EECS545: Machine Learning Practice Exam (Solution)

April 3, 2024

Please do not share or post these solutions anywhere.

- You have 2.5 hours to complete this Exam.
- You may use your book and notes to complete the exam.
- However, the use of any electronic devices, including phones and laptops are strictly prohibited. You are also NOT allowed to discuss any portion of the exam with any other student while the exam is being proctored.
- The total points of the Exam is 100. Please manage your time to receive as many points as possible.
- Note that Section 3 (Short answer questions and proof questions) may take longer than the first 2 sections.
- You can use scratchpad pages given at the end of this exam sheet. If you need additional scratch pages, please ask the instructors.

There are a total of **24 questions** (Page 2 - 13). Additional pages have been provided for scratch work (Page 14 - 16). Please let us know if any pages are missing from your exam.

Sections	Pages	Questions	Points
1. Multiple choice without explanations	2 - 6	1 - 15	/60
2. True/False with explanations	7 - 8	16 - 19	/14
3. Short answer and proof questions	9 - 13	20 - 24	/26
Total	2 - 13	24	/100

Name: _____

Uniqname: _____

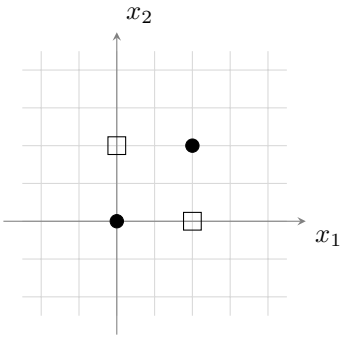
Honor Code: I attest that I have not given or received aid in this examination, and that I have done my share and taken an active part in seeing to it that others as well as myself uphold the spirit and letter of the University of Michigan College of Engineering Honor Code. Specifically for this examination, I attest that I have not used a laptop, a smartphone, any electronic device, or any online materials.

Signed: _____

Multiple choice without explanations

Write your answers [e.g. (a), (d)] in the box provided below for each question. No explanations are required. For each question, please select all answers that apply. Each question is worth 4 points, and each item within each question is worth one point. For example, if the answer to a question is (b) and (d), but you choose (a) and (d), you will get 2 points (1 point for not choosing (c) and 1 point for choosing (d)).

1. **[4 points]** Consider the following dataset with two features and two classes. Which of the following kernels would allow a logistic regression model to classify this dataset with 100% accuracy? **(Select all that apply)**



   (a) Linear kernel: $k(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle$
   (b) Quadratic kernel: $k(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle + 1)^2$
   (c) Cubic kernel: $k(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle + 1)^3$
   (d) RBF kernel: $k(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{2\sigma^2}\right)$

   Solution: (b), (c), (d).

   Just using quadratic features $x_1, x_2, x_1 x_2$ is sufficient to classify XOR.

2. **[4 points]** Which of the following sentences are correct about softmax and cross-entropy loss when performing a classification? **(Select all that apply)**

   (a) We use softmax classification with cross-entropy loss for multiclass classification problems where the classes are not mutually exclusive (i.e., inputs can belong in more than one class at once).
   (b) When we write the code for computing the Softmax function in practice, it is important to care about numerical instability as the intermediate terms could be very large due to the exponentials.
   (c) We can take the logit outputs from a previous layer and pass them directly to the Softmax function without applying a non-linear activation function to the previous outputs.
   (d) Softmax function mainly adjusts the scale of the input, so it is acceptable to train a model with cross-entropy loss by replacing the softmax output with the pre-softmax logits.

   Solution: (b), (c).

   (a): No, We generally apply softmax with cross-entropy loss over the mutually exclusive classes (i.e., class is a categorical variable)
   (b): In homeworks, we have seen the trick of subtraction of the maximum logit, and the log-sum-exp trick.
   (d): No, we can't, as cross-entropy is defined over probabilistic distribution. Also, it will end up putting negative values to the cross-entropy function.

3. **[4 points]** Suppose our model is underfitting. **Select all answers** that may alleviate this problem.

   (a) Increase the model complexity
   (b) Use regularization techniques such as L_1 or L_2 regularization
   (c) Use better feature engineering techniques to capture more relevant features
   (d) Simplify the model structure

Solution: (a), (c).

(b): we add regularization to alleviate overfitting case.
(d): simplifying would reduce the number of parameters. This would be the case for the overfitting case.

4. **[4 points]** Consider a neural network $\text{Net}(\mathbf{x}) = \mathbf{W}_2 f(\mathbf{W}_1\mathbf{x} + b_1) + b_2$ where $\mathbf{W}_1 \in \mathbb{R}^{H\times D}, \mathbf{W}_2 \in \mathbb{R}^{C\times H}$, where $f(\cdot)$ is an activation function. Which of the following sentences are correct? **(Select all that apply)**

   (a) If we introduce one additional layer and define $\text{Net}'(\mathbf{x}) = \mathbf{W}_3(\mathbf{W}_2\, g(\mathbf{W}_1\mathbf{x}+b_1)+b_2)+b_3$ where $\mathbf{W}_3 \in \mathbb{R}^{C\times C}$ and $g(\mathbf{x}) = \mathbf{x}$, it will allow $\text{Net}'(\mathbf{x})$ to represent wider set of functions compared to $\text{Net}(\mathbf{x})$ with $f(\mathbf{x}) = \text{ReLU}(\mathbf{x})$.

