Stanford CS 229, Spring 2021 Midterm

The midterm is open-book, closed-collaboration, and subject to the Honor Code. You may:

- consult inanimate materials or resources, including the course notes and reference materials online, as long as they do not violate the stipulations below. If you refer to any online sources as part of obtaining your answer, you must cite them. If you use any results from online sources (outside of course materials), you must include a proof or justification as applicable.
- cite without proof any result from lectures or lecture notes, unless otherwise stated.

You may not:

- talk to, consult, or collaborate with anyone about the exam.
- post questions on any public or private forums (e.g. Stack Overflow, Chegg, Slack, etc.).
- enter exam questions into any software, apps, or websites.
- access resources that directly explain how to answer questions from the actual exam. This includes equation solution finders, and software, apps, or websites where other people have provided answers to specific questions (e.g., Chegg's Expert Q&A, etc.).

To be fair to students taking the exams in multiple time zones, we cannot offer clarifications during the exam. In an ambiguous situation, state any assumptions you need to and answer as best you can.

Gradescope will close your submission window after 3 hours. We will not accept any submissions after that time, so please make sure you leave enough time to submit! Since you can resubmit (we will only use your last submission), we highly recommend submitting a draft around the 2 hour 30 minute mark just in case.

Good luck! We know you've been working hard, and we all want you to succeed!

Question	Points
1 True or False	/30
2 Generalized Discriminant Analysis	/10
3 Role Play!	/20
4 Classification with the Gaussian distribution	/25
5 Orthogonal Residuals	/20
Total	/105

1. **[30 points] True or False**

   For each question, answer True or False, and **briefly and convincingly justify your answer** in a sentence or two.

   (a) [3 points] Given its simplicity and linear structure, Linear Regression cannot overfit.

   (b) [3 points] The output of a Support Vector Machine model (for each example) is the probability that the input example belongs to class 1 (instead of class 0).

   (c) [3 points] When we want to train kernel method based learning algorithms (that will be capable of making predictions when deployed on new examples), we are *necessarily required* to save or memorize examples (or a subset of examples) from the training set.

   (d) [3 points] Overfitting and underfitting can be a problem only in supervised learning settings (e.g. classification or regression), and are not a problem in / not applicable to unsupervised learning settings.

   (e) [3 points] By choosing a suitably crafted loss function, any neural network training objective can be made convex in the model parameters.

(f) [3 points] Suppose a 2-dimensional labelled dataset is perfectly linearly separable into two classes. Then running k-means (ignoring the labels) with exactly two centroids, with the centroids initialized to random points (one from each class), is guaranteed to converge in a way such that each of the two clusters will have points from only one class each.

(g) [3 points] Just like how EM algorithm (on unlabelled data) can have multiple globally optimal solutions, the GDA model (on labelled data) can also have multiple global solutions.

(h) [3 points] For an always-positive random variable X over some event space Ω, it is always the case that $\mathbb{E}(\log X) \leq \log \mathbb{E}(X)$.

(i) [3 points] While the EM algorithm solution is dependent on initialization, K-means on the other hand converges to the same solution no matter how we initialize the centroids.

(j) [3 points] Suppose that we applied regularization to the bias terms of a neural network as well as the weight matrices. E.g., using the example of the MNIST problem on Homework 2, suppose our loss included the term $\lambda(||W^{[1]}||^2 + ||W^{[2]}||^2 + ||b^{[1]}||^2 + ||b^{[2]}||^2)$ instead of the usual $\lambda(||W^{[1]}||^2 + ||W^{[2]}||^2)$.

Then this would tend to *increase* the bias (in the sense of bias/variance tradeoffs) of the model.

2. **[10 points] Generalized Discriminant Analysis**

   In this problem we will see that the Gaussian Discriminant Analysis (GDA) algorithm can be generalized to any distribution in the Exponential Family to obtain a linear classifier (linear in the sufficient statistics).

   Let us denote the general Exponential Family distribution parametrized by the natural parameter η as $\text{ExpFam}(\eta)$. Recall that the exponential family distribution has density (or probability mass function if discrete):

$$p(x;\eta) = b(x)\exp\{\eta^\top T(x) - a(\eta)\},$$

   where $b(x)$ is called the base measure, $T(x)$ is the sufficient statistic, and $a(\eta)$ is the log-partition function.

