ECE 661 Spring 2025 Midterm Exam

Introduction to Machine Learning for Engineers

Prof. Gauri Joshi and Prof. Carlee Joe-Wong

Wednesday, February 26, 2025

9:00am - 10:50am PT/12:00pm - 1:50pm ET/7:00pm - 8:50pm CAT

Name ____________________________ **Andrew ID** ________________________

Instructions

a. If a problem asks you which of its choices is TRUE, you should treat choices that may be either true or false as FALSE.

b. **Unless otherwise stated, only one option is correct in each multiple-choice question.** No partial credit will be given for multiple-choice questions that have only one possible answer or for true/false questions.

c. For descriptive questions, make sure to explain your answers and reasoning. We will give partial credit for wrong answers if portions of your reasoning are correct. Conversely, correct answers accompanied by incomplete or incorrect explanations may not receive full credit.

d. You are allowed one **physical, handwritten** US-letter or A4 sized cheat sheet (two-sided). No other notes or material (aside from blank pieces of scratch paper) may be used.

e. You may only use a pen/pencil, eraser, and scratch paper. The backside of each sheet in the exam can also be used as scratch paper. If you do not wish for us to grade your scratch work, please clearly indicate which parts of your work we should ignore.

f. Calculators are not necessary and are not permitted.

g. If you would like to ask a clarification question during the exam, raise your hand and an instructor or TA will come over. Note that we will not help you answer the questions but can give clarifications.

Problem	**Type**	**Points**
1	Honor Pledge	0
2-6	True/False	6
7-15	Multiple Choice	18
16-18	Descriptive	21
Total		45 points

Problem 1: To affirm that you did not cheat on the exam, please write out the below statement. Sign your name beneath it. **Failure to do so will be taken as a sign that you have cheated on the exam.**

I pledge my honor that I neither gave nor received unauthorized assistance on this examination.

1 True or False

Problem 2: *[1 points]* MLE (Maximum Likelihood Estimation) and MAP (Maximum a Posteriori) estimation yield the same result when the prior distribution in MAP is uniform.

◯ True

◯ False

Solution: True because in the case that the prior distribution in MAP is strictly uniform, the P(θ) is uniform across all θ.

Problem 3: *[1 points]* Suppose a dataset has features x_1 and x_2 and label y. Both Naive Bayes and Logistic Regression classifiers assume conditional independence of the features given the label, that is, $p(x_1, x_2|y) = p(x_1|y)p(x_2|y)$.

◯ True

◯ False

Solution: False. Naive Bayes classifier assumes that the features of the data are conditionally independent given the class label. But Logistic regression does not assume this.

Problem 4: *[1 points]* When the training dataset has a large number of samples and has high-dimensional features, the test-time computational complexity (in terms of the numbers of training samples and features) of K-Nearest Neighbors (KNN) is smaller than that of logistic regression.

◯ True

◯ False

Solution: False. KNNs struggle with large, high-dimensional datasets because they require computation of distances to each point in the training dataset.

Problem 5: *[1 points]* The dual formulation of SVM (support vector machines) will have fewer variables to optimize on a training dataset $\mathcal{D}_1$ with 1,000 data points with 20 features each than on a dataset $\mathcal{D}_2$ with 200 data points with 100 features each.

◯ True

◯ False

Solution: False. In the dual formulation, the number of variables to optimize is dependent on the number of data points but independent of the number of features.

Problem 6: *[2 points]* Suppose you are given a linear regression problem with features x_1 and x_2, and target y. Which of the following statements are true about the bias and variance of the resulting model?

a. You decide to rescale the features to $(x_1 - a_1)/b_1$ and $(x_2 - a_2)/b_2$ for some constants $a_1, a_2, b_1, b_2 > 0$ such that the features are zero-centered and have unit standard deviation for the data points, and you find a new model that fits the rescaled data. The bias component of the expected risk of the new model with feature scaling is strictly smaller than that of the old model without feature scaling.

   ◯ True

   ◯ False

   Solution: False

b. You collect more data, keep the original features, and learn a new model. Then the variance component of expected risk for the new model is smaller or equal to the variance of the old model.

