities. Consider the technique for estimating the head probability of a coin. Let $y_1, y_2, \ldots, y_N$ denotes the flip results from a coin, with $y_n = 1$ meaning a head and $y_n = 0$ meaning a tail. Additive smoothing adds $2K$ "virtual flips", with K of them being head and the other K being tail. Then, the head probability is estimated by

$$\frac{(\sum_{n=1}^{N} y_n) + K}{N + 2K}$$

The estimate can be viewed as the optimal solution of

$$\min_{y \in \mathbb{R}} \frac{1}{N} \sum_{n=1}^{N} (y - y_n)^2 + \frac{2K}{N} \Omega(y),$$

where $\Omega(y)$ is a "regularizer" to this estimation problem. What is $\Omega(y)$? Choose the correct answer; explain your answer.

[a] $(y + 1)^2$
[b] $(y + 0.5)^2$
[c] y^2
[d] $(y - 0.5)^2$
[e] $(y - 1)^2$

8. (Lecture 14) On page 12 of Lecture 14, we mentioned that the ranges of features may affect regularization. One common technique to align the ranges of features is to consider a "scaling" transformation. Define $\mathbf{\Phi}(\mathbf{x}) = \Gamma^{-1}\mathbf{x}$, where Γ is a diagonal matrix with positive diagonal values $\gamma_0, \gamma_1, \ldots, \gamma_d$. Then, conducting L2-regularized linear regression in the $\mathcal{Z}$-space.

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

is equivalent to regularized linear regression in the $\mathcal{X}$-space

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

with a different regularizer $\Omega(\mathbf{w})$. What is $\Omega(\mathbf{w})$? Choose the correct answer; explain your answer.

[a] $\mathbf{w}^T \Gamma \mathbf{w}$
[b] $\mathbf{w}^T \Gamma^2 \mathbf{w}$
[c] $\mathbf{w}^T \mathbf{w}$
[d] $\mathbf{w}^T \Gamma^{-2} \mathbf{w}$
[e] $\mathbf{w}^T \Gamma^{-1} \mathbf{w}$

9. (Lecture 13/14) In the previous problem, regardless of which regularizer you choose, the optimization problem is of the form

$$\min_{\mathbf{w}\in\mathbb{R}^{d+1}} \frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n - y_n)^2 + \frac{\lambda}{N}\sum_{i=0}^{d}\beta_i w_i^2$$

with positive constants β_i. We will call the problem "scaled regularization."

Now, consider linear regression with virtual examples. That is, we add K virtual examples $(\tilde{\mathbf{x}}_1, \tilde{y}_1), (\tilde{\mathbf{x}}_2, \tilde{y}_2) \dots (\tilde{\mathbf{x}}_K, \tilde{y}_K)$ to the training data set, and solve

$$\min_{\mathbf{w}\in\mathbb{R}^{d+1}} \frac{1}{N+K}\left(\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n - y_n)^2 + \sum_{k=1}^{K}(\mathbf{w}^T\tilde{\mathbf{x}}_k - \tilde{y}_k)^2\right).$$

We will show that using some "special" virtual examples, which were claimed to be a possible way to combat overfitting in Lecture 13, is related to regularization, another possible way to combat overfitting discussed in Lecture 14.

Let $\tilde{\mathrm{X}} = [\tilde{\mathbf{x}}_1\tilde{\mathbf{x}}_2 \dots \tilde{\mathbf{x}}_K]^T$, $\tilde{\mathbf{y}} = [\tilde{y}_1, \tilde{y}_2 \dots \tilde{y}_K]^T$, and B be a diagonal matrix that contains $\beta_0, \beta_1, \beta_2, \dots, \beta_d$ in its diagonals. Set $K = d+1$, for what $\tilde{\mathrm{X}}$ and $\tilde{\mathbf{y}}$ will the optimal solution of this linear regression be the same as the optimal solution of the scaled regularization problem above? Choose the correct answer; explain your answer.

[a] $\tilde{\mathrm{X}} = \lambda \mathrm{I}_K, \tilde{\mathbf{y}} = \mathbf{0}$
[b] $\tilde{\mathrm{X}} = \sqrt{\lambda}\cdot\sqrt{\mathrm{B}}, \tilde{\mathbf{y}} = \mathbf{0}$
[c] $\tilde{\mathrm{X}} = \sqrt{\lambda}\cdot\mathrm{B}, \tilde{\mathbf{y}} = \mathbf{0}$
[d] $\tilde{\mathrm{X}} = \mathrm{B}, \tilde{\mathbf{y}} = \sqrt{\lambda}\mathbf{1}$
[e] $\tilde{\mathrm{X}} = \mathrm{B}, \tilde{\mathbf{y}} = \lambda\mathbf{1}$

(*Note: Both Problem 3 and this problem show that data hinting is closely related to regularization.*)

Leave-one-out

10. (Lecture 15) Consider a binary classification algorithm $\mathcal{A}_{\text{majority}}$, which returns a constant classifier that always predicts the majority class (i.e., the class with more instances in the data set that it sees). As you can imagine, the returned classifier is the best-E_{in} one among all constant classifiers. For a binary classification data set with N positive examples and N negative examples, what is $E_{\text{loocv}}(\mathcal{A}_{\text{majority}})$? Choose the correct answer; explain your answer.

