USC CSCI 467 Intro to Machine Learning	**Final Exam** May 7, 2024, 2:00-4:00pm	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 150 points.

Question 1: Language Model-Based Generative Classifier (25 points)

Lorena wants to analyze social media posts about the upcoming election. Therefore, she wants to build a classifier that takes in a post, and classifies it either as having a liberal (L), conservative (C), or neutral (N) perspective. However, she does not have any training data for this task.

Lorena has an idea: she can use a *pre-trained* Transformer language model to help. She writes three *prompts*, one for each label:

- Liberal prompt s_L: "This is a liberal-leaning social media post:"
- Conservative prompt s_C: "This is a conservative-leaning social media post:"
- Neutral prompt s_N: "This is a politically neutral social media post:"

To make a classification decision on a post x, she can ask the language model to predict the probability of generating post x after each one of these prompts. For instance, if x is the string "I support Joe Biden," she would measure the probability of generating "I support Joe Biden" after each of the three prompts.

More formally, she computes $p(x \mid s_L)$, $p(x \mid s_C)$, and $p(x \mid s_N)$ using the language model. She then treats these values as the probability of x given that the label y is either liberal, conservative, or neutral, respectively.

Finally, she decides to assume that all three labels are equally likely. This allows her to build a *generative* classifier. Recall that given a test input x, a generative classifier estimates $P(y)$ and $P(x \mid y)$ for each possible label y, and then uses Bayes Rule to make a prediction.

(a) (2 points) Let x be a post consisting of just two words x_1 and x_2 (in that order). Which of the following accurately describes the process of computing $p(x \mid s_L)$? For two strings u and v, let $u + v$ denote the concatenation of those strings. Circle the best answer.

A. Multiply $p(x_1 \mid s_L)$ and $p(x_2 \mid s_L)$
B. Multiply $p(x_1 \mid s_L)$ and $p(x_2 \mid s_L + x_1)$
C. Add $p(x_1 \mid s_L)$ and $p(x_2 \mid s_L)$
D. Add $p(x_1 \mid s_L)$ and $p(x_2 \mid s_L + x_1)$

Solution: B is correct. You condition x_1 only on the prompt, but condition x_2 on both the prompt and x_1. We are not taking the log of the probability values, so we should multiply, not add.

(b) (5 points) Using the notation from the question preamble, write the formula for $P(y = C \mid x)$, the probability that a post x is conservative according to Lorena's classifier.

Solution:

$$\begin{aligned} P(y = C \mid x) &= \frac{1/3 \cdot p(x \mid s_C)}{1/3 \cdot p(x \mid s_C) + 1/3 \cdot p(x \mid s_L) + 1/3 \cdot p(x \mid s_N)} \\ &= \frac{p(x \mid s_C)}{p(x \mid s_C) + p(x \mid s_L) + p(x \mid s_N)}. \end{aligned}$$

(c) Lorena remembers that you should always inspect your data when doing machine learning. After reading many social media posts, she realizes that her previous assumption that all three labels are equally likely is wrong. Actually, liberal and conservative posts each occur around 45% of the time, and only 10% of posts are neutral.

i. (3 points) Write the modified formula for $P(y = C \mid x)$, taking into account this information.

> **Solution:**
>
> $$P(y = C \mid x) = \frac{.45 \cdot p(x \mid s_C)}{.45 \cdot p(x \mid s_C) + .45 \cdot p(x \mid s_L) + .1 \cdot p(x \mid s_N)}$$

ii. (5 points) Will Lorena need to re-train her language model to account for this? Explain why or why not.

> **Solution:** No. She only needs to adjust the prior probabilities of y. The computation for $P(x \mid s)$ for each prompt s does not change.

(d) Lorena gets tired of looking at social media posts, and instead wants to analyze bills that were proposed in Congress. She again wants to identify whether they promote liberal, conservative, or neutral policies. These bills are much longer than social media posts. Lorena is considering whether she should switch from a Transformer language model to an RNN language model to analyze these much longer documents.

i. (5 points) Give one reason why switching to an RNN would be a **good** idea. Explain your reasoning.