- ◯ True
- ◯ False

Solution: True

2 Multiple Choice

Problem 7: *[2 points]* Increasing the value of k used in the k-Nearest Neighbors algorithm will have which of the following impacts on the bias and variance of the model?

○ Bias increases, variance decreases

○ Bias increases, variance remains the same

○ Bias remains the same, variance remains the same

○ Bias decreases, variance increases

Solution: **A** - Increasing k causes the model to underfit to the data, which increases the bias, but makes it less susceptible to changes in the data, which decreases the variance.

Problem 8: *[2 points]* Suppose you wish to train a ridge regression model on a training dataset $\mathcal{D}$. However, you have limited training data, so you decide to use cross-validation to choose λ, the weight of the regularization term in the ridge regression loss function. You decide on 10 possible values of λ and decide to use 12-fold cross-validation. How many ridge regression models will you train?

○ 21

○ 241

○ 240

○ 121

Solution: D. For each of the 12 folds, we train 10 models, one for each possible value of λ, for 120 trained models. We then find the value of λ that has best average performance across the folds, and train one more ridge regression model on the entire dataset with this value of λ.

Problem 9: *[2 points]* Consider a naive Bayes classification task with feature variables X_1 and X_2, and label variable Y. Which of the following statements are true? **More than one option can be true.**

☐ Naive Bayes is a generative classifier that models the joint distribution $P(X_1, X_2, Y)$.

☐ Naive Bayes assumes conditional independence by setting $P(X_1, X_2, Y) = P(X_1)P(X_2)P(Y)$.

☐ Laplacian smoothing with parameter $\alpha = 2$ increases the number of parameters that need to be estimated in the naive Bayes model.

☐ Classifying a new data point $(X_1 = x_1, X_2 = x_2)$ involves finding the class c that maximizes $P(Y = c \mid x_1, x_2)$.

Solution:

- True. True by definition
- False. $P(X_1, X_2, Y) = P(X_1)P(X_2)P(Y)$ is a result of an assumption of unconditional independence, not **conditional** independence.
- False. Laplacian smoothing does not affect the number of parameters that need to be estimated
- True. Classifying a new data point using a naive Bayes model involves maximizing $P(X_1, X_2, Y)$, which is proportional to the conditional likelihood $P(Y|X_1, X_2)$

Problem 10: *[2 points]* Consider a classification problem with three class labels. We use $\mathbf{y}^{(n)} \in \{0,1\}^3$ to denote the one-hot encoded class label and $\mathbf{x}^{(n)} \in \mathbb{R}^d$ to denote the feature vector of each data point n in our training dataset, where $n = 1, 2, \ldots, N$. Recall that Multinomial Logistic Regression aims to minimize the cross-entropy loss function

$$\mathcal{L}(\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3) = -\sum_{n=1}^{N}\sum_{j=1}^{3} y_j^{(n)} \log\left(\frac{\exp\left(\mathbf{w}_j^T\mathbf{x}^{(n)}\right)}{\exp\left(\mathbf{w}_1^T\mathbf{x}^{(n)}\right) + \exp\left(\mathbf{w}_2^T\mathbf{x}^{(n)}\right) + \exp\left(\mathbf{w}_3^T\mathbf{x}^{(n)}\right)}\right).$$

Which of the following statements about Multinoimal Logistic Regression and its optimization are true? **More than one option can be true.**

☐ Multinomial Logistic Regression directly maximizes the fraction of correctly classified data points during training.

☐ The cross-entropy loss function $\mathcal{L}(\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3)$ is convex, meaning gradient-based optimization methods are guaranteed to find a global minimum for a properly chosen learning rate.

☐ If $\mathcal{D}$ is linearly separable, Multinomial Logistic Regression will always perfectly classify all training data points.

☐ Multinomial Logistic Regression, as described above, always produces linear decision boundaries.

