CS 189 Introduction to Machine Learning
Fall 2017 EXAMFINALREVIEW

After the exam starts, please write your student ID (or name) on every odd page. On question 2, select *all* correct answers. On question 3, no need to show work. On questions 4-7, justify and show all your work. You may consult only one sheet of notes. Calculators, phones, computers, and other electronic devices are not permitted. There are **22** single sided pages on the exam. **Notify a proctor immediately if a page is missing.** You may, without proof, use theorems and lemmas that were proven in the notes and/or in lecture, unless we explicitly ask for a derivation. **You have 120 minutes**: there are 7 questions with 23 total parts on this exam. The multivariate Gaussian PDF is included on the last page; it is useful but may not be needed.

Exam Location: 2050 VLSB, SID Last Digit 6, 8, 9

PRINT and SIGN Your Name: _____, _____, _____
(last) (first) (signature)

PRINT Your Student ID: _____

Person before you: _____, _____
(name) (SID)

Person behind you: _____, _____
(name) (SID)

Person to your left: _____, _____
(name) (SID)

Person to your right: _____, _____
(name) (SID)

Row (front is 1): _____ Seat (leftmost is 1): _____ (*Include empty seats/rows.*)

# 1 Pre-exam Questions (3 points)

(a) **(1 point)** What are your favorite desserts?

(b) **(2 points)** What are your plans for Halloween, if any?

Do not turn this page until your instructor tells you to do so.

Extra page. If you want the work on this page to be graded, mention it on the problem's main page.

# 2 Multiple Choice

For the following multiple choice questions, select all that apply.

(a) What strategies can help reduce overfitting in decision trees?

- ◯ Pruning
- ◯ Enforce a minimum number of samples in leaf nodes
- ◯ Make sure each leaf node is one pure class
- ◯ Enforce a maximum depth for the tree

(b) Neural Networks...

- ◯ optimize a convex cost function
- ◯ always output values between 0 and 1
- ◯ can be used for regression as well as classification
- ◯ can be used in an ensemble

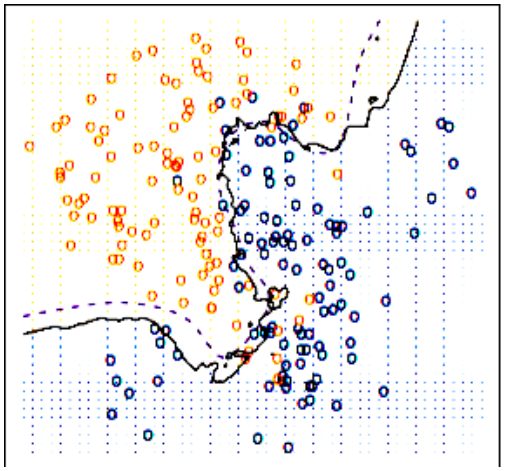
(c) Which of the following methods can achieve zero training error on *any* linearly separable dataset?

- ◯ Decision tree
- ◯ 15-nearest neighbors
- ◯ Hard-margin SVM
- ◯ Soft-margin SVM

(d) Which of the following can help to reduce overfitting in an SVM classifier?

- ◯ Use of slack variables
- ◯ High-degree polynomial features
- ◯ Normalizing the data
- ◯ Setting a very low learning rate

(e) Which value of $k$ in the $k$-nearest neighbors algorithm generates the solid decision boundary depicted here? There are only 2 classes. (Ignore the dashed line, which is the Bayes decision boundary.)



◯ $k = 1$ ◯ $k = 2$

◯ $k = 10$ ◯ $k = 100$

(f) Which of the following are properties that a kernel matrix always has?

◯ Invertible ◯ All the entries are positive

◯ At least one negative eigenvalue ◯ Symmetric

(g) You've just finished training a random forest for spam classification, and it is getting abnormally bad performance on your validation set, but good performance on your training set. Your implementation has no bugs. What could be causing the problem?

◯ Your decision trees are too deep ◯ You have too few trees in your ensemble

◯ You are randomly sampling too many features when you choose a split ◯ Your bagging implementation is randomly sampling sample points *without* replacement

(h) In terms of the bias-variance decomposition, a 1-nearest neighbor classifier has _____ than a 3-nearest neighbor classifier.

◯ higher variance ◯ higher bias

◯ lower variance ◯ lower bias

(i) Which of the following are true about bagging?

◯ In bagging, we choose random subsamples of the input points with replacement

◯ In bagging, we assign more weight to trees with higher accuracy

◯ The main purpose of bagging is to decrease the bias of learning algorithms.

◯ If we use decision trees that have one sample point per leaf, bagging never gives lower training error than one ordinary decision tree

(j) Why is PCA sometimes used as a preprocessing step before regression?

◯ To reduce overfitting by removing poorly predictive dimensions.

◯ To expose information missing from the input data.

◯ To make computation faster by reducing the dimensionality of the data.

◯ For inference and scientific discovery, we prefer features that are not axis-aligned.

(k) For which of the following does normalizing your input features influence the predictions?

◯ decision tree (with usual splitting method)

◯ Lasso

◯ neural network

◯ soft-margin support vector machine

(l) Why would we use a random forest instead of a decision tree?

◯ For lower training error.