   (b) If we introduce one additional layer and define $\text{Net}'(\mathbf{x}) = \mathbf{W}_3(\mathbf{W}_2\, g(\mathbf{W}_1\mathbf{x}+b_1)+b_2)+b_3$ where $\mathbf{W}_3 \in \mathbb{R}^{C\times C}$ and $g(\mathbf{x}) = \mathbf{x}$, the test accuracy of Net' always be lower than $\text{Net}(\mathbf{x})$ with $f(\mathbf{x}) = \mathbf{x}$.

   (c) It is valid to choose $f(\mathbf{x}) = \text{sigmoid}(\mathbf{x})$ as an activation function in $\text{Net}(\mathbf{x})$.

   (d) Choosing $f(\mathbf{x}) = \text{sigmoid}(\mathbf{x})$ is less vulnerable to vanishing gradients compared to using $f(\mathbf{x}) = \text{ReLU}(\mathbf{x})$.

Solution: (c).

(a) and (b): There will be no nonlinearity in the network, so it can be represented as a single $\mathbf{W}$ matrix.
(d): ReLU will bring less issues compared to sigmoid because it can too easily saturated.

5. **[4 points]** Suppose a trained deep neural network achieved 99% accuracy on the training set but 20% on the testing set. Which of the following can improve the model by decreasing the variance? **(Select all that apply)**

   (a) Pre-process training data using data-augmentation techniques (e.g., random crop/scaling in image input)

   (b) Add more fully-connected layers to the model

   (c) Decrease the learning rate and train the model until convergence

   (d) Implement a Bagging ensemble method

Solution: (a), (d).

(a): Data-augmentation can be used to generalize the input data more to reduce overfitting and reduce variance.
(b): Adding more FC layers would only further increase overfitting and increase the variance.
(c): Decreased learning rate will require more epochs of training, but it will suffer similar overfitting behavior as the current model exhibits.
(d): Bagging ensemble methods can be used to average multiple models to reduce variance.

6. **[4 points]** Which of the following statements are true regarding the vanishing/exploding gradient problems in RNNs? **(Select all that apply)**

   (a) The vanishing gradient problem occurs for RNN layers closest to the output layer during back-propagation.

   (b) Using ReLU activation while initializing the RNN weights and biases to be the identity and zeros, respectively, would alleviate the vanishing/exploding gradient problems.

   (c) LSTMs can be used to mitigate the vanishing and exploding gradient problems in RNNs.

   (d) Gradient clipping can be used to clip the maximum gradient used for weight updates to avoid the exploding gradient problem.

Solution: (b), (c), (d).

(a) The vanishing gradient problem occurs for RNN layers furthest from the output layer as the gradients decay as they back-propagate from output to the first few layers in large models.
(b) Identity initialization helps with stability during training
(c) LSTMs can be used to solve the vanishing and exploding gradient problems in RNNs since they can better retain long-term dependencies in large models.
(d) Gradient clipping is used to clip gradients to a max value to avoid exploding gradients.

7. **[4 points]** Suppose we want to reduce the spatial resolution (the height $\times$ width in the feature map) of a intermediate layer of a CNN model. One way to do this would be to add a (spatial, 2D) pooling layer. However, we could also reduce the spatial resolution by changing the parameters of a convolutional layer. Which of the following would reduce the spatial resolution of the convolution output by a factor of approximately 2 for an arbitrarily shaped image? **(Select all that apply)** Here, we can assume that the image size is much larger than the convolution kernel size.

   (a) Doubling the padding.

   (b) Halving the output feature dimension (i.e., the number of channels).

   (c) Doubling the kernel size.

   (d) Doubling the stride.

Solution: (d).

(a): It will increase the dimension.
(b): Feature vector does not change the spatial dimension.
(c): Kernel size does not change the output image size unless we introduce additional padding.

8. **[4 points]** Alice and Bob are designing a CNN architecture for their project. They consider a different option for implementing CNN. Specifically, Alice prefers a single 5×5 convolution layer with 9 filters, whereas Bob prefers two 3×3 convolution layers (the first layer with 16 filters and the next layer with 9 filters). When the input data has 16 channels, which of the following statements are true regarding their architecture design choices? **(Select all that apply)** Please assume that there are no bias parameters in convolution layers.

   (a) Both architectures will have the same receptive size (i.e., an individual hidden unit in the final convolution layer at a particular location will be influenced by the same-sized patch of input pixels in the image).

   (b) Both architecture choices will have the same number of parameters to train.

   (c) Both architectures will consume the same amount of memory when training a model.

   (d) Bob's architecture will tend to achieve better performance for function approximation as it includes non-linearity between the two layers with a wider range of filter combinations from the two layers.

Solution: (a), (b), (d).

(a) True. Both architectures will have a receptive size of 5×5.
(b) True. Both architectures are having the same number of parameters.
Tom: $5 \times 5 \times 16 \times 9 = 9 \times 16 \times 25$. John: $3 \times 3 \times 16 \times 16 + 3 \times 3 \times 16 \times 9 = 9 \times 16 \times 25$.
(c) False. John's model will consume more memory when training a network as it needs to compute the intermediate layer representation.
(d) True. It may cover more diverse data. However, it would require more memory to train, as in (c).

9. **[4 points]** Consider the soft linear SVM problem where we assume the bias of the separating hyperplane is zero:

$$\min_{\mathbf{w},\boldsymbol{\xi}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi^{(n)} \tag{1}$$

$$\text{subject to} \quad y^{(n)}\mathbf{w}^\top\phi(\mathbf{x}^{(n)}) \geq 1-\xi^{(n)}, \quad \forall n \tag{2}$$

$$\xi^{(n)} \geq 0, \quad \forall n \tag{3}$$

and its dual problem with the dual optimization variable (Lagrangian multiplier) $\alpha^{(1)},\cdots,\alpha^{(n)}$ associated with (2). Here $\phi(\mathbf{x})$ denotes a feature mapping for the data point $\mathbf{x}$. We assume $C > 0$.

