CS 229 Midterm (Fall 2023)

Read the following points before starting the exam:

- This exam is open book (notes, homework, and solutions) but closed to the Internet.
- You have three (3) hours to complete the exam.
- You may cite (without proof) any result from lectures, lecture notes, and problem-set solutions unless otherwise stated.
- Please take a look at all of the questions before you start.

Friday, November 3rd

Question	Points
1 True/False	/20
2 Role Play!	/20
3 Negative Binomial Regression	/25
4 Implementing Naïve Bayes	/15
5 Exponential Family Discriminant Analysis	/15
6 Node-splitting via Logistic Regression	/20
Total	/115

Good luck!

1. **[20 points] True/False**

   For each statement, just indicate whether it is TRUE (always completely correct) or FALSE (at least some aspect is sometimes wrong). No need to provide an explanation.

   Make sure to *completely* fill in the checkboxes (like this ■) corresponding to your answers. Answers with ☒ or ☑ will be marked wrong.

   (a) [2 points] Since the Expectation Maximization (EM) algorithm guarantees monotonic convergence of the likelihood function, we should only stop training if the likelihood function outputs the exact same value in successive iterations.

   □ True

   □ False

   (b) [2 points] The M step of training a 2-class Gaussian Mixture Model (GMM) is equivalent to fitting a Gaussian Discriminant Analysis model for the current guessed labels, with the exception that only soft assignment weights are used, and each cluster has its own covariance matrix.

   □ True

   □ False

   (c) [2 points] Suppose a K-Means model and a GMM are applied to the same dataset and suppose both of them reach their own global optima. If, for GMM, we use a "hard" cluster assignment in the final step (i.e. assign each datapoint x to the cluster with the highest $p(cluster|x)$), then the clusters generated by the two models will be the same.

   □ True

   □ False

   (d) [2 points] Unlike linear regression, logistic regression has a nonlinear activation, and as such, it does not have a linear decision boundary.

   □ True

   □ False

(e) [2 points] Although the EM algorithm (on unlabelled data) can have multiple optima (in terms of parameters), the GDA model (on labelled data) cannot.

□ True

□ False

(f) [2 points] Let's say we're fitting a linear model to a dataset with n features. Suppose the last feature is replaced by the output of a random number generator. We will find that both the bias and the variance increases.

□ True

□ False

(g) [2 points] When we add the regularization term $\lambda||\theta||_2^2$ to a loss function that is convex in θ, we are guaranteed to have a unique global optimum when $\lambda > 0$.

□ True

□ False

(h) [2 points] If we use a kernel method with feature map $\phi(x) = x$, we end up with the same results as with not using a kernel method.

□ True

□ False

(i) [2 points] In boosting, when adding a new model to the existing ensemble, we must train on unseen datapoints to avoid overfitting.

□ True

□ False

(j) [2 points] Let $[x_1 \ldots x_n]$ be samples from a mean-zero random variable, such that $\sum_{i=1}^n x_i = 0$, and $-1 < x_i < 1$. We claim that $\sum_{i=1}^n \log(1 + x_i)$ must be ≥ 0. HINT: is $\log(\cdot)$ a convex function or a concave function?

□ True

□ False

2. **[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, for example, 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:** 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:

(b) [5 points] **Client's Idea:** First, let's find the feature that has the highest correlation with the response variable (y). Now, find the other $n - 1$ features with the *highest* correlation with that one feature. It's impossible for a rectangular matrix to be full rank, but thankfully, now, we have an $n \times n$ matrix (and hence full-rank matrix) of features that will yield the most predictive power.

VP's comments: The client is incorrect in saying that rectangular matrices cannot be full rank, and that all square matrices are full rank. However, they are correct that with the described correlation-based feature selection, they will successfully only include the features with the most predictive power, and avoid over-fitting on noise in the dataset (hence reducing variance).

Your assessment of the VP's comments:

(c) [5 points] **Client's Idea:** If the main issue is that we don't have enough data, a solution is to create more. We can use our existing datapoints to create new ones and add these copies of the originals to our dataset. If we create lots of copies of our datapoints, add random noise to them, and then include these new datapoints with the old ones, we can generate whatever amount of data we need.