Solution:

- False: Logistic Regression does NOT directly maxminize the fraction of correctly classified data points. Instead, it maximizes the log-likelihood, which is a differentiable surrogate function that leads to a probabilistic interpretation.
- True: The cost function is convex, meaning gradient-based optimization methods are guaranteed to find a global minimum.
- False: Multinomial Logistic Regression cannot guarantee perfect classification of linearly separable data, e.g., if an outlier data point lies very close to another class.
- True: Logistic Regression computes linear functions $w_j^\top X$ and applies the softmax function to get probabilities. The decision boundary lies on those points with equal probability of belonging to two different classes, which is equivalent to comparing linear functions of the features $\mathbf{x}$.

Problem 11: *[2 points]* Consider a dataset of 6 data points $d_n = (\mathbf{x}^{(n)}, y^{(n)})$ for $n = 1, 2, \ldots, 6$, where

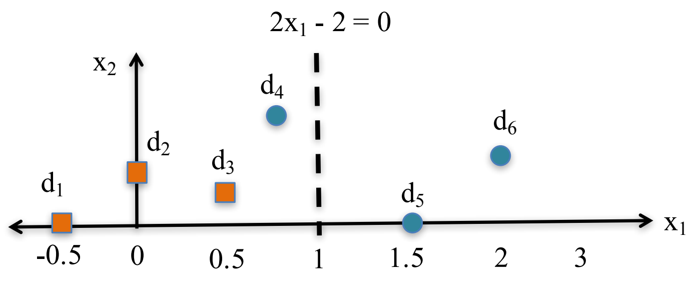

Figure 1

$\mathbf{x} = [x_1, x_2]$. Points d_1, d_2, and d_3 have the label $y = -1$, and d_4, d_5 and d_6 have the label $y = 1$. We train a linear SVM classifier and obtain the decision boundary $[w_1, w_2] = [2, 0]$ and $b = -2$ as shown in Figure 1. Which points are the support vectors?

◯ d_3, d_4, d_5

◯ d_1, d_2, d_3

◯ d_1, d_2, d_4, d_6

◯ d_1, d_2, d_6

Solution: A d_3, d_4, d_5. The margin boundaries are $x_1 = 1.5$ and $x_1 = 0.5$. These points are support vectors because they are inside the margin or on the margin

Problem 12: *[2 points]*
Which of the following statements about SVMs (support vector machines) are true? **More than one option can be true.**

☐ If the training data is linearly separable, hard-margin SVM with a linear kernel classifies all training data points correctly.

☐ If the training data is linearly separable, soft-margin SVM with a linear kernel classifies all training data points correctly.

☐ Kernel SVM can learn non-linear decision boundaries.

☐ The computational complexity of solving the dual formulation of SVM scales sublinearly with the number of features.

Solution:

- True; If the data is perfectly linearly separable, a hard-margin SVM with a linear kernel will achieve perfect classification, as it finds the maximum-margin hyperplane that separates the classes without misclassification.
- False; With soft-margin SVMs, some misclassification may occur to improve generalization.
- True: Non-linear kernels allow SVMs to project data into a higher-dimensional space, making linear separation possible.
- True: In the SVM dual formulation, the number of variables is independent of the feature dimension.

Problem 13: *[2 points]* Instead of directly minimizing the objective function $\mathcal{L}(\mathbf{w})$ in regression, we often use gradient descent methods (Batch GD, SGD, or mini-batch SGD) to iteratively optimize the objective function. Some of these methods may have an error floor convergence, that is, they do not converge exactly to the minimum of $\mathcal{L}(\mathbf{w})$. Which of the following statements is true (only one is true)?

◯ Increasing the mini-batch size, while keeping the learning rate fixed, yields faster convergence but to a higher error floor.

◯ Increasing the learning rate, while keeping the mini-batch size fixed, yields faster convergence but to a lower error floor.

◯ For small enough learning rate η, batch GD (which processes all the training data in each iteration) exactly converges to solution $\mathbf{w}^*$ that minimizes $\mathcal{L}(\mathbf{w})$.

◯ The per-iteration computational complexity of SGD and batch GD is the same.