Suppose we have found an optimal solution $\{\alpha^{(n)} : n = 1,\cdots,N\}$ for the dual problem of the soft SVM. Which of the following are *sufficient* condition(s) for a data point $\mathbf{x}^{(n)}$ in the training set to be a support vector? **(Select all that apply)**

(a) $\xi^{(n)} > 0$

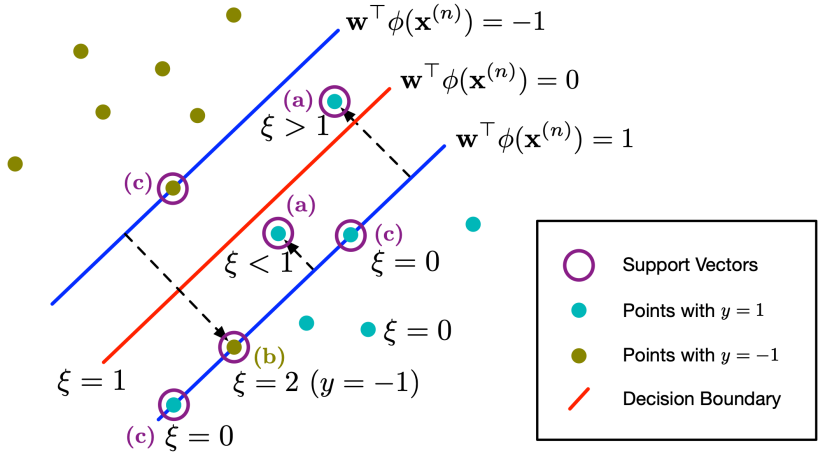
(b) $\xi^{(n)} > 0$ and $\mathbf{w}^\top\phi(\mathbf{x}^{(n)}) = 1$

(c) $\xi^{(n)} = 0$ and $y^{(n)}\mathbf{w}^\top\phi(\mathbf{x}^{(n)}) = 1$

(d) $\alpha^{(n)} > 0$

Solution: (a), (b), (c), and (d) are all True.

Explanation: Support vectors are the points in the training set that have margin of 1 or less (See p.22, Lecture 9). In other words, $\mathbf{x}^{(i)}$ is a support vector if and only if the equality holds in the inequality constraint (2), i.e., $y^{(n)}\mathbf{w}^\top\phi(\mathbf{x}^{(n)}) = 1-\xi^{(n)}$ (p.31, Lecture 10).



(a) True. $\xi^{(n)} > 0$ is when a slack variable is activated because either this point is misclassified or it has smaller margin than 1. So $\mathbf{x}^{(n)}$ is a support vector, contributing to the solution $\mathbf{w}$. The margin for data point $\mathbf{x}^{(n)}$ becomes $1-\xi^{(n)} < 1$.

(b) True. $\xi^{(n)} > 0$ already makes $\mathbf{x}^{(n)}$ a support vector, so if (a) is True then (b) is also True. Note that the latter condition is possible when $y^{(n)} = -1$ and the point is misclassified with $\xi^{(n)} = 2 > 0$: $y^{(n)}(\mathbf{w}^\top\phi(\mathbf{x}^{(n)})) = (-1)\cdot(1) = 1-\xi^{(n)} = -1$.

(c) True. This also describes the condition to be a support vector: The data point is correctly classified ($y^{(n)} = \mathbf{w}^\top\phi(\mathbf{x}^{(n)})$), but lies on margin exactly 1 (the blue line in the figure): $y^{(n)}(\mathbf{w}^\top\phi(\mathbf{x}^{(n)})) = 1 = 1-\xi^{(n)}$. See also (p.31, Lecture 10).

(d) True. From the dual problem's perspective, $\mathbf{x}^{(n)}$ is a support vector if and only if $\alpha^{(n)} > 0$. The complementary slackness (KKT) condition tells that $\alpha^{(n)}(y^{(n)}\mathbf{w}^\top\phi(\mathbf{x}^{(n)}) - 1 + \xi^{(n)}) = 0$, so if $\alpha^{(n)} \neq 0$ then we get $y^{(n)}(\mathbf{w}^\top\phi(\mathbf{x}^{(n)})) = 1-\xi^{(n)}$, so $\mathbf{x}^{(n)}$ becomes a support vector. We also have seen that the data points with $\alpha^{(n)} = 0$ do not contribute to $\mathbf{w}$ (p.29 and p.34, Lecture 10): $\mathbf{w} = \sum_n \alpha^{(n)}y^{(n)}\phi(\mathbf{x}^{(n)})$. Support vectors are precisely the points that have an effect on the solution of SVM, $\mathbf{w}$.

10. [**4 points**] Which of the following are true about the EM algorithm in training mixture models? **(Select all that apply)** Please assume that we can infer the exact posterior distribution.

(a) The M-step in the EM algorithm tries to find the parameters that increase the expected complete-data log-likelihood of the hidden (latent) variables, given the observed data.

(b) After every iteration of E-step and M-step, the log-likelihood of the model is guaranteed to improve (i.e., never decreases).

(c) The EM algorithm is only suitable for discrete data.

