USC CSCI 467 Intro to Machine Learning	**Midterm Exam** March 7, 2024, 3:30-4:50pm	Spring 2024 Instructor: Robin Jia

Name: _____

USC e-mail: _____@usc.edu

Answer the questions in the spaces provided. **If you run out of space, continue your work on the last two pages, and indicate that your answer is there**. You may use the backs of pages for scratch work only. **Please use pen for ease of grading**. This exam has 6 questions, for a total of 100 points.

Question 1: Fetal Weight Estimation (26 points)

In medicine, various formulas have been developed to estimate the weight of a developing fetus during pregnancy. This is important to estimate because low weight or excessive weight can both cause certain medical complications.

Fetal weight (abbreviated as FW) is difficult to measure directly; however, using ultrasound imaging, various lengths and circumferences of the fetus can be measured, from which FW can be estimated. In particular, the following can be measured using ultrasound (all in centimeters):

- Head circumference (HC).
- Abdominal circumference (AC).
- Femur length (FL): Length of the thigh bone.
- Biparietal Diameter (BPD): Distance between two parietal bones of the skull.

Ryan wants to train a machine learning model to predict the fetal weight using these measurements. He obtains a training dataset $D = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$, where each $x^{(i)} \in \mathbb{R}^4$ contains the HC, AC, FL, and BPD measurements for a fetus (in that order), and $y^{(i)}$ is the corresponding fetal weight (in grams).

(a) As a baseline approach, Ryan decides to train a linear model directly on this dataset with no other pre-processing.

i. (1 point) Should he use linear regression, logistic regression, or softmax regression?

Solution: It is a regression problem, so we would use linear regression.

ii. (2 points) Ryan will learn a weight w and bias b. State whether each one is a scalar, vector, or matrix. For each vector or matrix, state its dimension(s).

Solution: w is a vector of dimension 4. b is a scalar.

iii. (2 points) At test time, Ryan's model is given the HC, AC, FL, and BPD measurements for a new fetus. Write the formula that describes the prediction that the model will make in terms of these four measurements and the parameters of the model.

Solution: The formula will be

$$FW = w_1 \times HC + w_2 \times AC + w_3 \times FL + w_4 \times BPD + b.$$

This is equivalent to $w^\top x + b$ where x is the vector $[HC, AC, FL, BPD]$.

(b) After training this baseline linear model, Ryan then learns that in some cases, only AC, FL, and HC can be measured, but not BPD. In these cases, he would need to predict FW using only the first three features.

i. (4 points) Ryan has an idea: He can just set BPD equal to 0 and use the trained model from the previous part to make a prediction. Is this likely to work well or not? Explain your reasoning in 1-2 sentences.

Solution: No, this is a bad idea. BPD is always positive (since it is the length of something), so setting it to 0 is essentially saying that the fetus has a very

abnormally low BPD. This will likely result in bad predictions. In general, discriminative methods like linear regression are not very good at handling missing features.
The better solution would just be to train an entirely different model using only AC, FL, and HC.

ii. (2 points) Name one method we learned about in class that can be used even if some features, like BPD, are not observed. You may not use the same answer as part a(i).

Solution: We learned about two such methods. Naive Bayes is good in these situations because we can just omit the missing feature when computing $P(x \mid y)$ for each label y. Decision trees are also useful in these situations (this is another acceptable answer even though decision trees technically aren't covered on this exam).

(c) Ryan reasons that the weight should be proportional to volume, which is in units of cubic centimeters. Therefore, he think that a good formula for FW should involve features where 3 of the original features are multiplied together.

i. (3 points) In 1-2 sentences, explain why the model from the previous part cannot learn the type of formula that Ryan wants.

Solution: The product of three features is not a linear function of the original features. Since we were training a linear model, it can only learn a linear function of the four original features.

ii. (4 points) Describe **two** different ways Ryan could train a model to learn the type of formula he is looking for. Explain your answer in detail.

Solution:

1. Ryan could directly add all products of the three original features as new features to his model. This is a feature engineering-based solution.
2. Ryan could use a polynomial kernel with degree 3. This will implicitly be using all possible features involving the product of up to 3 original features.

Note that partial credit was given for answers that use a neural network. While neural networks are universal approximators, they would not exactly represent a product of three original features, unlike the two answers above.

