CS 189/289A Introduction to Machine Learning
Fall 2023 Jennifer Listgarten, Jitendra Malik

Midterm

- Please do not open the exam before you are instructed to do so.
- **Electronic devices are forbidden on your person**, including cell phones, tablets, headphones, and laptops. Leave your cell phone off and in a bag; it should not be visible during the exam.
- The exam is closed book and closed notes except for your one-page 8.5×11 inch cheat sheet.
- You have 1 hour and 50 minutes (unless you are in the DSP program and have a larger time allowance).
- Please write your initials at the top right of each page after this one (e.g., write "JD" if you are John Doe). Finish this by the end of your 1 hour and 50 minutes.
- Mark your answers on the exam itself in the space provided. Do **not** attach any extra sheets.
- For multiple choice questions, fill in the bubble for the single best choice.
- For short and long answer questions, write within the boxes provided. If you run out of space, you may use the last four pages to continue showing your work.
- **The last question is for CS289A students only**. Students enrolled in CS189 will **not** receive any credit for answering this question.

Your Name	
Your SID	
Name and SID of student to your left	
Name and SID of student to your right	
Doing anything fun this weekend?	
Favorite ML algorithm?	

○ CS 189

○ CS 289A

This page intentionally left blank.

initial here

1 Multiple Choice

For the following questions, select the **single best response**. Each question is worth 1.5 points.

1. Which of the following is a problem with the sigmoid activation function, in the context of deep neural networks?
   - ◯ Sigmoid is prone to vanishing gradients at extreme values.
   - ◯ Sigmoid can take on negative values.
   - ◯ Sigmoid is non-linear, which provides less representation power.
   - ◯ Sigmoid is numerically unstable when the input is large.
2. Which of the following is **not** a component/feature of standard Transformer models?
   - ◯ Masked decoding, which prevents attention lookups into the future.
   - ◯ Transformer model training can be highly parallelized.
   - ◯ Multi-head attention, which allows for attending to different parts of the sequence (e.g. long-range vs. short-range dependencies).
   - ◯ The runtime complexity of attention is $O(n \log n)$, where n is the input length.
3. Consider a multivariate Gaussian distribution with covariance matrix $\Sigma = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$. Which of the following correctly corresponds to the direction of the **major** axis of the Gaussian's isocontours?
   - ◯ $\begin{bmatrix} \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$
   - ◯ $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$
   - ◯ $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$
   - ◯ $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$
4. Suppose you train a logistic regression model with **cross-entropy loss**. Assume that you're updating the model using gradient descent with a sufficiently low learning rate. Which of the following are valid ways to initialize the parameters? Assume that numerical stability is not an issue (i.e. when running the model, you don't pass in an invalid argument to a function, such as 0 to log).
   - ◯ Initialize all parameters with 0's
   - ◯ Initialize all parameters with 1's
   - ◯ Randomly initialize parameters
   - ◯ All of the above

initial here

5. Suppose we want to apply dimensionality reduction to a high-dimensional dataset X. Let $X = U\Sigma V^T$ be the singular value decomposition (SVD) of X. Which of the following regarding PCA is **false**?

   ◯ The principal components must be orthogonal to each other.

   ◯ The principal components are given by the eigenvectors of X.

   ◯ Multiplying the first k principal component scores $U_k\Sigma_k$ (where U_k is the first k columns of U and Σ_k is the top-left $k \times k$ entries of Σ) by their corresponding principal axes V_k^T (where V_k is the first k columns of V) gives us a matrix of rank k.

   ◯ The variance along the ith principal component axis is given by σ_i^2.

6. Which of the following is **not** a feature of a standalone convolution layer? Assume that there is no pooling layer afterwards.

   ◯ Local Connectivity: The convolution layer assumes that local regions in the input are more relevant for learning features.

   ◯ Parameter Sharing: The same set of weights is used across different parts of the input in the convolution layer.

   ◯ Translational Invariance: The convolution layer is designed to produce an output that is insensitive to translations in the input.

   ◯ Channel-wise Feature Learning: In multi-channel inputs (like color images), each filter can learn features that are channel-specific.

7. Consider adding the elastic net regularization term $\lambda_1\|w\|_2^2 + \lambda_2\|w\|_1$ to linear regression. If we want our weights to be smaller, but dislike having many completely zero weights, what is most likely to be the best choice?