   Now suppose that our model is described as follows:

$$\begin{aligned} y &\sim \text{Bernoulli}(\phi) \\ x \mid y = 0 &\sim \text{ExpFam}(\eta_{(0)}) \\ x \mid y = 1 &\sim \text{ExpFam}(\eta_{(1)}) \end{aligned}$$

   where ϕ is the parameter of the class marginal distribution, and $\eta_{(0)}$ and $\eta_{(1)}$ are the class specific parameters for the distribution over input x given $y \in \{0, 1\}$.

   Derive an exact formula for $p(y = 1 \mid x)$ from the terms defined above, and also show that the resulting classifier has a linear decision boundary in $T(x)$.

3. **[20 points] Role Play!**

   You are the CEO of a data consulting firm. Your VP is working with a real estate client who is trying build a house price predictor. The client's speciality is in collecting extremely detailed and high quality data about properties from a variety of sources and combining them. They have created a unique dataset of all houses in the small town they operate in.

   Each example in the dataset corresponds to a house and has $d = 1000$ features per house. The features are all numeric, e.g., number of floors, size in square feet, or 1 vs. 0 for the presence or absence of a pool. Yet, because this is a small town, the number of examples in the dataset is just $n = 100$. The client is worried about this disparity and has proposed several ideas to your team.

   Your VP has commented on the client's ideas. Consider each of the four ideas independently. For each of your VP's comments, say whether it is CORRECT (i.e. completely correct) or WRONG (i.e. at least some aspect is wrong), and briefly and convincingly explain your decision (in 1-2 sentences). To receive full credit for an answer of WRONG, we expect you to point out a significant issue and not e.g. an edge case that involves additional assumptions. You only need to identify a single significant issue.

   Just to be clear: you only need to determine whether the VP's comments are correct. You do not need to comment on the quality of the client's idea.

   Ideas begin on the next page...

(a) [5 points] **Client's Idea:** Simulate additional examples by creating random convex combinations of the existing examples, so that the size of the dataset of new simulated examples plus the original examples (900 + 100) equals the number of features (1000). Then use this to fit a least-squares linear regression model.

(Note: A convex combination is a linear combination in which all coefficients are non-negative and sum to 1. The new y value for a convex combination is the same set of coefficients applied to the individual y values.)

VP's comments: We can use the Normal Equations here. Computing the inverse of $X^\top X$ might be slow since X is 1000×1000, but once we have that, we can multiply it by $X^\top y$ to get a fit that minimizes the least-squares error for our combined dataset. However, we could get different fits depending on which random examples we create.

Your assessment of the VP's comments:

(b) [5 points]

Client's Idea: Keep the data as is, and train a least-squares linear regression model, but then save only the 100 features whose weights have the largest absolute values. Then use only that subset of features to train a new model.

VP's comments: Given that the client wants to save only 100 features, the ones with the largest absolute weights are the best choice, since they should have the most explanatory power. Pruning away the remaining features could also help to reduce overfitting.

Your assessment of the VP's comments:

(c) [5 points]

Client's Idea: Even if we have too many features, deep learning can magically figure out which ones are useful, right? Train a neural network with multiple large hidden layers, using ReLUs as the activation functions. But I'm worried about reproducibility. Investors won't like hearing that the weight matrices and bias vectors were initialized with random Gaussian noise.

VP's comments: If the client is worried about that, we can initialize all of the elements of all of the weight matrices and bias vectors to 1. After enough epochs of forward propagation and back propagation, the model should converge to a fit of comparable quality to the one we would have gotten with the random Gaussian noise initialization.

Your assessment of the VP's comments:

(d) [5 points]

Client's Idea: What if we just don't worry about the fact that $d >> n$? Keep the data as is, and use full-batch gradient descent to fit a least-squares linear regression model. Declare convergence when the L_1 norm of the weights changes by no more than a very small value like 10^{-10}.

VP's comments: Because $d >> n$, there may be many optimal fits, so the model will keep iterating forever and its weights will continue to get larger and larger. It will never truly converge unless we force it to do so artificially by e.g. decreasing the learning rate over time.

Your assessment of the VP's comments:

4. **[20 points] Classification with the Gaussian distribution**

In class, we have seen the probabilistic interpretation of linear regression, where we posit

$$y = \theta^\top x + \epsilon$$

where $x \in \mathbb{R}^d$ is the input, $\theta \in \mathbb{R}^d$ are the model parameters, and $\epsilon \sim \mathcal{N}(0,1) \in \mathbb{R}$ is a random variable that takes on values per example distributed according to a standard normal distribution.

In this problem we will repurpose this model in a binary classification setting in the following way:

$$y = \begin{cases} 1 & \text{if } \theta^\top x + \epsilon \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

In the following sub-problems, we will derive the log-likelihood for a model with the above assumptions, as well as the corresponding update rule for a learning algorithm to udpate the model parameters iteratively.