(d) (8 points) One actual formula used to estimate fetal weight is the following:

$$FW = 10^{1.3596 - 0.00386 \times AC \times FL + 0.0064 \times HC + 0.00061 \times BPD \times AC + 0.0424 \times AC + 0.174 \times FL}$$

How do you think the six numbers in this formula (the 1.3596, -0.00386, etc.) were chosen? Describe in detail a procedure that could be used to come up with these numbers. Assume you have access to the same training dataset as in the previous parts.

Solution: First, notice that we can rewrite the formula as:

$$\log_{10}(FW) = 1.3596 - 0.00386 \times AC \times FL + 0.0064 \times HC + 0.00061 \times BPD \times AC + 0.0424 \times AC + 0.174 \times FL.$$

In other words, this formula assumes that $\log_{10}(FW)$ is a linear function of the following feature vector:

$$\phi(x) = \begin{pmatrix} 1 \\ AC \cdot FL \\ HC \\ BPD \cdot AC \\ AC \\ FL \end{pmatrix}$$

So, the way to learn the weights is to transform the dataset by replacing each $y^{(i)}$ with $\log_{10}(y^{(i)})$ and replace $x^{(i)}$ with the features listed above. We would then train linear regression with a weight vector $w \in \mathbb{R}^6$. (It is also acceptable to define ϕ as a 5-dimensional vector without the 1 term, and say you will also train with a bias term b).

Note that taking the log is important, because if we instead directly try to predict FW, then (according to the formula) FW is not a *linear* function of $AC \cdot FL$, HC, etc. We need to transform the data so that what we're predicting is a linear function of the features, so that we can run linear regression.

Many students had something close to the right answer, but described ϕ as a kernel. Kernels are not needed to answer this question. ϕ is just a function that adds features—it represents the process of doing feature engineering, not necessarily kernels. The connection with kernels is just that *particular* types of ϕ correspond to particular kernel functions that are useful.

Question 2: Combining Kernels (16 points)

(a) (8 points) In class, we discussed that many different functions $k(\mathbf{x}, \mathbf{z})$ can be used as kernels. It turns out that not *every* function is suitable as a kernel. To prove that a function k is a valid kernel, a necessary and sufficient condition is to show that there exists a feature mapping ϕ such that $k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^\top \phi(\mathbf{z})$ for all $\mathbf{x}$ and $\mathbf{z}$.

Let k_1 and k_2 be valid kernels; that is, $k_1(\mathbf{x}, \mathbf{z}) = \phi_1(\mathbf{x})^\top \phi_1(\mathbf{z})$ and $k_2(\mathbf{x}, \mathbf{z}) = \phi_2(\mathbf{x})^\top \phi_2(\mathbf{z})$, for some feature mappings ϕ_1 and ϕ_2, respectively. Show that the function

$$k(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z}) + k_2(\mathbf{x}, \mathbf{z})$$

is a valid kernel by explicitly constructing a corresponding feature mapping $\phi(\mathbf{z})$.

Solution:

$$\begin{aligned} k(x, y) &= k_1(x, y) + k_2(x, y) \\ &= \phi_1(\mathbf{x})^\top \phi_1(\mathbf{y}) + \phi_2(\mathbf{x})^\top \phi_2(\mathbf{y}) \\ &= \left[\phi_1(\mathbf{x}), \phi_2(\mathbf{x})\right]^\top \left[\phi_1(\mathbf{x}), \phi_2(\mathbf{x})\right], \end{aligned}$$

where $[\phi_1(\mathbf{x}), \phi_2(\mathbf{x})]$ denotes concatenation of the two vectors. If we let $\phi(\mathbf{z}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x})]$, then we have $k(x, y) = \phi(\mathbf{z})^\top \phi(\mathbf{z})$. Therefore, $k = k_1 + k_2$ is a valid kernel.

(b) (4 points) Soumya is training a kernelized SVM on a classification task. He tries a polynomial kernel of degree 3 and gets 100% training accuracy but 74% development accuracy. He also tries an RBF kernel and gets 100% training accuracy but 75% development accuracy. Soumya wonders if he could get much better accuracy by using the combined kernel that adds the polynomial kernel and RBF kernel (as in the previous part). Is this likely to make his model much better? Explain your reasoning.