   ◯ Increase λ_1

   ◯ Increase λ_2

   ◯ Decrease λ_1

   ◯ Decrease λ_2

8. PCA and t-SNE are techniques often used for representing high-dimensional data in a lower-dimensional space. Which of the following is true regarding these two techniques? Assume that PCA is run without a basis expansion (e.g. a polynomial basis expansion).

   ◯ Both PCA and t-SNE can accurately capture non-linear relationships in the data.

   ◯ t-SNE aims to minimize the pairwise distances in the data whereas PCA aims to maximize the variance of the data.

   ◯ Both PCA and t-SNE have convex optimization problems.

   ◯ t-SNE aims to maximize the pairwise distances in the data whereas PCA aims to minimize the variance of the data.

initial here

9. Consider the optimization problem

$$\min_w \ \left\|Xw - y\right\| + g(w)$$

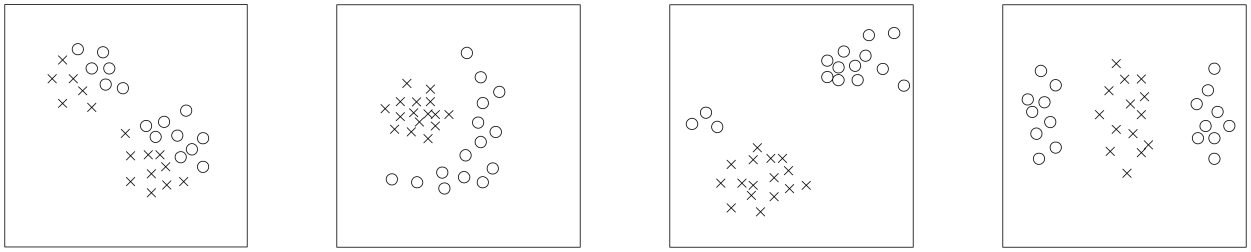
for some $g(w)$. We assume that $P(y|x; w) = N(w^\top x, \sigma^2)$. The minimizer w^* of this problem can be thought of as a MAP solution with which prior $P(w)$?

- ◯ $\frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-g(w)^2}{2\sigma^2}\right)$
- ◯ $\frac{\exp(-g(w))}{\int_{w'} \exp(-g(w'))\, dw'}$
- ◯ $g(w) \exp(-\|w\|)$
- ◯ $\frac{1}{2\sigma} \exp\left(\frac{-|g(w)|}{\sigma}\right)$

10. When training a neural network with attention blocks, which of the following would most likely cause outputs to become NaNs? Assume numerical instability is not an issue in the inputs.

- ◯ Having large variation in your attention scores. Recall attention scores are dot products between projected keys and queries.
- ◯ Not normalizing keys and queries before taking their dot product. This means keys and queries may not have norm 1.
- ◯ Apply a mask to all of our keys. Masking prevents the model from attending to certain keys.
- ◯ Randomly initializing our neural network weights with i.i.d. random variables that have unbounded expectation.

11. Which of the following k-means cluster assignments could be a possible result after running k-means to convergence for 2 clusters?



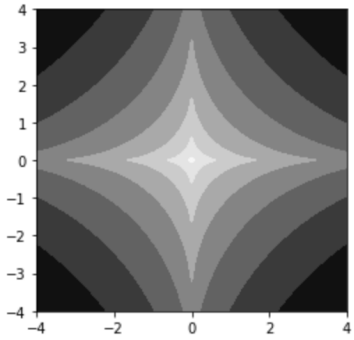
(a) (b) (c) (d)

- ◯ (a)
- ◯ (b)
- ◯ (c)
- ◯ (d)

initial here

2 Short Answer

1. In class you learned about regularizing linear regression with an L_1 penalty (LASSO). Now, instead consider an $L_{0.5}$ penalty on the weights. A plot of the isocontours of the $L_{0.5}$ norm is shown below:



How will the sparsity of the $L_{0.5}$ penalized linear regression compare to that of LASSO? Select the best answer choice **and** explain your answer.

◯ $L_{0.5}$ will have more sparse solutions than LASSO.

◯ $L_{0.5}$ will have less sparse solutions than LASSO.