> **Solution:** Either of the following would work:
>
> - The time it takes to run an RNN is linear in the length of the document, while for Transformers it is quadratic. So the RNN should be faster for very long documents.
> - Transformers learn positional embeddings, so they can only process a maximum document length. In contrast, RNN's can process documents of any length.

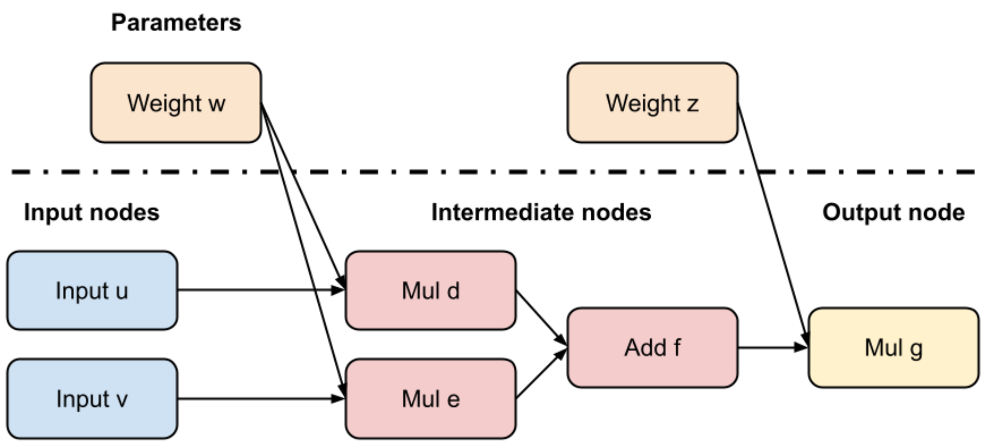
ii. (5 points) Give one reason why switching to an RNN would be a **bad** idea. Explain your reasoning.

> **Solution:** Transformers' attention module makes them better able at capturing long-range dependencies than RNN's. Thus, we expect Transformers to be more accurate at understanding these very long documents.

Question 2: Backpropagation (31 points)

In this question, we are going to work through the computation of gradients for a simplified neural network. Our focus will be on handling gradients for shared parameters. These are parameters that are used multiple times in the forward pass of the network.

Consider the simplified neural network depicted below. Let's work through the implementation of backpropagation for the network. The network consists of input nodes u and v and parameter nodes w and z, which are all scalar numbers for this question. Intermediate computations are named d, e, f. Notice that w is a shared parameter that is used twice in the computation of g.



$$
\begin{aligned}
d &= w \times u \\
e &= w \times v \\
f &= d + e \\
g &= z \times f
\end{aligned}
$$

(a) (3 points) The first thing we need to do is maintain a topological ordering of the nodes. This list will be used (in the reverse order) to compute gradients during backpropagation. Complete this topologically sorted list:

$$u, v, \_\_\_\_, \_\_\_\_, \_\_\_\_, \_\_\_\_, \_\_\_\_, g$$

Solution: There are several valid solutions. One natural solution is: u, v, $\underline{w}$, $\underline{d}$, $\underline{e}$, $\underline{f}$, $\underline{z}$, g.
You may swap the order of d and e, and z can go anywhere in the list before g. The important thing is just to have an order where all the arrows in the diagram are pointing forward.

(b) Now, compute the partial derivatives of g with respect to other nodes in the graph. Remember that you can reuse gradient expressions that you have previously computed, just like in backpropagation. Show your work. We have copied over the formulas from the previous page for convenience:

$$
\begin{aligned}
d &= w \times u \\
e &= w \times v \\
f &= d + e \\
g &= z \times f
\end{aligned}
$$

(c) (2 points) Compute $\frac{\delta g}{\delta z}$.

Solution:

$$\frac{\delta g}{\delta z} = f$$

(d) (2 points) Compute $\frac{\delta g}{\delta f}$.

Solution:

$$\frac{\delta g}{\delta f} = z$$

(e) (3 points) Compute $\frac{\delta g}{\delta d}$.

Solution:

$$\frac{\delta g}{\delta d} = \frac{\delta g}{\delta f} \cdot \frac{\delta f}{\delta d} = z \cdot 1 = z$$

(f) (3 points) Compute $\frac{\delta g}{\delta e}$.