(d) The EM algorithm can get stuck in local optima.

Solution: (a), (b), (d).

(a) True. M-step would not decrease the expected expected complete-data log-likelihood of the hidden (latent) variables, once we can infer the exact posterior distribution.
(b) It is true once we can infer the exact posterior distribution (please see the assumption of the question).
(c) The EM algorithm can be used for both continuous and discrete data, so it is not limited to discrete data.
(d) The algorithm may converge to a parameter estimate that is not the global maximum of the likelihood function but rather a local maximum. This can happen if the likelihood function has multiple peaks or if the initial parameter estimates are not sufficiently close to the global maximum.

11. [**4 points**] Suppose we ran k-means clustering on a dataset with $k = 4$ and $k = 6$ clusters, once for each k, and found that the objective function (the distortion measure) value is much higher for $k = 6$ than for $k = 4$. Which of the following are true? **(Select all that apply)**

(a) This is mathematically impossible to happen.

(b) There must be a better cluster solution for $k = 6$ that will bring a smaller objective function value than the current cluster solution.

(c) We should choose the clusters with $k = 6$.

(d) k-means may have converged to a bad local minimum in the $k = 6$ case. We may want to re-run k-means with multiple random initializations.

Solution: (b), (d)

(a): No, it is possible once the initialization of $k = 6$ is too bad.
(b), (d): True. Any time you add two more clusters from $k = 4$ will give you a better objective than the current one. We may want to rerun k-means for $k = 6$ (and for other k) multiple times.
(c): False. $k = 6$ is not a reasonable cluster, based on the current condition.

12. [**4 points**] Which of the following are true about PCA? **(Select all that apply)**

(a) PCA requires an assumption that the data follows a Gaussian distribution.

(b) PCA requires an assumption that the data is zero-centered.

(c) A principle component generated by PCA is always a linear combination of the input data.

(d) PCA minimizes the distortion error between the data and its projection onto the subspace spanned by the principle components.

Solution: (c), (d).

(a) is False (see Lecture 16 quiz);

(b) is False. PCA can work with the data that does not have this assumption.
(c) is True. PCA is a linear algorithm. $\mathbf{Su} = \lambda\mathbf{u} \implies \mathbf{u} = \frac{1}{\lambda N}\mathbf{X}(\mathbf{X}^\top\mathbf{u})$. Here $\mathbf{X}^\top\mathbf{u}$ is a $[N \times 1]$ matrix, $\mathbf{X}$ is a $[d \times N]$ matrix, so $(\mathbf{X}^\top\mathbf{u}/\lambda N)$ can be seen as the linear combination weight that mixes $\mathbf{X}$ into $\mathbf{u}$.
(d) is True, this option describes the minimization perspective.

13. **[4 points]** In Homework 5, we obtained eigenfaces by obtaining the principal components of PCA over the face dataset. (Note: In this problem, eigenfaces are the principal components, which does not include the mean of the face images). Assume the face images are flattened as $X \in \mathbb{R}^{1,024\times N}$ where N is the number of face images and $1,024$ is the data dimension, i.e., the number of pixels in the flattened grayscale image (32×32). Which of the following are true when computing Eigenfaces? **(Select all that apply)**

    (a) Eigenfaces are the eigenvectors of X.

    (b) The dimension of each eigenface vector must be $1,024$ dimension in this case.

    (c) When obtaining the eigenfaces, all eigenfaces are always orthogonal to each other.

    (d) If some of the images in X included more random blurs, the number of principal components needed to represent 95% of the total variance would decrease.

Solution: (b), (c).

(a): We run a PCA over X by computing eigenvectors over covariance matrix. Eigenfaces are the eigenvectors of the covariance matrix of X.
(d): It could possibly increase the number of Eigenfaces as there would be more variability introduced by random blurring, thus more principal components would be needed to explain the increased variability.

14. **[4 points]** Which of the following statements describe the conceptual differences between variational auto-encoders (VAEs) and diffusion models? **(Select all that apply)**

    (a) Both of VAEs and diffusion models compute the lower bound of the density of samples $p(x)$.

    (b) Both VAEs and diffusion models are generative models that learn a low-dimensional representation of the input data.

    (c) VAEs use an encoder-decoder architecture to generate new data samples, while diffusion models iteratively transform data samples.

    (d) VAEs typically suffer from inefficient sequential generation, but are able to produce samples with better qualities.

Solution: (a), (c).

(b) is incorrect because diffusion models do not learn a low-dimensional representation. Rather, they directly model the data distribution using a Markov chain.
(d) is incorrect because the behaviors are reversed. Diffusion models are able to produce many of the best quality samples, but they suffer from inefficient sequential generation.

15. **[4 points]** Consider a binary classification problem with the following options of model choice. Which of the following models produce a linear separating hyperplane in the feature space? **(Select all that apply)**

    (a) Logistic Regression

    (b) Gaussian Discriminant Analysis with shared covariance

    (c) Gaussian Discriminant Analysis with different covariances for each Gaussian

    (d) Neural Network

Solution: (a), (b) are linear.

(a) linear. Logistic Regression has the form of sigmoid($\mathbf{w}^\top \mathbf{x}$).
(b) linear. See (p.13-22, Lecture 5)
(c) non-linear. It learns a quadratic boundary. (p.25, Lecture 5)
(d) non-linear, can represent arbitrary non-linear functions.

True/False with explanations

Clearly state your choice and provide a 1 - 2 sentence(s) explanation for each question.