2. We have a dataset with binary labels $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$ where $y_n = \{0, 1\}$. Write the formula for the binary cross entropy loss between a model's predicted probabilities for the positive class $\{p_\theta(y = 1|x_n)\}_{n=1}^N$ and the true labels $\{y_n\}_{n=1}^N$. Assume that the loss is for the whole dataset. Please box your final answer.

initial here

3. Suppose you have two coins. Coin A has probability p of landing heads, and coin B has probability $3p$ of landing heads. Suppose you flip both coins and get the following result:
   Coin A: T
   Coin A: H
   Coin B: H
   Coin B: H
   Coin A: T
   Coin B: H

   What is the maximum likelihood estimate for p? Please box your final answer.

4. For your latest Transformer model, you have devised this positional encoding scheme:

$$PE_{(x,i)} = \left(\frac{x \pmod{189}}{189} \right)^{i/d_{model}}$$

   where x is the position, i is the index of the feature dimension, and d_{model} is the total number of dimensions in an input embedding. This value is added to index i of the x-th token's embedding vector.

   Describe in one or two short sentences what problems you may encounter with this encoding, as well as what inputs you would encounter them on.

initial here

5. Suppose that $X \in \mathbb{R}^3$ is a random vector with a multivariate Gaussian distribution that can be written as $AZ + \mu$, where $Z \sim N(0, I_3)$, $\mu = [1, 1, 1]^T$ and $A = \begin{bmatrix} 1 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$. Compute the correlation matrix C corresponding to X. In a correlation matrix, entry $C_{ij} = \frac{\text{cov}(X_i, X_j)}{\sqrt{\text{var}(X_i)\text{var}(X_j)}}$, where X_i, X_j are the i and jth entries of X. Please box your final answer.

initial here

6. In this problem, we consider a solution to the vanishing gradient problem: residual connections.

   To examine this phenomenon, let's consider a slightly different residual layer than what was presented in lecture. Our layer processes vector inputs $x_{in} \in \mathbb{R}^d$. Let $W \in \mathbb{R}^{d \times d}$ and $b \in \mathbb{R}^d$. Mathematically, a layer is represented as:

   $$x_{out} = x_{in} + \max(Wx_{in} + b, 0)$$

   Compute the Jacobian $\frac{\partial x_{out}}{\partial x_{in}}$ for this residual block and explain why this prevents the gradients from dying at this layer. For this problem, you may assume that $Wx_{in} + b \neq 0$, and therefore we don't need to worry about the derivative at the non-differentiable point. Box your answer for $\frac{\partial x_{out}}{\partial x_{in}}$.

   Hint: Start by deriving the formula for entry (i, j) of the Jacobian: $\frac{\partial x_{out,i}}{\partial x_{in,j}}$. Then, assemble the Jacobian matrix.

7. Say we have four points: (0, 1), (0, -1), (1, 1), (1, -1). Using PCA, find the first principal direction.

initial here

8. Let $f(x) = Wx$ be parameterized by a $d \times d$ matrix W. Suppose a data point $x \in \mathbb{R}^d$ has a label $y \in \mathbb{R}^d$ associated with it. We use the L_2 loss function $L(W) = \|f(x) - y\|_2^2 = \|Wx - y\|_2^2$ to describe the error between our prediction $f(x)$ and the ground truth label y. Recall that a gradient update step for W looks like $W^{(t+1)} = W^t - \alpha * \nabla_W L(W^t)$, where α is our learning rate. Let us now consider the problem of selecting an optimal α. We can do this by solving the following optimization problem:

$$\alpha^\star = \min_\alpha \; L(W - \alpha \nabla_W L(W))$$

What is $\alpha^\star$ in this case? Please box your final answer for $\alpha^\star$.

Hint 1: Recall from Homework 1 that $\nabla_W L(W) = 2(Wx - y)x^\top$.

Hint 2: Performing a change of variables $z = Wx - y$ may help to simplify algebra.

initial here

9. Consider the function $f : \mathbb{R}^d \mapsto \mathbb{R}^d$ whose components are given by

$$f_i(x) = \frac{x_i^2}{\sum_{k=1}^{d} x_k^2}$$

for $i = 1, \ldots, d$ and $x \in \mathbb{R}^d$.