Solution: C

a. A: Increasing mini-batch size does not affect the speed of convergence and it converges to a lower error floor

b. B: Increasing learning rate gives faster convergence but to a higher error floor

c. C: Batch GD has no noise in its gradients and thus converges exactly to the optimal solution

d. D: Batch GD has higher computational complexity

Problem 14: *[2 points]* Suppose that you wish to use gradient descent to train a linear regression model on a training dataset $\mathcal{D}$ with N data points and d features, without regularization. You obtain the gradient descent update equation

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta \mathbf{X}^T \left(\mathbf{X}\mathbf{w}_t - \mathbf{y} \right),$$

where $\eta > 0$ is the learning rate, $\mathbf{y} \in \mathbb{R}^N$ is the vector of training labels, $\mathbf{X} \in \mathbb{R}^{N \times d}$ is the matrix of training data features, and $\mathbf{w}_t \in \mathbb{R}^d$ is the value of the parameter vector after t iterations.
You randomly initialize the gradient descent algorithm at a parameter vector $\mathbf{w}_0$ and obtain the solution $\mathbf{w}_{100}$ after running 100 iterations of gradient descent. You then decide to check your work by initializing the gradient descent algorithm at $\mathbf{v}_0 \neq \mathbf{w}_0$ and obtain the solution $\mathbf{v}_{100}$ after 100 iterations. Unfortunately, $\mathbf{v}_{100} \neq \mathbf{w}_{100}$. Which of the following are possible explanations of this result? **More than one option can be correct.**

☐ You stopped the gradient descent algorithm too early, before it had converged.

☐ The gradient descent algorithms converged, but your linear regression problem has multiple optimal solutions.

☐ You calculated the gradient incorrectly.

☐ None of the above.

Solution: A is correct, since the gradient descent algorithm may take more than 100 iterations to converge. B is correct, since $\mathbf{X}^T\mathbf{X}$ may not be invertible and we may not have a unique optimal solution. C is incorrect, since the gradient update equation provided is correct.

Problem 15: *[2 points]* Let T be the number of requests ChatGPT receives per unit time, and assume that it follows a Poisson distribution with parameter $\lambda > 0$:

$$P(T = k) = \frac{\lambda^k e^{-\lambda}}{k!}, \quad k = 0, 1, 2, \ldots$$

Suppose you are given n independent observations $t_1, t_2, \ldots, t_n$ of the request counts. What is the maximum likelihood estimate λ_{MLE} of the parameter λ?

○ $\lambda_{MLE} = \frac{\sum_{i=1}^n t_i}{n}$

○ $\lambda_{MLE} = \frac{n}{\sum_{i=1}^n t_i}$

○ $\lambda_{MLE} = \sum_{i=1}^n \frac{1}{t_i}$

○ $\lambda_{MLE} = \frac{n}{\sum_{i=1}^n \frac{1}{t_i}}$

Solution: A. The likelihood function for the Poisson distribution is:

$$L(\lambda) = \prod_{i=1}^n \frac{\lambda^{t_i} e^{-\lambda}}{t_i!}$$

$$\begin{aligned}\log L(\lambda) &= \sum_{i=1}^n (t_i \log \lambda - \lambda - \log t_i!) \\ &= \left(\sum_{i=1}^n t_i \right) \log \lambda - n\lambda + \text{constant}\end{aligned}$$

Taking the derivative and setting it to zero:

$$\frac{d}{d\lambda}\log L(\lambda) = \frac{\sum_{i=1}^{n} t_i}{\lambda} - n = 0$$
$$\lambda_{MLE} = \frac{\sum_{i=1}^{n} t_i}{n}$$

Since the second derivative is negative, this is the maximizer.

3 Descriptive

Problem 16: *[7 points]* Suppose we have a training dataset with two data points $(x_1, y_1) = (1, 0)$ and $(x_2, y_2) = (0, 1)$, where x_n represents the (one-dimensional) feature and y_n its label, $n = 1, 2$. We attempt to fit a logistic regression classifier with weight $w \in \mathbb{R}$ for this data. Our classifier function is defined as $\sigma(w \cdot x)$ where $\sigma(a) = \frac{1}{1+e^{-a}}$.

a. *[2 points]* Suppose we use the loss function

$$L(w) = -\sum_{n=1}^{2}(y_n \log(\sigma(w \cdot x_n)) + (1 - y_n)\log(1 - \sigma(w \cdot x_n))).$$

Show that if we minimize this loss with respect to w on our dataset, the optimal value of w is $-\infty$.