Solution:

$$\frac{\delta g}{\delta e} = \frac{\delta g}{\delta f} \cdot \frac{\delta f}{\delta e} = z \cdot 1 = z$$

(g) (4 points) Compute $\frac{\delta g}{\delta w}$.

Solution:

$$
\begin{aligned}
\frac{\delta g}{\delta w} &= \frac{\delta g}{\delta f} \cdot \frac{\delta f}{\delta w} \\
&= \frac{\delta g}{\delta f} \cdot (\frac{\delta f}{\delta d} \cdot \frac{\delta d}{\delta w} + \frac{\delta f}{\delta e} \cdot \frac{\delta e}{\delta w}) \\
&= z \cdot (1 \cdot u + 1 \cdot v) \\
&= z \cdot (u + v)
\end{aligned}
$$

(h) (8 points) Let's bring it all together. Suppose we are given the following loss function:

$$\mathcal{L} = \sum_{i=1}^{N} (y^{(i)} - g^{(i)})^2 = \sum_{i=1}^{N} (y^{(i)} - NN(u^{(i)}, v^{(i)}))^2,$$

where $g^{(i)} = NN(u^{(i)}, v^{(i)})$ is the output of applying the neural network on the features $(u^{(i)}, v^{(i)})$ of datapoint i and $y^{(i)}$ is the scalar target output for datapoint i. Compute $\frac{\delta \mathcal{L}}{\delta w}$. Show your work.

Solution:

$$\begin{aligned}\frac{\delta \mathcal{L}}{\delta w} &= \sum_{i=1}^{N} 2 \cdot (y^{(i)} - g^{(i)}) \cdot -1 \cdot \frac{\delta g^{(i)}}{\delta w} \\ &= -2 \sum_{i=1}^{N} (y^{(i)} - g^{(i)}) \cdot z \cdot (u^{(i)} + v^{(i)})).\end{aligned}$$

(i) (6 points) In this architecture, w is a shared parameter, as it is used in multiple independent computations. Name two architectures from class that use shared parameters, and state which parameters are shared.

Solution: Reasonable answers include:

1. Kernels in a Convolutional Neural Network: Each patch of the image is convolved with the same convolution kernels. These kernels accumulate gradients from all patches that they act on.
2. Parameters in RNNs/LSTMs/Transformers: All sequence-based models reuse the parameters at each time step. For example, the key, query, and value weights of each attention head are applied to every token. They accumulate gradients from all steps of the sequence.

Question 3: Principal Component Analysis (23 points)

Consider a dataset $\mathbf{X}$ consisting of the following four data points in a two-dimensional space:

$$\mathbf{X} = \begin{bmatrix} 4 & 1 \\ 2 & 3 \\ 5 & 4 \\ 1 & 0 \end{bmatrix}$$

We want to represent the data in only one-dimension and turn to principal components analysis (PCA) for this.

(a) To run PCA, there is an important preprocessing step that must first be done to $\mathbf{X}$.

i. (3 points) In English, describe what this step is.

Solution: The data needs to be mean-centered. In other words, we need to subtract the mean of each column from the respectively column entries.

ii. (3 points) Compute $\tilde{\mathbf{X}}$, the result of applying this preprocessing step to $\mathbf{X}$.

Solution:

$$\tilde{\mathbf{X}} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \\ 2 & 2 \\ -2 & -2 \end{bmatrix}$$

(b) (4 points) Calculate the covariance matrix $\mathbf{\Sigma}$, using your $\tilde{\mathbf{X}}$ computed in the previous part.

Solution:

$$\mathbf{\Sigma} = \frac{1}{4}\mathbf{X}^\top\mathbf{X} = \frac{1}{4}\begin{bmatrix} 10 & 6 \\ 6 & 10 \end{bmatrix} = \begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{bmatrix}$$

Using 1/3 instead of 1/4 is also acceptable as the sample covariance matrix can be computed using $\frac{1}{N-1}$.

(c) If the answer to your previous part is correct, the eigenvectors of $\mathbf{\Sigma}$ are $\begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}^\top$ with eigenvalue 1 and $\begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}^\top$ with eigenvalue 4.

i. (4 points) By running PCA, you can find the one-dimensional subspace that best fits the shape of your data. Based on the eigenvectors and eigenvalues above, this subspace is spanned by what vector? How do you know?