16. **[3 points]** Consider a version of linear regression in which the outputs $y \in \mathbb{R}^2$ are two-dimensional vectors (and $x \in \mathbb{R}^{d+1}$ by setting $\mathbf{w} \leftarrow [\mathbf{w}, b]$ and $\mathbf{x} \leftarrow [\mathbf{x}, 1]$). We overwrite the feature $\phi(\mathbf{x})$ as $\mathbf{x}$ for simplicity and assume $h(\mathbf{x})$ to give a 2D vector,

$$h_{\mathbf{w}_1,\mathbf{w}_2}(x) = \begin{bmatrix} \mathbf{w}_1^\top \mathbf{x} \\ \mathbf{w}_2^\top \mathbf{x} \end{bmatrix} \in \mathbb{R}^2 \tag{4}$$

where $\mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^{d+1}$ are the parameters of the model. During training, we want to choose parameters by solving the following minimization (we do not introduce any additional regularization):

$$\min_{\mathbf{w}_1,\mathbf{w}_2} \sum_{i=1}^{N} \left\| h_{\mathbf{w}_1,\mathbf{w}_2}(\mathbf{x}^{(i)}) - \mathbf{y}^{(i)} \right\|^2 \tag{5}$$

Let $\mathbf{y}_1 \in \mathbb{R}^{N\times 1}$ and $\mathbf{y}_2 \in \mathbb{R}^{N\times 1}$ contain all the target values of the matched dimension, from the training set.
(True/False) The optimal values for $\mathbf{w}_1$ and $\mathbf{w}_2$, respectively, are $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}_1$ and $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}_2$.
[Provide a justification to your answer.]

Solution: True.

Maximize with respect to $\mathbf{w}_1$ and $\mathbf{w}_2$ will give the expressions. The first term is independent of $\mathbf{w}_2$, and the second term is independent of $\mathbf{w}_1$. Therefore

$$J(\mathbf{w}_1, \mathbf{w}_2) = (\mathbf{X}\mathbf{w}_1 - \mathbf{y}_1)^\top(\mathbf{X}\mathbf{w}_1 - \mathbf{y}_1) + (\mathbf{X}\mathbf{w}_2 - \mathbf{y}_2)^\top(\mathbf{X}\mathbf{w}_2 - \mathbf{y}_2). \tag{6}$$

Taking the partial derivatives with respect to $\mathbf{w}_1$ and $\mathbf{w}_2$ will give $\nabla_{\mathbf{w}_1} J(\mathbf{w}_1, \mathbf{w}_2) = \mathbf{X}^\top\mathbf{X}\mathbf{w}_1 - \mathbf{X}^\top\mathbf{y}_1$ and $\nabla_{\mathbf{w}_2} J(\mathbf{w}_1, \mathbf{w}_2) = \mathbf{X}^\top\mathbf{X}\mathbf{w}_2 - \mathbf{X}^\top\mathbf{y}_2$, respectively.

Setting these two zero will give us $\mathbf{w}_1 = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}_1$ and $\mathbf{w}_2 = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}_2$.

17. **[3 points] (True/False)** An RBF/Gaussian Kernel will always have better classification performance at a testing time compared to a linear kernel.

[Provide a justification to your answer.]

Solution: Answer: False.

This answer is false because of several possible reasons:

(a) There may be a bad hyperparameter choice for RBF.

(b) Small (finite) training data size could lead the RBF to overfit.

18. **[4 points] (True/False)** When we perform a backpropagation through a layer $\mathbf{z} = f(\mathbf{x})$ with a non-linear activation function f, each element of $\frac{\partial \mathcal{L}}{\partial \mathbf{x}}$ (the gradient of the loss function $\mathcal{L}$ with respect to input $\mathbf{x}$) will always have the same sign as the corresponding element of the gradient $\frac{\partial \mathcal{L}}{\partial \mathbf{z}}$ (the gradient of the loss function $\mathcal{L}$ with respect to $\mathbf{z}$) if $f \in \{\text{sigmoid}, \tanh\}$.

[Instruction: Answer with either (True/False) for *each* of the cases $f = \text{sigmoid}$ and $f = \tanh$. Don't forget to provide a justification to your answer on each choice.]

Solution: True (for both).
Reason: Both sigmoid and tanh are *monotonically increasing.*
Note that $\dfrac{\partial \mathcal{L}}{\partial x_i} = \dfrac{\mathrm{d}z_i}{\mathrm{d}x_i}\dfrac{\partial \mathcal{L}}{\partial z_i}$, and if $\dfrac{\mathrm{d}z_i}{\mathrm{d}x_i} = f'(z) > 0$, then the sign remains the same.

19. **[4 points]** Recall that AdaBoost is an ensemble learning algorithm that combines a set of weak learners to form a strong learner.

(True/False) The weights α_m assigned to the binary classifiers after the m-th boosting round in AdaBoost are always non-negative, assuming that the weak learners are able to at least perform better than random guessing in terms of weighted misclassification error.

[Hint: note that random guess would lead to 50% weighted classification error in this context.]

[Provide a justification to your answer.]

Solution: True.
The weight α_m is updated as $\alpha_m = \ln\left\{\dfrac{1-\epsilon_m}{\epsilon_m}\right\}$ where the weighted error function ϵ_m with data weights w_m is defined as $\epsilon_m = \dfrac{\sum_{n=1}^{N} w_m^{(n)} \mathbb{I}\left(h_m\left(\mathbf{x}^{(n)}\right) \neq y^{(n)}\right)}{\sum_{n=1}^{N} w_m^{(n)}}$.
As the weak classifiers perform better than the random guessing, the weighted error function will give $\epsilon_m < 0.5$ (based on the assumption). Thus, when $\epsilon_m < 0.5$, $\alpha_m = \ln\left\{\dfrac{1-\epsilon_m}{\epsilon_m}\right\}$ will always have non-negative value.