(a) Find an expression for $\frac{\partial f_i}{\partial x_j}$. *Hint: It may help to consider two cases: $i = j$ and $i \neq j$.*

(b) We can see that $0 \leq f_i \leq 1$ for each $i = 1, \ldots, d$ and $\sum_{i=1}^{d} f_i = 1$. Thus, f normalizes the input vector x to a probability distribution, just like Softmax does! That said, what is one way in which our function f graphically differs from the Softmax function?

initial here

3 PCA Fundamentals

Principal Component Analysis (PCA) is a commonly used technique for dimensionality reduction. Suppose we are given a set of data points $\mathcal{D} = \{x_n \in \mathbb{R}^D : n = 1, ..., N\}$. PCA can be viewed as a change of basis that represents each data point x_n as a weighted combination of a new set of basis functions $w_1, ..., w_L \in \mathbb{R}^D$, where the weights are given by $z_n \in \mathbb{R}^L$. That is, each x_n is approximated by $\sum_{k=1}^{L} z_{nk} w_k$. Here we explore one of the many aspects of this change of basis operation.

(a) As a warm up exercise, show that the sample covariance matrix Σ is PSD, where

$$\Sigma = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mu)(x_n - \mu)^T$$

and $\mu = \frac{1}{N} \sum_{n=1}^{N} x_n$ is the sample mean.

(b) One way to view the change of basis that PCA performs is that it is the optimal solution that minimizes the average reconstruction error of the data:

$$\mathcal{L}(W, Z) = \frac{1}{N} \left\| X - ZW^T \right\|_F^2 = \frac{1}{N} \sum_{n=1}^{N} \|x_n - W z_n\|^2$$

where $X \in \mathbb{R}^{N \times D}$ is the data matrix, $W \in R^{D \times L}$ is the matrix of "latent factors", and $Z \in \mathbb{R}^{N \times L}$ is the matrix of "latent vectors".

Suppose we wish to minimize the error, subject to the constraint that W is an orthogonal matrix. We will first find the optimal Z and then find the optimal W when $L = 1$. Express and expand out the reconstruction error as a function of w_1 and its associated coefficients $z_1 = [z_{11}, ..., z_{N1}] \in \mathbb{R}^N$, and show the optimal solution for each entry of z_1, z_{n1}, is $z_{n_1} = w_1^T x_n$.

initial here

(c) Given the optimal solution of z_n, show the optimal solution for w_1 corresponds to finding the largest eigenvalue of some matrix. What is the matrix in the case in which the data is centered, i.e. the sample mean $\mu = 0$? **Hint**: This will require using constrained optimization.

initial here

(d) It is often said that the optimal solution for w_1 that minimizes the reconstruction error corresponds to maximizing the sample variance of the projected data. Show that this is the case when the data is centered. Give a brief explanation for what might go wrong when the data is not centered.

initial here

4 Motivating Logistic Regression

In class, you learned that a common justification for logistic regression is the fact that Gaussian class-conditional probability densities result in a posterior probability $p(y|x)$ that takes the form of a logistic function with a linear argument, assuming the Gaussians all have the same variances. In this question, we will explore the case when this assumption does not hold true.

Consider the case in which we perform binary classification for two classes $y = 0$ and $y = 1$. Suppose that we know both classes have Gaussian class-conditionals, i.e. $P(X \mid Y = 0) \sim N(\mu_1, \sigma_1)$ and $P(X \mid Y = 1) \sim N(\mu_2, \sigma_2)$. For simplicity, we assume that the covariance matrix for each class is diagonal with the same variance along each dimension (but still with differing variances between the two classes). Recall the Gaussian PDF formula:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

We also know the two classes have prior probabilities $P(Y = 0) = \pi_1$ and $P(Y = 1) = 1 - \pi_1$.

(a) Show that the posterior distribution $P(Y = 0 \mid X)$ is a logistic function with a quadratic argument $\alpha\|x\|^2 + \beta^T x + \gamma$. What are α, β, and γ? Box your answers for each.

initial here

(b) Consider the decision boundary, i.e. the point at which $P(Y = 1 \mid X) = P(Y = 0 \mid X)$. Assume that $\pi_1 = 0.5$, i.e. the prior probabilities of the two classes are the same. How does the shape of the decision boundary compare qualitatively when the two class variances are the same vs. not the same? 1-2 sentences is enough.

initial here

5 Laplacian Discriminant Analysis

You are playing a fun game with Alice and Bob: in each turn, either Alice or Bob sends you a number, and you have to guess who sent it!