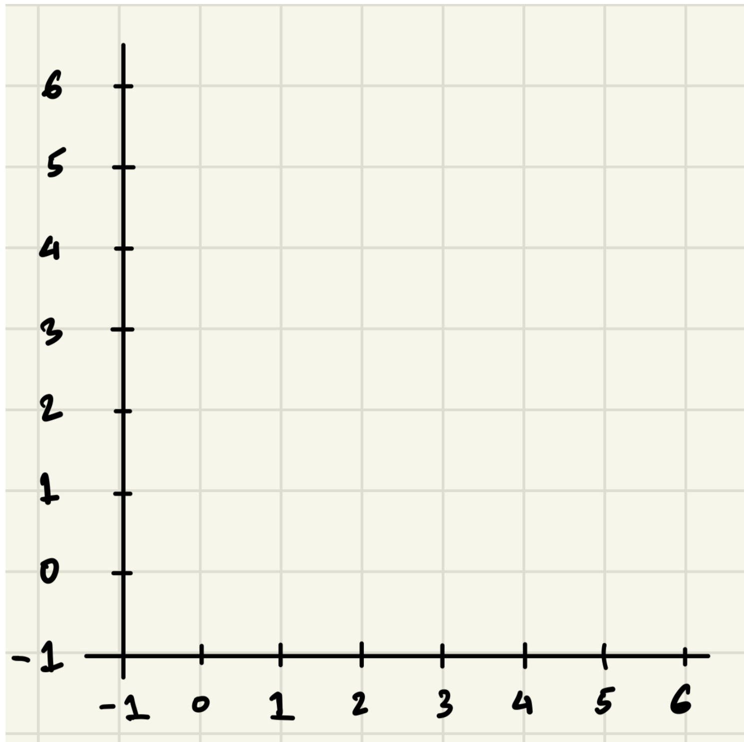
Solution: We choose the eigenvector corresponding to the largest eigenvalue, so the answer is $\begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}^\top$.

ii. (3 points) Compute the result of projecting the four datapoints onto one-dimensional space.

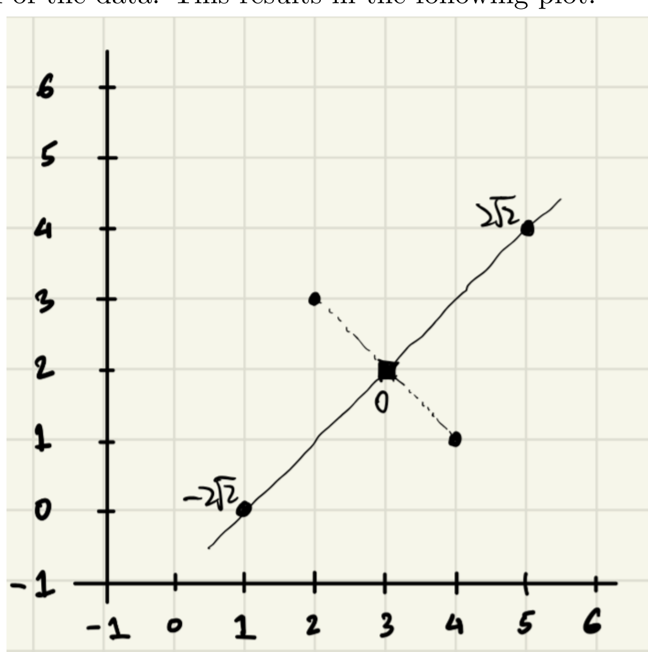
Solution: We compute the principal components by taking the dot product of the eigenvector with each data point. The principal coordinates are 0, 0, $2\sqrt{2}$, and $-2\sqrt{2}$, respectively.
It was also acceptable, though not required, to further multiply this value with the eigenvector again. This follows the formula we used in class, where the vector projection is $(w^\top x)w$ for the eigenvector w. This would yield the vectors