[Hint:] You may take as given the facts that $\sigma(a) = 1 - \sigma(-a)$ and $\frac{d\sigma(a)}{a} = \sigma(a) \cdot (1 - \sigma(a))$ for any $a \in \mathbb{R}$.

b. *[2 points]* Now suppose we add an ℓ_2 regularization term to the loss function L, resulting in a new loss function

$$\xi(w) = \frac{\lambda}{2}||w||^2 - \sum_{n=1}^{2}(y_n \log(\sigma(w \cdot x_n)) + (1 - y_n)\log(1 - \sigma(w \cdot x_n))).$$

Let $\lambda = 1$ and let w^* denote the minimizer of $\xi(w)$. Is $w^* = -\infty$? Demonstrate your answer mathematically.

[Hint:] Try to write an expression that relates w^* with $\sigma(w^*)$. You do not need to explicitly solve for w^*.

c. *[2 points]* Now suppose that you decide to collect two new training data points, which turn out to be identical copies of your original training data, i.e., $(x_3, y_3) = (1, 0)$ and $(x_4, y_4) = (0, 1)$. Will the optimal (i.e., minimizing the cross-entropy loss) weight for your four training data points still be $w^* = -\infty$ as in part (a)? Briefly explain your answer.

d. *[1 points]* In 1-2 sentences, explain how increasing the coefficient λ of the ℓ_2 regularization term in part (b)'s loss function affects the magnitude of the optimal w.

Solution:

a. Loss for the dataset can be expressed as $-\log(1 - \sigma(w)) - \log(\sigma(0)) = -\log(\sigma(-w)) - \log(\frac{1}{2})$.
Taking the derivative and setting it to 0, we get
$\frac{d\xi(w)}{w} = -(\frac{1}{\sigma(-w)} \cdot \sigma(-w) \cdot (1 - \sigma(-w)) \cdot (-1)) = 0$
$\implies \sigma(w) = 0$
$\implies w = -\infty$
With this choice of loss function, our optimal weight w is unbounded.

b. The loss in this case is $\xi(w) = \frac{1}{2}||w||^2 - \log(\sigma(-w)) - \log(\frac{1}{2})$
Again, taking the derivative and setting it to 0, we get
$\frac{d\xi(w)}{w} = w - (\frac{1}{\sigma(-w)} \cdot \sigma(-w) \cdot (1 - \sigma(-w)) \cdot (-1)) = 0$
$\implies w + \sigma(w) = 0$
$\implies w = -\sigma(w)$ This expression relates w to $\sigma(w)$. Since this equation has a nonzero solution for w, w is not unbounded in this case.

c. Yes, since our new cross-entropy loss function is just two times the original one.

d. Increasing the coefficient of the ℓ_2 regularization term decreases the magnitude of the optimal w.

WRITE YOUR ANSWER BELOW

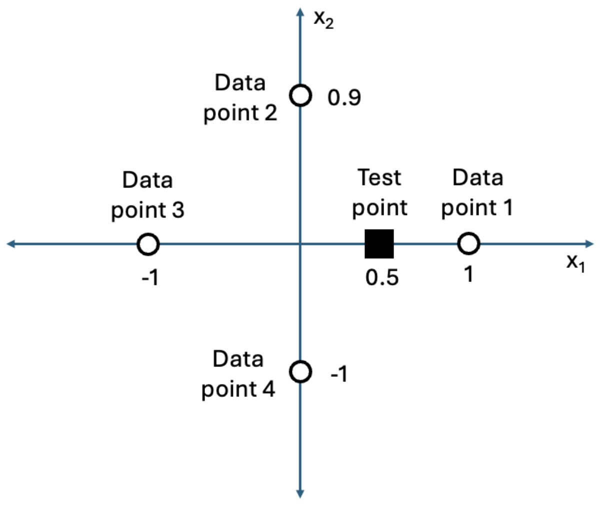

Figure 2: Training dataset (open circles) and test data point (solid square) for Problem 17.