VP's comments: The client is looking to extrapolate additional datapoints from the n that we already have. To do so, we can take our original datapoints, add random noise to them, and then split into training and testing datasets (validation as well if desired) afterwards. Now, we won't have to deal with overfitting to too few datapoints, and we'd still have plenty of data for testing (and/or validation).

Your assessment of the VP's comments:

(d) [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:

3. [**25 points**] **Negative Binomial Regression**

Definition: The *negative binomial distribution*,

$$p(y;\phi,k) = \binom{y-1}{k-1}(1-\phi)^{y-k}\phi^k,$$

is a probability distribution over the number of trials until the kth success. Recall that

$$\binom{a}{b} = \frac{a!}{b!(a-b)!},$$

and is also known as the "choose" operator. **For the purposes of this problem, this is all you will need to know about the Negative Binomial distribution.**

Background: Let's say you've been contacted by FIDE, the World Chess Federation, to construct a GLM for the number of tournaments a player needs to play until they achieve their kth big victory[1][2] (and hence achieve the title of Grandmaster).

In particular, you have a response variable y (the number of tournaments) and some fixed k. You decide a Negative Binomial Regression is appropriate, and so y takes on values $\{k, k+1, k+2, \ldots\}$, and its distribution (parametrized by ϕ and k) is modeled as a Negative Binomial distribution.

(a) [6 points] Show that the *negative binomial* distribution, for some fixed k, is an exponential family distribution. You should explicitly specify $b(y), \eta, T(y), a(\eta)$. Also specify what ϕ is in terms of η.

[1] (off-topic) more precisely, this is called a "norm"

[2] (off-topic) historically, in chess, $k = 3$. However, FIDE feels that there are too many grandmasters. So, they are interested in building a system that will work if they decide to raise the cut-off (and so they've contacted you).

(b) [6 points] You're given an IID[3] train set $\{(x^{(i)}, y^{(i)})$ for $i = 1, \ldots, m\}$, and as mentioned earlier, FIDE would like you to model this using a GLM based on a negative binomial distribution. Given some fixed k, find the log-likelihood

$$\ell(\theta) := \log \prod_{i=1}^{m} p(y^{(i)}|x^{(i)};\theta)$$

defined with respect to the *entire train set*, and expressed it only in terms of $k, \theta, x^{(i)}$'s, and $y^{(i)}$'s.

[3] independently and identically distributed

(c) [8 points] Now, for this log-likelihood expression, **derive**

- H – the Hessian matrix,
- $\nabla_\theta \ell(\theta)$ – the gradient vector w.r.t.[4] θ,

and **state** (in terms of those two quantities) one step of Newton's method for maximizing the log-likelihood.

[4] with respect to

(d) [4 points] Show that the Hessian matrix you derived is *negative semi-definite*. This shows that the optimization obejctive is concave, and thus, Newton's method is indeed maximizing log-likelihood.

(e) [**1 point**] If we set $k = 1$, we find that $y|x;\theta$ is distributed according to some other member of the exponential family. What is it?

4. **[15 points] Implementing Naive Bayes**

Suppose you decide to implement a Naive Bayes spam classifier from scratch, just to see what will happen. You will try to implement a **Bernoulli event model** and will not apply Laplace smoothing (for now).

(a) [4 points] Being the machine learning whiz you are, both your math and your code are perfected. However, when running this model (on your training dataset), the prediction almost always gives a division-by-zero error.

When checking through your code, you realize that the error occurs because your code is making predictions by following these steps:

i. Calculate $p(y = 1|x)$ and $p(y = 0|x)$ by equations

$$p(y = 1|x) = \frac{\left(\prod_{j=1}^{d} p(x_j|y = 1)\right) p(y = 1)}{\left(\prod_{j=1}^{d} p(x_j|y = 1)\right) p(y = 1) + \left(\prod_{j=1}^{d} p(x_j|y = 0)\right) p(y = 0)}$$

$$p(y = 0|x) = 1 - p(y = 1|x)$$

ii. Select the y with the higher posterior probability as the predicted label.