Solution: No, in this situation Soumya's main problem is that his model is overfitting. Adding the two kernels together essentially adds features together. It can improve the bias of the method by making it more expressive, but it would not reduce variance, which is the source of overfitting.

(c) (4 points) Soumya wants to keep using SVM with an RBF kernel. Suggest two things Soumya should try to improve the performance of this model.

Solution: The expected answers are:

- Increase the L2 regularization.
- Acquire more training data.

Note that for SVM's, the only regularization that is used is L2 regularization. Other forms of regularization did not receive full credit.

Question 3: Receptive Fields in CNNs (18 points)

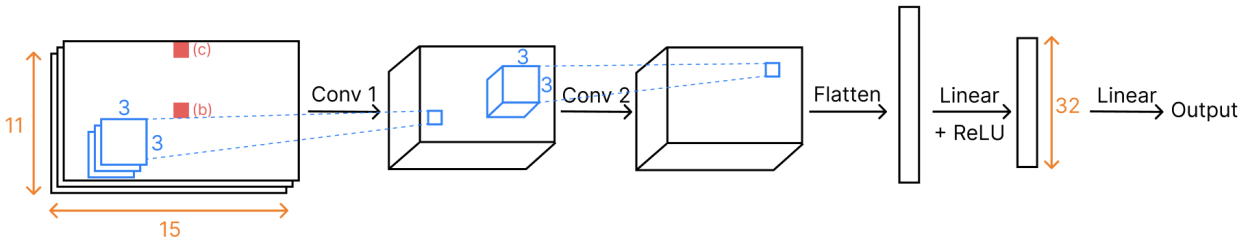
Recall from class that in a Convolutional Neural Network, one of the motivations is that "each neuron should only look at a small patch of input." The portion of the input that affects the activation of a neuron is called its receptive field. Let's make this notion concrete.

Consider a CNN defined by the following operations:

- Input: $3 \times 15 \times 11$ pixel image (number of channels $\times$ width $\times$ height)
- 1st Convolutional Layer: 3×3 kernel, 4 output channels, no padding, stride is 1. ReLU is applied afterwards.
- 2nd Convolutional Layer: 3×3 kernel, 4 output channels, no padding, stride is 1. ReLU is applied afterwards.
- Flatten into a single vector, feed to an MLP with hidden state size of 32 to make a prediction.

Note that there is no max-pooling in this CNN. The diagram below visualizes the entire model, as well as the pixels referred to in parts (b) and (c).

(a) What are the sizes of the activations produced by each convolutional layer? Give your answers in the form $C \times W \times H$ where C is the number of channels, W is width, and H is height.



i. (3 points) First convolutional layer: _____

Solution: There are 4 output channels, and convolving with a 3×3 kernel reduces the width and height by $3 - 1 = 2$. Thus, the answer is $4 \times 13 \times 9$.

ii. (3 points) Second convolutional layer: _____

Solution: By similar reasoning, the width and height reduces by 2 again, so the answer is $4 \times 11 \times 7$.

(b) Suppose we modify the value of the pixel in the **exact middle** of the input image.

i. (2 points) For the **first** convolutional layer, how many of the output values would be affected? Remember that there are multiple output channels.

Solution: This pixel can appear in 9 different positions within a 3×3 window: top left, top middle, ..., bottom right. So, this means that 9 different positions of the kernel are affected. There are 4 output channels, so the total number of affected values is $9 \times 4 = \boxed{36}$.

ii. (2 points) For the **second** convolutional layer, how many of the output values would be affected? Remember that there are multiple output channels.

Solution: From the previous part, we know that there is a 3×3 window of the first convolutional layer that was affected. The second convolutional layer now applies a 3×3 convolution to the output of the first convolutional layer. There will be a 5×5 patch that will touch at least 1 of the 3×3 changed values from the first convolutional layer's output. Since there are again 4 output channels, the total answer is $5 \times 5 \times 4 = \boxed{100}$.

(c) Now suppose we modify the value of the pixel in the **middle of the top edge** of the input image.

i. (2 points) For the **first** convolutional layer, how many of the output values would be affected? Remember that there are multiple output channels.

Solution: Since the pixel is on the edge, only 3 possible 3×3 windows contain that pixel. So the answer is $3 \times 4 = \boxed{12}$.

ii. (2 points) For the **second** convolutional layer, how many of the output values would be affected? Remember that there are multiple output channels.