(a) [8 points] As a first step in deriving the log-likelihood of the above model, what is $p(y = 1 \mid x; \theta)$? You may express the answer in terms of Φ, the cumulative distribution function of the standard normal distribution. Recall that the CDF of a distribution evaluated at a given value is the combined probability assigned to all values less than or equal the given value, i.e. $\Phi(x) = P(X \leq x)$ where $X \sim \mathcal{N}(0,1)$.

(b) [2 points] Using the result from the previous sub-problem, derive an expression for $p(y = 0 \mid x; \theta)$.

(c) [8 points] Now we have expressions for both $p(y = 1 \mid x; \theta)$ and $p(y = 0 \mid x; \theta)$. Given a dataset $\{x^{(i)}, y^{(i)}\}_{i=1}^{n}$ where $x^{(i)} \in \mathbb{R}^d$ and $y^{(i)} \in \{0, 1\}$, write out the log-likelihood of the model parameters $\ell(\theta)$ using a similar approach as we used in logistic regression.

Hint: Treat y as a Bernoulli random variable with parameter whose expression is the solution of $p(y = 1 \mid x; \theta)$.

(d) [7 points] Based on the log-likelihood arrived at in the previous sub-question, derive the gradient ascent based update rule for the model parameters θ.

Note: Your update rule expression may involve ϕ, the standard normal probability density function.

5. **[20 points] Orthogonal Residuals**

In this problem we will explore properties of the *errors* of linear models on the training data. To analyze these errors, we will define something called the *residual*, denoted r, which is the difference between the label y and the predicted value $\hat{y}$:

$$r = y - \hat{y}.$$

In vector notation, let X represent the full training set (one row per example), Y the vector of labels corresponding to X, $\hat{Y}$ the vector of predicted values by the model, and $R = Y - \hat{Y}$ the corresponding vector of residuals. In the following sub-problems we will explore the relationship between the residuals for linear models, specifically that the residuals R are orthogonal to the column span of the inputs X. Recall that two vectors $\vec{a}$ and $\vec{b}$ are orthogonal to each other if $\vec{a}^\top \vec{b} = 0$, and that $\vec{a}$ is orthogonal to a column span if $\vec{a}$ is orthogonal to *every* vector in that span. Intuitively, this means that the remaining errors, i.e. the residuals, cannot be explained by the input features anymore since they are exactly orthogonal.

(a) [10 points] First we will consider linear regression, with real valued labels $Y \in \mathbb{R}^n$, input features $X \in \mathbb{R}^{n \times d}$ (one example per row), estimated parameter $\hat{\theta} \in \mathbb{R}^d$, and predicted values $\hat{Y} = X\hat{\theta} \in \mathbb{R}^n$. For simplicity we will assume the bias term is already included in $\hat{\theta} \in \mathbb{R}^d$, and that $\hat{\theta}$ is estimated by minimizing the squared error:

$$\hat{\theta} = \arg\min_{\theta \in \mathbb{R}^d} \|Y - X\theta\|_2^2.$$

The residuals, as defined already, are just $R = Y - \hat{Y} \in \mathbb{R}^n$. Show that $R \perp X$, i.e. $R^\top X = \vec{0}$.

Hint: use the normal equations for a short proof.

(b) [10 points] We will now generalize the previous result to Generalized Linear Models (GLMs) of which linear regression is a specific form. We will assume that the GLM model is trained with the *maximum likelihood* estimation approach. Suppose now $y \sim \text{ExpFam}(\eta; a, b)$, where $p(y) = b(y)\exp\{\eta^\top y - a(\eta)\}$ (assuming $T(y) = y$). Notice that y is not necessarily real valued anymore, and could be $y \in \{0, 1\}$ when ExpFam corresponds to Bernoulli, or $y \in \mathbb{N}_+$ when ExpFam corresponds to Poisson, etc.

Recall the standard assumptions of GLM, that $\hat{y} = g(\theta^\top x)$ where g is the *link function*. In vectorized notation this can be written as $\hat{Y} = g(X\hat{\theta})$. The residuals are still defined the same way, i.e. $R = Y - \hat{Y}$, using this definition of $\hat{Y}$.

Show that, even in this generalized setting involving the non-linearity g, the orthogonality of residuals and the inputs still hold: $R \perp X$ (i.e. $R^\top X = 0$).

Hint: Start with the gradient condition necessary for a model to have completed training with the maximum likelihood objective.

That's all! Congratulations on completing the midterm exam!