Why is this problematic? Explain your answer in 1-2 sentences. (**Hint**: this is **not** due to not applying Laplace smoothing.)

(b) [4 points] Come up with a way of avoiding this error and write out the new steps for prediction. Explain why the new method will be able to produce the same predicted label as the process in (a).

(c) [4 points] After implementing the solution in (b), your model runs smoothly for the training data. However, you still need to deal with unseen words to make predictions over the test dataset. To save yourself from the headache of implementing Laplace smoothing, you come up with another way of treating unseen words:

- If the model encountered a word in both spam and regular emails in the training dataset, we use its ϕ values from training.
- If the model did not encounter a word j in any spam emails, we consider it equally likely that j will appear or not appear in any given spam email. Therefore, we set its $\phi_{j|y=1} = \frac{1}{2}$ and use its $\phi_{j|y=0}$ from training as usual.
- Conversely, if the model did not encounter j in any regular email, we set its $\phi_{j|y=0} = \frac{1}{2}$ and use the $\phi_{j|y=1}$ from training.
- Based on this logic, if j is never encountered in any email during training, we let $\phi_{j|y=1} = \phi_{j|y=0} = \frac{1}{2}$.

Recall that without Laplace smoothing:

$$p(x_j|y=1) = \phi_{j|y=1} = \frac{\sum_{i=1}^n \mathbf{1}\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\}}{\sum_{i=1}^n \mathbf{1}\{y^{(i)} = 1\}}$$

$$p(x_j|y=0) = \phi_{j|y=0} = \frac{\sum_{i=1}^n \mathbf{1}\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\}}{\sum_{i=1}^n \mathbf{1}\{y^{(i)} = 0\}}$$

Please provide a one-sentence answer to each of the following questions.

i. What critical assumption are we making when using this method?

ii. What is wrong with this assumption?

(d) [3 points] Hooray! After implementing everything (including Laplace smoothing) based on the previous questions, you have created a spam classifier that, for a test dataset, achieves a prediction accuracy of 95%. However, when you use this classifier for your own inbox, the model performs quite poorly. In fact, it does not block any spam email.

Suppose the model implementation and the adaptation to your inbox are all done perfectly, and suppose there is no significant difference between the new spam/regular emails you are receiving and the emails in the dataset. What is a possible explanation for the high accuracy and the poor performance? (Hint: in your answer, we are looking for a particular characteristic of the dataset that the model trained on.)

5. **[15 points] Exponential Family Discriminant Analysis**

In class, we have seen Gaussian Discriminant Analysis. In this problem we generalize the approach to work with abstract exponential families.

Consider the model

$$y \sim \text{Bernoulli}(\phi) \tag{1}$$
$$x \mid y = 0 \sim \text{EF}(\eta_0) \tag{2}$$
$$x \mid y = 1 \sim \text{EF}(\eta_1) \tag{3}$$

where EF(η) denotes the exponential family distribution parameterized by η, which has density

$$p(x; \eta) = b(x) \exp\left(\eta^\top T(x) - a(\eta)\right) \tag{4}$$

(a) [5 points] We are given data $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$, where $x^{(i)} \in \mathbb{R}^d$ and $y^{(i)} \in \{0, 1\}$. Let $n_j = \sum_{i=1}^{n} 1[y^{(i)} = j]$ i.e. the number of examples with label j. Given that $T(x) = x$, show that the maximum likelihood estimate of η_j, denoted $\hat{\eta}_j$, must satisfy

$$\mathbb{E}[X; \hat{\eta}_j] = \frac{1}{n_j} \sum_{i=1}^{n} 1[y^{(i)} = j] T(x^{(i)}) \tag{5}$$

You may use any result from class or problem sets without proof.

(b) [6 points] Show that the posterior $p(y = 1 \,|\, x, \phi, \eta_0, \eta_1)$ can be written in the familiar form $\sigma(\tilde{\eta}^\top T(x) + c)$, where σ is the sigmoid function $\sigma(t) = \frac{1}{1+\exp(-t)}$, and $\tilde{\eta}$ and c are expressed in terms of the parameters $\{\phi, \eta_0, \eta_1\}$ and possibly the functions a and/or b. Your answer should specify these expressions.