Solution: From the previous part, we know that a row of 3 pixels on the top edge of the first convolution's output was changed. When we apply the second convolution, there's now 5 windows that touch one of those 3 changed pixels from the first convolutional layer's output. So, the total answer is $5 \times 4 = \boxed{20}$.

(d) (4 points) Based on your responses, does the first or second convolutional layer have a larger receptive field? Explain your answer.

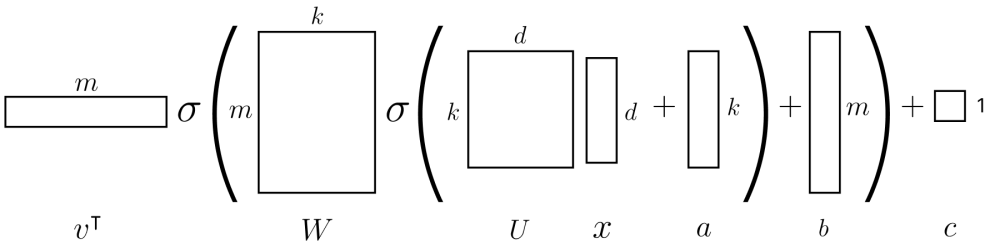
Solution: The second convolutional layer has a larger receptive field, because the outputs of the second convolutional layer are sensitive to a larger number of pixels in the input.

Question 4: Symmetries in Neural Networks (10 points)

Consider a 3-layer fully-connected neural network defined by:

$$g(x) = v^\top \sigma(W\sigma(Ux + a) + b) + c$$

where $U \in \mathbb{R}^{k \times d}$, $a \in \mathbb{R}^k$, $W \in \mathbb{R}^{m \times k}$, $b \in \mathbb{R}^m$, $v \in \mathbb{R}^m$, $c \in \mathbb{R}$ are the parameters, σ is the sigmoid activation function, and $x \in \mathbb{R}^d$ is the input. The diagram below visualizes the dimensions of everything in this formula.



(a) (4 points) Suppose we replace U with U' where two rows i and j of U are swapped, and also replace a with a' where the i-th and j-th entries of a are swapped. All other values stay the same. How do the activations of the original first layer, $\sigma(Ux + a)$, differ from the activations of the new first layer, $\sigma(U'x + a')$?

Solution: The result is that the new first layer activations are the same as the old ones, except the i-th and j-th neurons switch places. You can see this because Ux is just the vector whose i-th entry is the i-th row of U dot-producted with x. So if we swap the i-th and j-th rows of U, this just causes the i-th and j-th entries of Ux to swap.

(b) (6 points) Find values $W' \in \mathbb{R}^{m \times k}$ and $b' \in \mathbb{R}^m$ such that

$$g'(x) = v^\top \sigma(W'\sigma(U'x + a') + b') + c$$

is equal to $g(x)$ for all inputs x.

Solution: The solution is to have W' equal to W except that columns i and j are swapped. b' can remain the same. This works because $\sigma(U'x + a')$ is a vector where the i-th and j-th entries are reversed relative to before. Multiplying this vector with W' is the same as taking a linear combination of the columns of W. So to keep things the same, we should swap columns i and j of W.

Question 5: Short Response (10 points)

Answer the following questions and **explain your reasoning fully**. You may also draw explanatory diagrams when appropriate.

(a) (5 points) Vishesh is using Naive Bayes for text classification. He gathers enough training data so that every word in the vocabulary occurs multiple times in his dataset. Because of this, Vishesh decides that he does not need to use Laplace Smoothing, since every word has a non-zero probability of occurring in his dataset. Is Vishesh right? Explain your answer.

Solution: No, Vishesh still needs to use Laplace Smoothing. Even if each word occurs multiple times, it may not occur with each label. For instance, suppose we are classifying articles as being about sports vs. non-sports. If "bat" appears once in sports and never in non-sports, we don't want $p(\text{bat}|\text{non-sports}) = 0$. Then, any document that contains "bat" can never be classified as non-sports. That is extreme. Using smoothing alleviates this.

Some students wrote that Vishesh is wrong because he will still encounter new words at test time. While this is true, this would happen even if he did do Laplace Smoothing, so it's not an explanation of why Laplace Smoothing is needed. To deal with new words at test time, the easiest solution is simply to exclude them from the calculation of $P(x \mid y)$.