$$[0,0], [0,0], [2,2], [-2,-2].$$

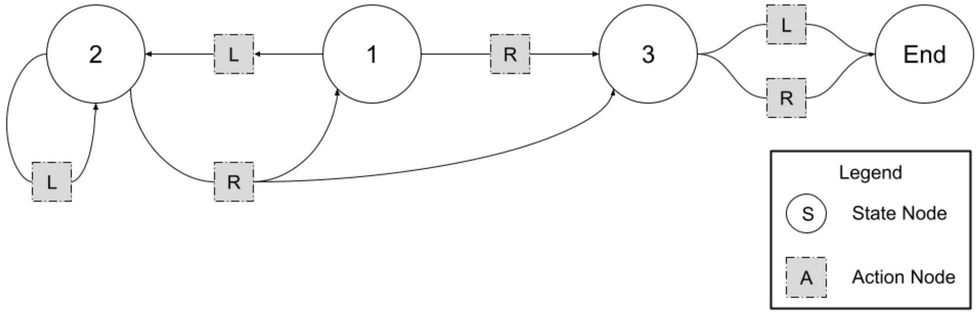
(d) (6 points) Plot the **original** data points, the principal component direction (as a line), and the projections of all four sample points onto the principal direction. Label each projected point with its principal coordinate value.



Solution: We first apply the projection to the mean-centered data. Then, to plot the original data, we undo the mean-centering, i.e., we shift both the datapoints and the line by the mean of the data. This results in the following plot:



Question 4: Reinforcement Learning (29 points)



Consider the simple Markov Decision Process (MDP) in the above figure. It consists of the 3 normal states `1`, `2`, `3` and the special `End` state. An agent starts in State `1` and roams around this MDP with two actions: move left '`L`' and move right '`R`'. The action nodes represent the chance nodes that represent the outcomes of actions (these are not under our control).

Consider that we know the following about the MDP:

- State Transition Function $\mathbf{T}$:
  - $\mathbf{T}(1, L, 2) = 1$
  - $\mathbf{T}(1, R, 3) = 1$
  - $\mathbf{T}(2, L, 2) = 1$
  - $\mathbf{T}(2, R, 1) = 0.5$
  - $\mathbf{T}(2, R, 3) = 0.5$
  - $\mathbf{T}(3, L, End) = 1$
  - $\mathbf{T}(3, R, End) = 1$
- Reward Function $\mathbf{R}$:
  - $\mathbf{R}(1, L, 2) = 5$
  - $\mathbf{R}(1, R, 3) = 10$
  - Rewards for all other transitions (s, a, s') are 0

(a) (2 points) What is $\mathbf{T}(1, R, 2)$?

> **Solution:** $\mathbf{T}(1, R, 2) = 0$ Since we know that $\mathbf{T}(1, R, 3) = 1$, this means that the agent will always transition from State `1` to State `3` if it takes Action `R`. Thus, $\mathbf{T}(1, R, 2) = 0$, i.e. there is no probability of transitioning from State `1` to State `2` on Action `R`

(b) Ameya is running tabular Q-learning on this MDP. So far, he has learned the following Q-value estimates:

$\hat{Q}(s, a)$	Action L	Action R
State 1	15	10
State 2	4	8
State 3	0	0

The Q-learning agent is now about to select the next action to perform. The agent is currently in state 2. Ameya is using ε-Greedy with an ε value of 0.2.

i. (3 points) What is the probability that the agent chooses the R action?

> **Solution:** R is the best action according to our current Q-values. So with probability 0.8, we will choose it, and otherwise we choose a random action, which has a 50% chance of being R. So the total probability is $\boxed{0.9}$.

ii. (3 points) What is the probability that the agent lands in state 1 at the next timestep?

> **Solution:** If we choose action R, then there's a 50% chance of transitioning to state 1. If we choose action L, it is impossible to go to state 1. So the answer is $.9 \times .5 = .45$.

iii. (5 points) During Q-Learning, does the learner have enough information to compute the answer to part (i)? Explain your answer.

> **Solution:** Yes, this is determined only by the current Q-value predictions and the choice of ε. All of this is known to the learner.

iv. (5 points) During Q-Learning, does the learner have enough information to compute the answer to part (ii)? Explain your answer.

> **Solution:** No, this depends on the underlying transition probabilities of the MDP. This is not known to the learner during Q-Learning.

v. (6 points) Finally, suppose that the agent does the action R and transitions to state 1. How should the Q-value table be updated? Specify which cell(s) need to be updated, and what new value they will have. Use a discount factor of $\gamma = 0.8$ and learning rate of $\eta = 0.25$.