(c) [4 points] Under what condition(s) is the decision boundary linear in x? Be as general as possible.

6. **[20 points] Node-splitting via logistic regression**

Let's revisit our use of the decision tree model on the binary classification task with d-dimensional real-valued datapoints. That is, we have $x^{(i)} \in \mathbb{R}^d$ and $y^{(i)} \in \{0, 1\}$ for each datapoint. We learned a greedy algorithm in class for fitting a decision tree to a training set of real-valued data. In this algorithm, to determine how to split a node, we iterate over all feature-value pairs to find a threshold that yields the best split, leaving us with two new nodes in the tree (in this problem, we'll consider "best" to mean that the split yields the highest classification accuracy). We'll refer to this node splitting practice as "threshold split selection".

In this problem, we'll also consider an alternative method of node-splitting where we train a logistic regression classifier on the data in a node and use the resulting model to determine the split. Concretely, the two resulting nodes from the logistic regression split will contain the datapoints classified as 0 and 1 by the model, respectively. We'll refer to this as "logistic regression split selection".

Overall note: While we make plenty of assumptions throughout this problem, the solutions that received full credit made no assumptions beyond those made explicit in the problem statements. Strong assumptions about the number of splits in each tree (e.g., $j \approx k$), relationships between m, w, and d (e.g., $m >> w$), and certain values of m, w, and d (e.g., $d \approx 1$) were not necessary to answer the questions correctly and thus did not receive full credit.

(a) [4 points] Suppose we have m datapoints for training. What is the (asymptotic) running time for training a logistic regression model on these datapoints via stochastic gradient descent? Assume that the model converges to an optimum after exactly w epochs. Express your final answer for this and future running time complexity questions using Big O notation (e.g., $O(x + y^3 \log z)$).

(b) [3 points] Supposing again that we have m datapoints, what is the running time for choosing the best threshold via the threshold split selection algorithm from lecture? Assume that evaluating the classification error of a certain split is $O(1)$ (this is a result of an common amortization technique—no need to think about/explain why this assumption holds).
Hint: The running time associated with sorting a list of z values is $O(z \log z)$.

(c) [5 points] Suppose that we want to create two decision trees T_1 and T_2 to fit to a training set of size m, where T_1 uses logistic regression split selection and T_2 uses threshold split selection.

Provide brief explanations (1–2 sentences) for each of the following questions:

i. Can any split achieved via threshold split selection be achieved via logistic regression split selection? And can any split via logistic regression split selection be achieved via a threshold split selection?

ii. Assume that on any individual node, the split that results from logistic regression split selection achieves a classification accuracy no worse than that of threshold split selection. If we allow both trees to perfectly fit the data, which of T_1 and T_2 would you expect to be smaller (have fewer nodes)?

(d) [4 points]

We will now evaluate total running time for training each of these models. To do so, we'll make the following simplifying assumptions:

i. We will restrict the training of T_1 to involve exactly j splits.

ii. We will restrict the training of T_2 to involve exactly k splits.

iii. We will assume that there are $O(m)$ datapoints present at any node we want to split.

With these assumptions in place, what is the running time for *training* for T_1 and for T_2? State once in **mathematical notation** and once in **plain language** what condition would make the running times for training (asymptotically) equal for T_1 and T_2.

Hint: You've already done part of the work here in parts (a) and (b).

(e) [4 points] What is the running time of *prediction* for both T_1 and T_2 (where by prediction we mean evaluating $T_t(x)$ for some datapoint $x \in \mathbb{R}^d$)? State once in **mathematical notation** and once in **plain language** what condition would make the running times for prediction (asymptotically) equal for T_1 and T_2. The restrictions from part (d) still apply, namely that we have exactly j splits in T_1 and k splits in T_2.
Hint: If we have z splits in a tree, then the tree's depth is $O(z)$ in the worst case.

That's all! Congratulations on completing the exam!