(b) (5 points) Consider a linear model with parameter vector $w \in \mathbb{R}^d$ for binary classification. For a given training dataset, we can compute the **zero-one loss** as follows:

$$L(w) = \sum_{i=1}^{n} \mathbb{I}[y^{(i)} w^\top x^{(i)} \leq 0].$$

Recall that $\mathbb{I}[\cdot]$ is the indicator function that is 1 if the input is true, and 0 if it is false. Is it possible to use gradient descent to learn w by minimizing this loss function?

Solution: No, this would not work. The indicator function's derivative is 0 everywhere, since it is piecewise constant. Thus, gradient descent would never do anything.

Partial credit is given for just saying that the indicator function is not differentiable everywhere. While this is true, this is also true for the hinge loss, and the hinge loss is a viable training objective.

Question 6: Multiple Choice (20 points)

In the following questions, circle the correct answer(s). There is no need to explain your answer.

(a) (2 points) **True** or **False**: The objective function used in L_2-regularized logistic regression is convex.

Solution: True. The normal logistic regression loss function is convex, the L2 regularization term is convex, so their sum is also convex.

(b) (2 points) **True** or **False**: If a loss function is convex, then there is a unique value of the model parameters that minimizes this loss function.

Solution: False. There may be multiple solutions that are equally good. For example, if we do linear regression with $n < d$ (number of examples less than number of features), then there are many equally good solutions.

(c) (2 points) **True** or **False**: In SVMs, the support vectors are the $x^{(i)}$'s for which the corresponding α_i is equal to 1 or -1.

Solution: False. The support vectors have non-zero α_i but it need not be equal to ± 1.

(d) (2 points) **True** or **False**: Non-parametric methods have no learnable parameters.

Solution: False. The definition of a non-parametric method is that it uses the training set during test time. Kernel-based methods are non-parametric but also have learnable parameters (i.e., α).

(e) (3 points) Which of the following classification methods make a prediction on an input x by computing $\max_y P(y \mid x)$? Choose all that apply.

A. Logistic regression
B. Naive Bayes
C. k-Nearest Neighbors
D. Support Vector Machines

Solution: A and B. This describes both Logistic Regression and Naive Bayes: Logistic Regression directly predicts $P(y \mid x)$, and Naive Bayes computes it via Bayes' Rule. Neither k-NN nor SVM generate probabilistic predictions.

(f) (3 points) Which of the following could be a reasonable choice for an activation function in a neural network? Choose all that apply.

A. $g(z) = \max(z, 0)$
B. $g(z) = \sigma(z)$
C. $g(z) = e^z$
D. $g(z) = -\max(z, 0)$

Solution: A, B, C, D. All four choices are possible activation functions, since they are all differentiable non-linear functions.

(g) (3 points) Which of the following are recommended ways to reduce overfitting in a neural network model? Choose all that apply.

A. Increase the number of neurons in the hidden layers.

B. Monitor the training loss and stop running gradient descent if the training loss goes up.

C. Randomly set half the activations to zero during training, and multiply the other activations by 2.

D. Add a term to the loss that encourages the L_2 norm of the weight matrices to be large.

Solution: C only. A would not reduce overfitting as it adds parameters. B is an incorrect description of early stopping; you should use the development loss, not training loss. C describes dropout. D is an incorrect description of L_2 regularization, as you want to make the L_2 norm smaller.

(h) (3 points) Which of the following describe reasonable ways to learn word vectors? Choose all that apply.

A. Train the word vectors so that if two words occur next to each other, their word vectors are made more similar.

B. Use word vectors as part of a language model, and train the language model on a training dataset of text.

C. Train the word vectors so that they can predict which other words each word co-occurs with.

D. Use a training dataset containing word analogies (e.g., "apple is to tree as grape is to vine").

Solution: B and C. B is how word vectors are learned for an RNN language model, and C is a description of word2vec. A is not correct: words should have similar vectors if they tend to co-occur with the same other words, but not necessarily if they co-occur with each other. For D, word vectors trained via word2vec are good for analogies, but you would not use the analogies to train the word vectors.

[This page provides extra space for answers]

[This page provides extra space for answers]