You know the following about Alice and Bob's behavior:

- The probability that Alice sends the number is π_A and the probability that Bob sends the number is $\pi_B = 1 - \pi_A$.
- Each person generates numbers from an independent Laplace distribution, with parameters (μ_A, b) for Alice, and (μ_B, b) for Bob. (Note that the second parameter is the same for both.)
- $\mu_A > \mu_B$.

The Laplace distribution is a continuous distribution. It is parameterized with two values (μ, b), and its probability density function is as follows:

$$f(x) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$$

(a) You receive a number x and you want to determine if it's more likely to be from Alice or Bob. What kind of classifier are you trying to build?

◯ Generative

◯ Discriminative

(b) Compute the probability that the number is from Alice. Box your final answer.

Your answer should be expressed in the form $\frac{1}{1+\exp(-z)}$, where z is some expression that may (or may not) be in terms of $\pi_A, \pi_B, \mu_A, \mu_B, x, b$, and constants.

initial here

(c) You decide to play a new game. All the conditions stay the same, except this time, Alice and Bob both send you a number, and you have to match the two numbers to their senders! You receive two numbers: $x_1 > x_2 > \mu_A$. To maximize the probability of being correct, which one should you guess to be Alice's number? Justify your answer.

initial here

6 Machine Unlearning: Linear Regression

We often care about validating our model on data that wasn't used to train it. We've seen that K-fold cross validation can be a powerful way to ensure that our models are robust to dropping out chunks of the data.

In this question, we're going to engage with a version of K-fold validation called *leave-one-out* validation. Let's say we have a design matrix $X \in \mathbb{R}^{n \times d}$ and a corresponding set of labels $Y \in \mathbb{R}^n$. As the name suggests, we train n different models leaving out a single data point each time and then test our model on this held out datapoint.

Typically, we denote the model fit on all the data except the ith point with a subscript $[i]$, so the model trained without (X_i, Y_i) is $f_{[i]}$. The leave-one-out error is therefore defined as

$$J_{loo}(X, Y, f) = \frac{1}{n} \sum_i \text{loss}(f_{[i]}(X_i), Y_i)$$

This kind of validation is quite cumbersome for most models, but we'll see for linear models and the squared error loss it can actually be computed efficiently. We'll then explore the consequences of this fact for updating linear models with new data.

(a) First compute $(X_{[i]}^T X_{[i]})^{-1}$ in terms of $(X^T X)^{-1}$ and X_i, the ith row of X. You may use the Sherman-Morrison identity without proof, which is: Let $A \in \mathbb{R}^{k \times k}$ and $u, v \in \mathbb{R}^k$, then

$$\left(A + uv^\intercal\right)^{-1} = A^{-1} - \frac{A^{-1}uv^\intercal A^{-1}}{1 + v^\intercal A^{-1}u}.$$

Hint: Try writing $X^T X$ in outer product form.

initial here

(b) Write out the closed form solution for $\beta_{[i]}$, the coefficients of a linear model trained without the ith datapoint. Your answer should be in terms of β (the coefficients fitted on all the data), X_i, $e_i = X_i^T(X^TX)^{-1}X^TY - Y_i$, $(X^TX)^{-1}$, and $h_i = X_i^T(X^TX)^{-1}X_i$.

Hint: Begin by writing out the closed form solution for $\beta_{[i]}$, then plugging in part (a).

(c) Finish by computing the leave-one-out error $e_{[i]}$ in terms of the original training error $e_i = \hat{Y}_i - Y_i$ and h_i.

initial here

(d) Use parts (a) and (b) to show how you could add a single datapoint to an existing OLS fit given just the new point X_{n+1}, an updated covariance matrix $(X_{n+1}^T X_{n+1})^{-1} = (X^T X + X_{n+1} X_{n+1}^T)^{-1}$, and the original fit β.

initial here

7 Xavier Initialization for Neural Networks (CS 289A Only)

When you optimize a neural network's weights using Gradient Descent, you must start with an initial guess for your model parameters that is then iteratively updated during training. Before the advent of modern deep learning techniques like Batch Normalization, the way in which these parameters were initialized could significantly impact model training: more often than not, poor model performance was attributed to poor weight initialization. In this problem, we will think about a scheme for initializing model weights that leads to stable training.