> **Solution:** First, we only will update the cell for state 2 and action R, since that is the state-action pair that we observed at this step.
> The update formula is
>
> $$\hat{Q}(s,a) \leftarrow (1-\eta)\hat{Q}(s,a) + \eta(r + \hat{V}(s')).$$
>
> Note that $r = 0$ because this transition does not receive any immediate reward. We also have that $\hat{V}(s') = 15$ based on the Q-value table. So, we get
>
> $$\hat{Q}(2,R) = .75 \times 8 + .25 \cdot .8 \cdot 15 = 6 + 3 = \boxed{9}.$$

(c) (5 points) For this same scenario, Ameya wants to train a reinforcement learning agent that obtains higher expected reward than tabular Q-Learning. He has the idea to define a policy $\pi(s_{t-1}, s_t)$ which is a function of the current and the past state. Do you think this policy would obtain a higher expected reward? Explain your answer.

> **Solution:** No. By definition of an MDP, the future trajectory of the MDP is described completely by the current state. Thus, past visited states will have no impact on the future of the agent. A policy that uses the past state information in addition to the

future will be no better than a policy that only uses the current state to select an action.

Question 5: Short Response (12 points)

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

(a) (6 points) Consider an attention head in a multi-headed attention layer within a Transformer encoder model. The input is a sequence of vectors $x_1, \ldots, x_T$. Your friend reasons that because x_T is more similar to itself than to any other vector, the attention from x_T will always attend more to x_T than to any other vector. Based on the mathematical formula for multi-headed attention, explain why your friend's reasoning is incorrect.

Solution: This reasoning is incorrect because the attention of x_T is actually determined by the similarity (i.e., dot product) between $W^K x_i$ and $W^Q x_T$. Even if two vectors are similar, the transformation by W^K and W^Q may make them much less similar. So, even though $x_T^\top x_T$ may be large, there is no guarantee that $(W^K x_T)^\top (W^Q x_T)$ will be large.

(b) (6 points) The first problem in the midterm exam involved a real-life machine learning application of estimating the current weight of a fetus based on ultrasound measurements of the baby's width/height. You might be wondering, how did they collect training data for this model? Researchers collected data on various babies' weights when they were born, and paired this with ultrasound measurements taken right before they were born. To get babies who were in a variety of developmental stages, the researchers included many babies who were born prematurely. Identify and describe a spurious correlation in this data that could bias the model's predictions.

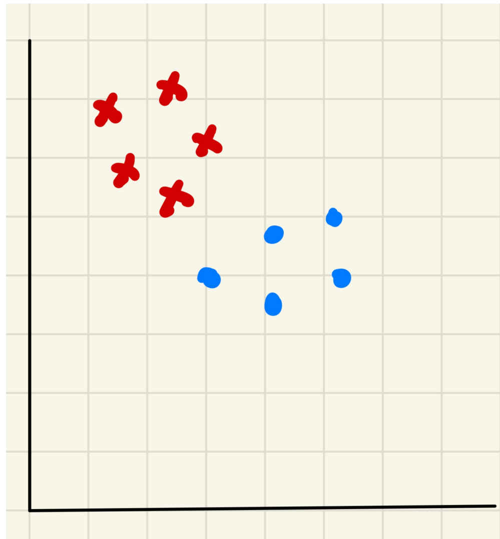
Solution: The spurious correlation is that small babies in the dataset are also the ones who were born prematurely. So, when the model is asked to make a prediction about a baby with small measurements, it will make a prediction as if it were about to be born prematurely. But premature birth could be associated with medical conditions that also affect weight. This would skew the predictions of the model when applied to fetuses who are not going to be born prematurely.

Question 6: Multiple Choice (30 points)

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

(a) (3 points) The picture below shows a set of points with two clusters in crosses and circles. Which algorithm(s) could have generated these cluster assignments?

Choose all that apply.