[a] 0
[b] $1/N$
[c] $1/2$
[d] $(N-1)/N$
[e] 1

11. (Lecture 15) Consider the decision stump model and the data generation process mentioned in Problem 16 of Homework 2, and use the generation process to generate a data set of N examples (instead of 2). If the data set contains at least two positive examples and at least two negative examples, which of the following is the tightest upper bound on the leave-one-out error of the decision stump model? Choose the correct answer; explain your answer.

[a] 0
[b] $1/N$
[c] $2/N$
[d] $1/2$
[e] 1

12. (Lecture 15) You are given three data points: $(x_1, y_1) = (3, 0), (x_2, y_2) = (\rho, 2), (x_3, y_3) = (-3, 0)$ with $\rho \geq 0$, and a choice between two models: constant (all hypotheses are of the form $h(x) = w_0$) and linear (all hypotheses are of the form $h(x) = w_0 + w_1 x$). For which value of ρ would the two models be tied using leave-one-out cross-validation with the squared error measure? Choose the correct answer; explain your answer.

[a] $\sqrt{4 + 9\sqrt{6}}$
[b] $\sqrt{16 + 81\sqrt{6}}$
[c] $\sqrt{9 + 4\sqrt{6}}$
[d] $\sqrt{36 + 16\sqrt{6}}$
[e] $\sqrt{81 + 36\sqrt{6}}$

13. (Lecture 15) Consider a probability distribution $\mathcal{P}(\mathbf{x}, y)$ that can be used to generate examples $(\mathbf{x}, y)$, and suppose we generate K i.i.d. examples from the distribution as validation examples, and store them in $\mathcal{D}_{\text{val}}$. For any fixed hypothesis h, we can show that

$$\underset{\mathcal{D}_{\text{val}} \sim \mathcal{P}^K}{\text{Variance}}\left[E_{\text{val}}(h)\right] = \square \cdot \underset{(\mathbf{x},y)\sim\mathcal{P}}{\text{Variance}}\left[\text{err}(h(\mathbf{x}), y)\right].$$

Which of the following is $\square$? Choose the correct answer; explain your answer.

[a] K
[b] 1
[c] $\frac{1}{\sqrt{K}}$
[d] $\frac{1}{K}$
[e] $\frac{1}{K^2}$

Learning Principles

14. (Lecture 16) 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 4 vertices of a rectangle in $\mathbb{R}^2$ as input vectors $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, $\mathbf{x}_4$, 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 four random labels y_1, y_2, y_3, y_4, each in ± 1 and generated by i.i.d. fair coin flips, and then compute

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

For a perfect fitting, $\min E_{\text{in}}(\mathbf{w})$ will be 0; for a less perfect fitting (when the data is not linearly separable), $\min E_{\text{in}}(\mathbf{w})$ will be some non-zero value. The expectation above averages over all 16 possible combinations of y_1, y_2, y_3, y_4. What is the value of the expectation? Choose the correct answer; explain your answer.

[a] 0/64
[b] 1/64
[c] 2/64
[d] 4/64
[e] 8/64

(*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.*)

15. (Lecture 16) Consider a binary classifier g such that

$$\begin{aligned} P(g(\mathbf{x}) = -1 | y = +1) &= \epsilon_+ \\ P(g(\mathbf{x}) = +1 | y = -1) &= \epsilon_-. \end{aligned}$$

When deploying the classifier to a test distribution of $P(y = +1) = P(y = -1) = 1/2$, we get $E_{\text{out}}(g) = \frac{1}{2}\epsilon_+ + \frac{1}{2}\epsilon_-$. Now, if we deploy the classifier to another test distribution $P(y = +1) = p$ instead of $1/2$, the $E_{\text{out}}(g)$ under this test distribution will then change to a different value. Note that under this test distribution, a constant classifier g_c that always predicts $+1$ will suffer from $E_{\text{out}}(g_c) = (1 - p)$ as it errors on all the negative examples. At what p, if its value is between $[0, 1]$, will our binary classifier g be as good as (or as bad as) the constant classifier g_c in terms of E_{out}? Choose the correct answer; explain your answer.