Problem 17: *[6 points]* Suppose that you wish to use nearest neighbors to predict the class of the data point $\begin{bmatrix}0.5\\0\end{bmatrix}$, given the training dataset $\left\{\left(\begin{bmatrix}1\\0\end{bmatrix}, l_1\right), \left(\begin{bmatrix}0\\0.9\end{bmatrix}, l_2\right), \left(\begin{bmatrix}-1\\0\end{bmatrix}, l_3\right), \left(\begin{bmatrix}0\\-1\end{bmatrix}, l_4\right)\right\}$. Here l_1, l_2, l_3, l_4 are respectively the labels of training data points 1, 2, 3, and 4. They may take the values "red" or "blue." Figure 2 illustrates this dataset, where the open circles represent the training data points and the square represents the test data point.

a. *[1 points]* Suppose you decide to use 1-nearest neighbors with Euclidean distances. Find all combinations of labels $\{l_1, l_2, l_3, l_4\}$ such that the data point $\begin{bmatrix}0.5\\0\end{bmatrix}$ is classified as "red."

Solution: Since we use Euclidean distance, we see immediately that the test data point $\begin{bmatrix}0.5\\0\end{bmatrix}$ is closest to the training data point $\begin{bmatrix}1\\0\end{bmatrix}$. Thus, we must have $l_1 = \text{red}$; l_2, l_3, l_4 can take either "red" or "blue" values.

b. *[2 points]* Now suppose that you decide to use 1-nearest neighbors, but with the distance given by $d\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}, \begin{bmatrix}z_1\\z_2\end{bmatrix}\right) = \sqrt{(x_1 - z_1)^2 + \alpha (x_2 - z_2)^2}$ for some $\alpha > 0$. Given that $l_1 = l_4 = \text{red}$ and $l_2 = l_3 = \text{blue}$, find all values of α such that the test data point $\begin{bmatrix}0.5\\0\end{bmatrix}$ is labeled as "red."

Solution: We calculate the distance between $\begin{bmatrix}0.5\\0\end{bmatrix}$ and each data point:

- $\begin{bmatrix}1\\0\end{bmatrix}$: distance $\sqrt{0.5^2 + \alpha(0^2)} = 0.5$
- $\begin{bmatrix}0\\0.9\end{bmatrix}$: distance $\sqrt{0.5^2 + \alpha(0.9^2)} = \sqrt{0.25 + 0.81\alpha}$
- $\begin{bmatrix}-1\\0\end{bmatrix}$: distance $\sqrt{1.5^2 + \alpha(0^2)} = 1.5$

- $\begin{bmatrix} 0 \\ -1 \end{bmatrix}$: distance $\sqrt{0.5^2 + \alpha(1^2)} = \sqrt{0.25 + \alpha}$

In order for $\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$ to be labeled as "red," the blue data points, $\begin{bmatrix} -1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$, cannot be the closest to $\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$. However, this will not happen for any value of α, since the distance between $\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ is always 0.5, which is smaller than any of its distances to the other training data points. Thus, we may use any value of $\alpha > 0$.

c. *[3 points]* Suppose you decide to use 3-nearest neighbors, with the same distance function and labels as in part (b). Find all values of α such that the test data point $\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$ is labeled as "red."

Solution: In order for $\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$ to be labeled as "red," both red data points, $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ -1 \end{bmatrix}$, must be part of the three nearest neighbors of $\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$. From part (b), we know that $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ will always be part of this set, as it is always the training data point closest to $\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$. It therefore suffices to ensure that data point $\begin{bmatrix} 0 \\ -1 \end{bmatrix}$ is not the furthest of the four data points from $\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$. Mathematically, using the distance expressions from part (b), we have:

$$\sqrt{0.25 + \alpha} < 1.5 \text{ or}$$
$$\sqrt{0.25 + \alpha} < \sqrt{0.25 + 0.81\alpha}.$$

Since the second equation is never valid for $\alpha > 0$, we must impose $\sqrt{0.25 + \alpha} < 1.5$, or simplifying, $\alpha < 2.25 - 0.25 = 2$. Full credit was given for the expression $\alpha < 1.5^2 - 0.25$.