A. Gaussian Mixture Models
B. k-Means
C. k-Nearest Neighbors
D. Principal Components Analysis

> **Solution:** A and B. Both k-means and GMM can generate this clustering. k-means always learns spherical clusters, and GMM can learn spherical clusters by learning covariance matrices that are close to the identity matrix. The other two are not clustering methods.

(b) (3 points) Wenyang is learning a discriminative model with parameters θ and wants to apply the principle of maximum likelihood estimation. She has a dataset of examples $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$. Which of the following describes what she should do? Choose all that apply.

A. Maximize $\log\left(\sum_{i=1}^{n} p(y^{(i)} \mid x^{(i)}; \theta)\right)$
B. Maximize $\sum_{i=1}^{n} \log p(y^{(i)} \mid x^{(i)}; \theta)$
C. Maximize $\prod_{i=1}^{n} p(y^{(i)} \mid x^{(i)}; \theta)$
D. Minimize $\sum_{i=1}^{n} \log p(y^{(i)} \mid x^{(i)}; \theta)$

> **Solution:** B and C are correct. A is wrong, you do not take the log of the sum but rather the sum of the logs. D is wrong, you minimize the *negative* log likleihood.

(c) (3 points) Which of the following would cause running stochastic gradient descent to be equivalent to normal gradient descent? Choose all that apply.

A. Setting the batch size to the size of the training dataset.
B. Setting the batch size of 1.
C. Setting the number of epochs to the size of the training dataset.
D. Setting the number of epochs to 1.

Solution: A only, this does one update per epoch so it's equivalent to normal gradient descent.

(d) (3 points) Which of the following conditions could be a sensible stopping condition while building a decision tree?

A. Stop if you find the validation error is decreasing as the tree grows

B. Don't split a tree node that has an equal number of sample points from each class

C. Don't split a tree node whose depth exceeds a specified threshold

D. Don't split a tree node if the split would cause a large reduction in the weighted average

Solution: C only, you can set a limit on the depth of the tree. A is wrong because validation error decreasing is a good thing, and means you should keep going. B is wrong because if the node has samples from all different classes, probably you should try to split it up. D does not really make sense.

(e) (3 points) For which of the following is gradient descent is used for optimization? Choose all that apply.

A. Normal equations for Linear Regression

B. Feed-forward Neural Networks

C. k-NN Classifier

D. Gaussian Mixture Models

Solution: B only. Normal equations are a closed-form solution. k-NN does not use any optimization. GMM's use EM, and the M step is done with a closed-form solution, not gradient descent.

(f) For each of the statements below, indicate if they are true for:

A. Bandits only

B. Full reinforcement learning only

C. Both bandits and full reinforcement learning

D. Neither bandits nor reinforcement learning

i. (2 points) The agent's actions can influence what data is observed by the learning algorithm.

i. ____________

Solution: C. In both cases, the actions affect what is observed.

ii. (2 points) The agent's action in the current timestep can influence which action is optimal in the next timestep.

ii. ____________

> **Solution:** B. Only in reinforcement learning can you move from one state to a new state. The optimal action in one state may be different.

iii. (2 points) If the agent takes the same action at timestep 1 and timestep 2, the expected reward received immediately after taking the action is the same.

iii. ____________

> **Solution:** A. This is true in bandits again because there is no variable state, so each action has the same expected outcome at all timestep.

(g) Soumya has a dataset of images $x_1, x_2, \ldots, x_n$. For each image i, he has a label y_i that says if the image is a cat or a dog. In each scenario, which of the following methods would be appropriate? Choose all that apply.

A. Expectation-Maximization with Gaussian Mixture Models
B. k-Nearest Neighbors
C. Linear Regression
D. Logistic Regression
E. Policy Gradient
F. Upper Confidence Bound Algorithm

i. (3 points) Soumya wants to find out whether there are distinct subtypes of cats within his group of cat pictures.

i. ____________

> **Solution:** A. This is a clustering problem.

ii. (3 points) Soumya wants to determine whether a new image x is a cat or a dog.

ii. ____________

> **Solution:** Both B and D. This is a classification problem.

iii. (3 points) Given a new image x, Soumya wants to estimate the posterior probability of x being a cat.

iii. ____________

> **Solution:** D only. Only logistic regression produces probability predictions; k-NN does not.

[This page provides extra space for answers]

[This page provides extra space for answers]