[a] $p = \frac{1-\epsilon_-}{\epsilon_+ - \epsilon_- + 1}$

[b] $p = \frac{1-\epsilon_-}{\epsilon_- - \epsilon_+ + 1}$

[c] $p = \frac{1-\epsilon_+}{\epsilon_- - \epsilon_+ + 1}$

[d] $p = \frac{1-\epsilon_+}{\epsilon_+ - \epsilon_- + 1}$

[e] $p = \frac{1}{2}$

Experiments with Regularized Logistic Regression

Consider L2-regularized logistic regression with second-order polynomial transformation.

$$\mathbf{w}_\lambda = \operatorname*{argmin}_{\mathbf{w}} \frac{\lambda}{N}\|\mathbf{w}\|^2 + \frac{1}{N}\sum_{n=1}^{N} \ln(1 + \exp(-y_n \mathbf{w}^T \mathbf{\Phi}_2(\mathbf{x}_n))),$$

Here $\mathbf{\Phi}_2$ is the second-order polynomial transformation introduced in page 2 of Lecture 12 (with $Q = 2$), defined as

$$\mathbf{\Phi}_2(\mathbf{x}) = (1, x_1, x_2, \ldots, x_d, x_1^2, x_1x_2, \ldots, x_1x_d, x_2^2, x_2x_3, \ldots, x_2x_d, \ldots, x_d^2)$$

Given that $d = 6$ in the following data sets, your $\Phi_2(\mathbf{x})$ should be of 28 dimensions (including the constant dimension).

Next, we will take the following file as our training data set $\mathcal{D}$:

`http://www.csie.ntu.edu.tw/~htlin/course/ml20fall/hw4/hw4_train.dat`

and the following file as our test data set for evaluating E_{out}:

`http://www.csie.ntu.edu.tw/~htlin/course/ml20fall/hw4/hw4_test.dat`

We call the algorithm for solving the problem above as $\mathcal{A}_\lambda$. The problem guides you to use LIBLINEAR (`https://www.csie.ntu.edu.tw/~cjlin/liblinear/`), a machine learning packaged developed in our university, to solve this problem. In addition to using the default options, what you need to do when running LIBLINEAR are

- set option `-s 0`, which corresponds to solving regularized logistic regression
- set option `-c C`, with a parameter value of `C` calculated from the λ that you want to use; read `README` of the software package to figure out how `C` and your λ should relate to each other
- set option `-e 0.000001`, which corresponds to getting a solution that is really really close to the optimal solution

LIBLINEAR can be called from the command line or from major programming languages like python. If you run LIBLINEAR in the command line, please include screenshots of the commands/results; if you run LIBLINEAR from any programming language, please include screenshots of your code.

We will consider the data set as a *binary classification problem* and take the "regression for classification" approach with regularized logistic regression (see Page 6 of Lecture 10). So please evaluate all errors below with the 0/1 error.

16. Select the best λ^* *in a cheating manner* as

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

Break the tie, if any, by selecting the largest λ. What is $\log_{10}(\lambda^*)$? Choose the closest answer; provide your command/code.

[a] -4

[b] -2

[c] 0

[d] 2

[e] 4

17. Select the best λ^* as

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

Break the tie, if any, by selecting the largest λ. What is $\log_{10}(\lambda^*)$? Choose the closest answer; provide your command/code.

[a] -4

[b] -2

[c] 0

[d] 2

[e] 4

18. Now split the given training examples in $\mathcal{D}$ to two sets: the first 120 examples as $\mathcal{D}_{\text{train}}$ and 80 as $\mathcal{D}_{\text{val}}$. (*Ideally, you should randomly do the* 120/80 *split. Because the given examples are already randomly permuted, however, we would use a fixed split for the purpose of this problem*). 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

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

Break the tie, if any, by selecting the largest λ. Then, estimate $E_{\text{out}}(\mathbf{w}_{\lambda^*}^-)$ with the test set. What is the value of $E_{\text{out}}(\mathbf{w}_{\lambda^*}^-)$? Choose the closest answer; provide your command/code.

[a] 0.10

[b] 0.11

[c] 0.12

[d] 0.13

[e] 0.14

19. For the λ_* selected in the previous problem, compute $\mathbf{w}_{\lambda^*}$ by running $\mathcal{A}_{\lambda^*}$ with the full training set $\mathcal{D}$. Then, estimate $E_{\text{out}}(\mathbf{w}_{\lambda^*})$ with the test set. What is the value of $E_{\text{out}}(\mathbf{w}_{\lambda^*})$? Choose the closest answer; provide your command/code.

[a] 0.10

[b] 0.11

[c] 0.12

[d] 0.13

[e] 0.14

20. Now split the given training examples in $\mathcal{D}$ to five folds, the first 40 being fold 1, the next 40 being fold 2, and so on. Again, we take a fixed split because the given examples are already randomly permuted. Select the best λ^* as

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

Break the tie, if any, by selecting the largest λ. What is the value of $E_{\text{cv}}(\mathcal{A}_{\lambda^*})$ Choose the closest answer; provide your command/code.

[a] 0.10

[b] 0.11

[c] 0.12

[d] 0.13

[e] 0.14