Short answer and proof questions

Write your answer in the space provided below each question.

20. **[3 points]** Suppose we have a (linear) Soft SVM that does binary classification. If we remove a point from the training data that is classified correctly and is far away from the decision boundary, and re-train the SVM, will the decision boundary change or remain the same? Justify your answer briefly.

Solution: It will not change.
Reason: Such points will not be a support vector and the decision bounday of soft SVM is determined by only the support vectors.

21. **[3 points]** Consider the following k-means algorithm formulation for the cluster distortion objective $\mathcal{J}$:

$$\mathcal{J}_k(\boldsymbol{\mu}) = \sum_{c=1}^{k} \sum_{n=1}^{N} r_{nc} ||\mathbf{x}^{(n)} - \mu_c|| \tag{7}$$

where μ_c is the center for cluster μ_c and

$$r_{nc} = \begin{cases} 1 & \mathbf{x}^{(n)} \text{ is in cluster } c \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Suppose we also want to determine the number of clusters, k. Is it appropriate to find the best k by optimizing $\mathcal{J}_k$, i.e., $k = \text{argmin}_{k'} [\min_{\boldsymbol{\mu}} \mathcal{J}_{k'}(\boldsymbol{\mu})]$? Justify your answer briefly.

Solution: No.
If you treat K as a (model) parameter and optimize for J_K, the solution will always converge to $k = N$ with $\mu_k = \mathbf{X}^{(i)}$ since this will result in zero distortion but is of no practical value since every point is in a single cluster.

22. **[5 points] Logistic Regression**

In this problem, we use logistic regression to predict the class label $y \in \{-1, +1\}$ instead of $y \in \{0, 1\}$ as in the ordinary logistic regression. Show that maximizing the log-likelihood of logistic regression, $\sum_{n=1}^{N} \log P(y^{(n)}|\mathbf{x}^{(n)})$, is equivalent to minimizing the following loss function:

$$\sum_{n=1}^{N} \log\left(1 + \exp(-y^{(n)} \cdot \mathbf{w}^\top \phi(\mathbf{x}^{(n)}))\right). \tag{9}$$

[Hint: You can expand the log-likelihood as follows: $\log P(y^{(n)}|\mathbf{x}^{(n)}) = \mathbb{I}(y^{(n)} = 1) \log P(y^{(n)} = 1|\mathbf{x}^{(n)}) + \mathbb{I}(y^{(n)} = -1) \log P(y^{(n)} = -1|\mathbf{x}^{(n)})$ then plug in the class posterior probability of the logistic regression model.]

Solution: The log-likelihood of the data can be re-written as:

$$\sum_{n} \Big(\mathbb{I}(y^{(n)} = 1) \log P(y^{(n)} = 1 \mid \mathbf{x}^{(n)}) + \mathbb{I}(y^{(n)} = -1) \log P(y^{(n)} = -1 \mid \mathbf{x}^{(n)})\Big)$$

First, we plug in the definition class posterior probability of logistic regression model:
$P(y^{(n)} = 1 \mid \mathbf{x}^{(n)}) = \dfrac{1}{1 + \exp(-\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))}$ then we have:

$$\sum_{n} \mathbb{I}(y^{(n)} = 1) \log\left[\frac{1}{1 + \exp(-\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))}\right] + \mathbb{I}(y^{(n)} = -1) \log\left[1 - \frac{1}{1 + \exp(-\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))}\right]$$

In other words, this term can be written in case by case as follows:

- if $y^{(n)} = 1$, then
  $\log\left[\dfrac{1}{1 + \exp(-\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))}\right] = -\log\left[1 + \exp(-y^{(n)}\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))\right]$
- if $y^{(n)} = -1$, then
  $\log\left[1 - \dfrac{1}{1 + \exp(-\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))}\right] = \log\left[\dfrac{1}{1 + \exp(\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))}\right] = -\log\left[1 + \exp(-y^{(n)}\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))\right]$

Therefore, the objective can be compactly written as

$$\sum_{n} \log\left[1 + \exp(-y^{(n)}\mathbf{w}^\top \phi(\mathbf{x}^{(n)}))\right].$$

□

23. **[6 points] Naive Bayes with Bayesian Smoothing**

Recall that Naive Bayes can be solved with MLE, in which we count the occurences of each feature (or word). Adding Laplace smoothing, we get:

$$P(C_i) = \phi_i = \frac{N^{C_i}}{\sum_{i'} N^{C_{i'}}} \tag{10}$$

$$P(x_j|C_i) = \mu_j^i = \frac{N_j^{C_i} + \alpha}{\sum_{j'} N_{j'}^{C_i} + \alpha K} \tag{11}$$

where K is the number of classes and M is the dimension of $\mathbf{x}$ (total number of features or words). $N_j^{C_i}$ is the count of the occurrences of x_j with class C_i. $\alpha > 0$ is the Laplace smoothing hyperparameter.

Show that Laplace smoothing is equivalent to solving the MAP estimate of Naive Bayes, where we have a prior on the values of $\boldsymbol{\mu}$ which follow a symmetric Dirichlet distribution:

$$P(\boldsymbol{\mu}) = \frac{1}{Z} \prod_{i=1}^{K} \prod_{j=1}^{M} \left(\mu_j^i\right)^{\alpha} \tag{12}$$

where Z is some normalizing constant.