Problem 18: *[8 points]* Suppose that you train a kernel SVM model with kernel $k(\cdot,\cdot)$ on a training dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$. Let α_n denote the dual variable corresponding to each training data point $n = 1, 2, \ldots, N$, and suppose that the feature vectors $\mathbf{x}_n$ are d-dimensional. Recall that the SVM decision boundary, $\mathbf{w}^T\phi(\mathbf{x}) + b = 0$, where $\phi(\mathbf{z}_1)^T\phi(\mathbf{z}_2) = k(\mathbf{z}_1, \mathbf{z}_2)$, can be recovered from the dual solution by computing

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \phi(\mathbf{x}_n), \ b = y_m - \mathbf{w}^T\phi(\mathbf{x}_m)$$

for a data point m on the margin of the learned boundary.

a. *[2 points]* Suppose that you wish to use your dual SVM solution to predict the class of a given test data point $\mathbf{x}$. Given that your solution has s support vectors, how many times will you need to evaluate the kernel function $k(\cdot,\cdot)$?

Solution: To predict the class of $\mathbf{x}$, we must evaluate $\mathbf{w}^T\phi(\mathbf{x}) + b$, which we can write in terms of our dual solution as

$$\sum_{n=1}^{N} \alpha_n y_n \phi(\mathbf{x}_n)^T \phi(\mathbf{x}) + y_m - \sum_{n=1}^{N} \alpha_n y_n \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m) = \sum_{n=1}^{N} \alpha_n y_n \left(k(\mathbf{x}_n, \mathbf{x}) - k(\mathbf{x}_n, \mathbf{x}_m)\right).$$

Since $\alpha_n \neq 0$ if and only if data point n is not a support vector, we must evaluate the kernel $2s$ times. Full credit was given for the answer s times (i.e., skipping the computation of b, as it can be pre-computed).

b. *[2 points]* Unfortunately, you are now told that you need to explicitly provide the vector $\mathbf{w}$ of your SVM decision boundary, but you do not know the nonlinear transformation ϕ corresponding to your chosen kernel function $k(\cdot,\cdot)$. Thus, you instead decide to use the kernel function $k(\mathbf{z}_1, \mathbf{z}_2) = \mathbf{z}_1^T M \mathbf{z}_2$, where $M = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d)$ is the $d \times d$ diagonal matrix with entries $\lambda_1, \lambda_2, \ldots, \lambda_d$ on the diagonal and each $\lambda_i > 0$.

Given this kernel function k, find a function $\phi(\cdot)$ for which $\phi(\mathbf{z}_1)^T \phi(\mathbf{z}_2) = k(\mathbf{z}_1, \mathbf{z}_2)$.

Solution: We can take $\phi(\mathbf{x}) = \operatorname{diag}\left(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \ldots, \sqrt{\lambda_n}\right)\mathbf{x}$.

c. *[2 points]* Recall that the primal SVM problem is given by

$$\min_{\mathbf{w}, b, \xi_n} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{n=1}^{N} \xi_n \text{ s.t. } y_n\left[\mathbf{w}^T\phi(\mathbf{x}_n) + b\right] \geq 1 - \xi_n, \ \xi_n \geq 0, \ n = 1, 2, \ldots, N,$$

where $C > 0$ is a hyperparameter. Given your expression for $\phi(\mathbf{x})$ from part (b), find the number of optimization variables in the primal SVM problem. Each entry of $\mathbf{w}$ should be counted as a separate optimization variable, e.g., if $\mathbf{w}$ is 3-dimensional, it has 3 optimization variables. Your answer should be in terms of N and d.

Solution: From part (b), $\phi(\mathbf{x}_n)$ is a d-dimensional vector. Thus, so is $\mathbf{w}$ and we have $d + 1 + N$ optimization variables.

d. *[2 points]* The SVM problem that you solved in part (c) allowed you to explicitly write out the decision boundary, but it mis-classified several training data points. Suggest a way to modify the SVM problem from part (c) to reduce the number of mis-classified training data points, and explain why it would do so. Your answer should be 1-2 sentences long.

Solution: We can increase C, thus penalizing the slack variables more and encouraging fewer mis-classifications.