Consider a feed forward neural network with L layers. Assume that we use the activation function $\sigma(\cdot)$ after each layer. Suppose the input to layer l is the $n^{[l-1]}$-dimensional vector $x^{[l-1]}$ and let this layer have an $n^{[l]} \times n^{[l-1]}$ weight matrix $W^{[l]}$ and $n^{[l]}$-dimensional bias vector $b^{[l]}$. Then, the forward propagation equations for layer l are

$$z^{[l]} = W^{[l]}x^{[l-1]} + b^{[l]}$$
$$x^{[l]} = \sigma(z^{[l]})$$

By convention, we let $x^{[0]}$ and $x^{[L]}$ be the input to and the output of the whole network, respectively. We denote $x_k^{[l]}$ (consequently $b_k^{[l]}$ and $z_k^{[l]}$) to be the kth component of $x^{[l]}$ (consequently $b^{[l]}$ and $z^{[l]}$). Similarly, we denote $W_{ij}^{[l]}$ to be the ijth component of $W^{[l]}$.

(a) Suppose $\sigma(x) = x$, i.e., the identity activation function, is used throughout the neural network. For the sake of simplicity, assume that each $b_k^{[l]}$ is initialized to 0. Furthermore, assume that

- Each $x_k^{[0]} \sim X$ is independent and identically distributed, where X is a distribution with $\mathbb{E}[X] = 0$.
- Each $W_{ij}^{[l]} \sim W_l$ is independent and identically distributed, where W_l is a distribution with $\mathbb{E}[W_l] = 0$.
- All $x_k^{[0]}$ and $W_{ij}^{[l]}$ are mutually independent.

Find an expression for $\text{Var}(x_k^{[1]})$ in terms of $n^{[0]}$, $\text{Var}(W_1)$ and $\text{Var}(X)$.

Hint: you may use the following fact without proof: if Y and Z are independent, zero-mean random variables, then $\text{Var}(YZ) = \text{Var}(Y)\,\text{Var}(Z)$.

initial here

(b) Show that $\mathbb{E}[x_k^{[1]}] = 0$.

(c) It turns out each $x_k^{[1]}$ (for $1 \leq k \leq n^{[1]}$) has the same zero-mean distribution. In fact, by induction, one can show that each $x_k^{[l]}$ (for $1 \leq k \leq n^{[l]}$) has the same zero-mean distribution, which we denote by X_l, and that each $x_k^{[l]}$ is mutually independent of each $W_{ij}^{[l]}$. Then, operating under the same assumptions as part (a), find an expression for $\text{Var}(X_l)$ in terms in terms of $n^{[l-1]}$, $\text{Var}(W_l)$ and $\text{Var}(X_{l-1})$.

Hint: you may use the following fact without proof: if Y and Z are independent, zero-mean random variables, then $\text{Var}(YZ) = \text{Var}(Y)\,\text{Var}(Z)$.

initial here

(d) Show that setting $\text{Var}(W_l) = \frac{1}{n^{[l-1]}}$ will yield $\text{Var}(X_L) = \text{Var}(X)$, i.e., each component of the network input and output will have the same variance. This weight initialization where

$$W_{ij}^{[l]} \sim \mathcal{N}\left(0, \frac{1}{n^{[l-1]}}\right)$$

is called the *Xavier* initialization.

(e) What happens when $\text{Var}(W_l) > \frac{1}{n^{[l-1]}}$ instead? What about $\text{Var}(W_l) < \frac{1}{n^{[l-1]}}$?

initial here

You may use this page to show extra work. Clearly mark your work with the problem number here, and also mention in the problem-specific box that your work is continued here.

initial here

You may use this page to show extra work. Clearly mark your work with the problem number here, and also mention in the problem-specific box that your work is continued here.

initial here

You may use this page to show extra work. Clearly mark your work with the problem number here, and also mention in the problem-specific box that your work is continued here.

initial here

You may use this page to show extra work. Clearly mark your work with the problem number here, and also mention in the problem-specific box that your work is continued here.