[Hint: You may use the Naive Bayes likelihood and MLE derivations from lecture without proof.]

Solution: Recall that the likelihood from lecture is:

$$\begin{aligned} P(\mathbf{X}, \mathbf{y} \mid \boldsymbol{\mu}, \phi) = \prod_i P(\mathbf{x}^{(i)}, y^{(i)} \mid \boldsymbol{\mu}, \phi) &= \prod_{s=1}^{K} \left[\left(\prod_{i:y^{(i)}=s}^{N} \prod_{k=1}^{\text{len}\left(x^{(i)}\right)} \prod_{j=1}^{M} \left(\mu_j^s\right)^{\mathbb{I}\left(x_k^{(i)} = \text{“}j\text{”th word }\right)} \right) \left(\prod_{i:y^{(i)}=1}^{N} \phi_s \right) \right] \\ &= \prod_{s=1}^{K} \left[\left(\prod_{j=1}^{M} \left(\mu_j^s\right)^{N_j^s} \right) \phi_s^{N^s} \right] \end{aligned}$$

Then, the MAP objective is:

$$\begin{aligned} P(\mathbf{X}, \mathbf{y} \mid \boldsymbol{\mu}, \phi) P(\boldsymbol{\mu}) &= \prod_{s=1}^{K} \left[\left(\prod_{j=1}^{M} \left(\mu_j^s\right)^{N_j^s} \right) \phi_s^{N^s} \right] \left(\frac{1}{Z} \prod_{s=1}^{K} \prod_{j=1}^{M} \left(\mu_j^s\right)^{\alpha} \right) \\ &= \frac{1}{Z} \prod_{s=1}^{K} \left[\left(\prod_{j=1}^{M} \left(\mu_j^s\right)^{N_j^s} \right) \phi_s^{N^s} \left(\prod_{j=1}^{M} \left(\mu_j^s\right)^{\alpha} \right) \right] \\ &= \frac{1}{Z} \prod_{s=1}^{K} \left[\left(\prod_{j=1}^{M} \left(\mu_j^s\right)^{N_j^s + \alpha} \right) \phi_s^{N^s} \right] \end{aligned}$$

Note this MAP objective is the same as the likelihood, but we added α to each N_j^s count! Using the MLE derivation from lecture, we maximize the likelihood w.r.t. μ's and we get the final objective:

$$P(x_j \mid C_i) = \mu_j^i = \frac{N_j^{C_i} + \alpha}{\sum_{j'} N_{j'}^{C_i} + \alpha K}.$$

□

24. **[9 points] EM for mixture of linear models**

We have seen that solving linear regression under the least square objective is equivalent to maximizing the likelihood of the 1-D Gaussian $\mathcal{N}(\mathbf{w}^\top \phi(\mathbf{x}), \sigma^2)$ for some fixed variance σ^2. Note that, without the loss of generality, we can drop the bias term in the linear regression model (i.e., $y = \mathbf{w}^\top \phi(\mathbf{x})$ instead of $y = \mathbf{w}^\top \phi(\mathbf{x}) + b$, by setting $\mathbf{w} \leftarrow [\mathbf{w}, b]$ and $\phi(\mathbf{x}) \leftarrow [\phi(\mathbf{x}), 1]$) for simpler notations.

Instead of solving a plain linear regression which fits $y = \mathbf{w}^\top \phi(\mathbf{x})$, we can consider K number of linear regression models $y = \mathbf{w}_k^\top \phi(\mathbf{x}_k)$ and assume the random variable y is generated by their *mixtures*, i.e.,

$$\begin{aligned} z &\sim \text{Categorical}(\pi_1, \cdots, \pi_K) \\ y \mid \mathbf{x}, z = k &\sim \mathcal{N}(y; \mathbf{w}_k^\top \phi(\mathbf{x}), \sigma^2) \end{aligned}$$

where z is a *latent* random variable that can take a value one of $\{1, \cdots, K\}$, $\pi_1, \cdots, \pi_K$ are the mixture weights for the K linear regression components ($\sum_{k=1}^K \pi_k = 1$ and $\pi_k \geq 0, \forall k$), and σ^2 is a fixed hyperparameter (constant). In this setting, the set of model parameters is: $\theta = \{\mathbf{w}_k, \pi_k; k = 1 \cdots, K\}$.

In this problem, you are given a training data $\mathcal{D} = \left\{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \cdots, (\mathbf{x}^{(N)}, y^{(N)})\right\}$ with N samples where $\phi(\mathbf{x}^{(i)}) \in \mathbb{R}^d, y^{(i)} \in \mathbb{R}$. Since this *mixture of linear regression model* is a latent variable model, you can introduce latent variables $z^{(i)}$ for $i = 1, \ldots, N$ such that $z^{(i)} = k$ means $y^{(i)}$ is generated from the k-th linear regression model. Now derive an EM algorithm that solves the MLE (maximum likelihood estimate) of the mixture of linear regression models given the data $\mathcal{D}$.

(a) **[2 points] E-Step:** Write down the E-step. Specifically, find the distribution $q^{(n)}(z^{(n)} = k)$ for each latent variable $z^{(n)}$ that would maximize the variational lower bound of the log-likelihood of the data $\mathcal{D}$, i.e., $q^{(n)}(z^{(n)} = k) = p(z^{(n)} = k \mid y^{(n)}, \mathbf{x}^{(n)}; \theta)$. Show the steps for derivation using the Bayes Rule.

[Note: You do not need to expand the p.d.f. of the Gaussian distribution $\mathcal{N}(\cdot, \cdot)$.]

Solution: Intuitively speaking, the result should be very similar to that of EM-GMM.

For each latent variable $z^{(n)}$, we set $q^{(n)}(z^{(n)}) = p(z^{(n)} \mid y^{(n)}, \mathbf{x}^{(n)}; \theta)$ (the posterior).

$$\gamma_{nk} = p(z^{(n)} \mid y^{(n)}, \mathbf{x}^{(n)}; \theta) = \frac{p(z^{(n)} = k, y^{(n)} \mid \mathbf{x}^{(n)}; \theta)}{p(y^{(n)} \mid \mathbf{x}^{(n)}; \theta)} \tag{13}$$

$$= \frac{p(z^{(n)} = k; \theta) p(y^{(n)} \mid z^{(n)} = k, \mathbf{x}^{(n)}; \theta)}{\sum_{j=1}^K p(z^{(n)} = j; \theta) p(y^{(n)} \mid z^{(n)} = j, \mathbf{x}^{(n)}; \theta)} \tag{14}$$

$$= \frac{\sum_k \pi_k \mathcal{N}(y^{(n)}; \mathbf{w}_k^\top \phi(\mathbf{x}^{(n)}), \sigma^2)}{\sum_{j=1}^K \pi_j \mathcal{N}(y^{(j)}; \mathbf{w}_j^\top \phi(\mathbf{x}^{(n)}), \sigma^2)} \tag{15}$$

[Continued in the next page]

(b) **[5 points] M-Step:** Derive the M-step for updating $\mathbf{w}_k$. More specificially, prove that the maximization problem in the M-step for $\mathbf{w}_k$ can be reduced to a locally-weighted linear regression problem, and use the closed form solution to derive the update rule for $\mathbf{w}_k$.

[Note: Please use $\gamma_{nk} = p(z^{(n)} = k \mid y^{(n)}, \mathbf{x}^{(n)}; \theta)$ in your answer.]

[Hint: The p.d.f. of a 1-D Gaussian is: $\mathcal{N}(y; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right]$.]

[Hint: You can reuse the closed form solution of the locally-weighted linear regression without proof.]

Solution: In the M-step, we maximize the "complete data" log-likelihood weighted by γ_{nk}:

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} \underbrace{q^{(n)}(z^{(n)} = k)}_{=\gamma_{nk}} \underbrace{\log p(z^{(n)} = k, y^{(n)} \mid \mathbf{x}^{(n)}, \theta)}_{\text{"complete data" log-likelihood}^{\clubsuit} \text{ (or "data completion" log-likelihood)}} \tag{16}$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk} \left(\log \pi_k + \log \mathcal{N}(y^{(n)}; \mathbf{w}_k^\top \phi(\mathbf{x}^{(n)}), \sigma^2)\right) \tag{17}$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk} \left(\log \pi_k + \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2}\left[y^{(n)} - \mathbf{w}_k^\top \phi(\mathbf{x}^{(n)})\right]^2\right) \tag{18}$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk} \left((\text{const w.r.t. } \mathbf{w}_k) - \frac{1}{2\sigma^2}\left[y^{(n)} - \mathbf{w}_k^\top \phi(\mathbf{x}^{(n)})\right]^2\right) \tag{19}$$

with respect to $\mathbf{w}_k$, for each $k = 1, \cdots, K$.

Since $\sigma^2 > 0$ is a fixed constant, maximization of J w.r.t. $\mathbf{w}_k$ is equivalent to

$$\min_{\mathbf{w}} \quad \sum_{n=1}^{N} \gamma_{nk}\left[y^{(n)} - \mathbf{w}_k^\top \phi(\mathbf{x}^{(n)})\right]^2 \tag{20}$$

which has the form of a locally-weighted linear regression problem. Its closed form solution is

$$\mathbf{w}_k = (\mathbf{X}^\top \mathbf{R}_k \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{R}_k \mathbf{y} \tag{21}$$

where $\mathbf{X} = [\phi(\mathbf{x}_1)^\top; \phi(\mathbf{x}_2)^\top; \cdots; \phi(\mathbf{x}_n)^\top] \in \mathbb{R}^{N \times d}$ is the data feature matrix, $\mathbf{y} = [y_1; y_2; \cdots; y_n] \in \mathbb{R}^N$ and $\mathbf{R}_k = \text{diag}(\gamma_{1,k}, \gamma_{2,k}, \cdots, \gamma_{nk}) \in \mathbb{R}^{N \times N}$. □

(c) **[2 points] M-Step:** Write down the M-step for updating π_k. You don't need to give a full derivation; please give the final form based on your guess (this is a short answer question).

[Note: Please use $\gamma_{nk} = p(z^{(n)} = k \mid y^{(n)}, \mathbf{x}^{(n)}; \theta)$ in your answer.]

Solution: The derivation is exactly same as in GMM; e.g., take the derivative of Equation (18) w.r.t π_k.

$$\pi_k = \frac{\sum_{n=1}^{N} \gamma_{nk}}{\sum_{k=1}^{K}\sum_{n=1}^{N} \gamma_{nk}} = \frac{\sum_{n=1}^{N} \gamma_{nk}}{N} \tag{22}$$

Additional page provided for scratch work (1/3)

Additional page provided for scratch work (2/3)

Additional page provided for scratch work (3/3)