◯ To reduce the variance of the model.

◯ To better approximate posterior probabilities.

◯ For a model that is easier for a human to interpret.

(m) A low-rank approximation of a matrix can be useful for

◯ removing noise.

◯ discovering latent categories in the data.

◯ filling in unknown values.

◯ matrix compression.

(n) Which of the following statements is true about the standard $k$-means clustering algorithm?

◯ The random partition initialization method usually outperforms the Forgy method.

◯ After a sufficiently large number of iterations, the clusters will stop changing.

◯ It is computationally infeasible to find the optimal clustering of $n = 15$ points in $k = 3$ clusters.

◯ You can use the metric $d(x,y) = \frac{x \cdot y}{|x| \cdot |y|}$.

# 3 Maximum Likelihood Estimation for Reliability Testing

Suppose we are reliability testing $n$ units taken randomly from a population of identical appliances. We want to estimate the mean failure time of the population. We assume the failure times come from an exponential distribution with parameter $\lambda > 0$, whose probability density function is $f(x) = \lambda e^{-\lambda x}$ (on the domain $x \geq 0$) and whose cumulative distribution function is $F(x) = \int_0^x f(x)\,\mathrm{d}x = 1 - e^{-\lambda x}$.

(a) In an ideal (but impractical) scenario, we run the units until they all fail. The failure times are $t_1, t_2, \ldots, t_n$.

Formulate the likelihood function $\mathscr{L}(\lambda; t_1, \ldots, t_n)$ for our data. Then find the maximum likelihood estimate $\hat{\lambda}$ for the distribution's parameter.

(b) In a more realistic scenario, we run the units for a fixed time $T$. We observe $r$ unit failures, where $0 \leq r \leq n$, and there are $n - r$ units that survive the entire time $T$ without failing. The failure times are $t_1, t_2, \ldots, t_r$.

Formulate the likelihood function $\mathscr{L}(\lambda; n, r, t_1, \ldots, t_r)$ for our data. Then find the maximum likelihood estimate $\hat{\lambda}$ for the distribution's parameter.

*Hint 1:* What is the probability that a unit will not fail during time $T$? *Hint 2:* It is okay to define $\mathscr{L}(\lambda)$ in a way that includes contributions (densities and probability masses) that are not commensurate

with each other. Then the constant of proportionality of $\mathscr{L}(\lambda)$ is meaningless, but that constant is irrelevant for finding the best-fit parameter $\hat{\lambda}$. *Hint 3:* If you're confused, for part marks write down the likelihood that $r$ units fail and $n-r$ units survive; then try the full problem. *Hint 4:* If you do it right, $\hat{\lambda}$ will be the number of observed failures divided by the sum of unit test times.

# 4 PCA

You are given a design matrix $X = \begin{bmatrix} 6 & -4 \\ -3 & 5 \\ -2 & 6 \\ 7 & -3 \end{bmatrix}$. Let's use PCA to reduce the dimension from 2 to 1.

(a) Compute the covariance matrix for the sample points. (Warning: Observe that $X$ is not centered.) Then compute the **unit** eigenvectors, and the corresponding eigenvalues, of the covariance matrix. *Hint:* If you graph the points, you can probably guess the eigenvectors (then verify that they really are eigenvectors).

(b) Suppose we use PCA to project the sample points onto a one-dimensional space. What one-dimensional subspace are we projecting onto? For each of the four sample points in $X$ (not the centered version of $X$!), write the coordinate (in principal coordinate space, not in $\mathbb{R}^2$) that the point is projected to.

(c) Given a design matrix $X$ that is taller than it is wide, prove that every right singular vector of $X$ with singular value $\sigma$ is an eigenvector of the covariance matrix with eigenvalue $\sigma^2$.

# 5 Gradient Descent for $k$-means Clustering

Recall the loss function for $k$-means clustering with $k$ clusters, sample points $x_1, ..., x_n$, and centers $\mu_1, ..., \mu_k$:

$$L = \sum_{j=1}^{k} \sum_{x_i \in S_j} \|x_i - \mu_j\|^2,$$

where $S_j$ refers to the set of data points that are closer to $\mu_j$ than to any other cluster mean.

(a) Instead of updating $\mu_j$ by computing the mean, let's minimize $L$ with **batch** gradient descent while holding the sets $S_j$ fixed. Derive the update formula for $\mu_1$ with learning rate (step size) $\varepsilon$.

(b) Derive the update formula for $\mu_1$ with **stochastic** gradient descent on a single sample point $x_i$. Use learning rate $\varepsilon$.

(c) In this part, we will connect the batch gradient descent update equation with the standard $k$-means algorithm. Recall that in the update step of the standard algorithm, we assign each cluster center to be the mean (centroid) of the data points closest to that center. It turns out that a particular choice of the learning rate $\varepsilon$ (which may be different for each cluster) makes the two algorithms (batch gradient descent and the standard $k$-means algorithm) have identical update steps. Let's focus on the update for the first cluster, with center $\mu_1$. Calculate the value of $\varepsilon$ so that both algorithms perform the same update for $\mu_1$. (If you do it right, the answer should be very simple.)

# 6 Kernels

(a) What is the primary motivation for using the kernel trick in machine learning algorithms?

(b) Prove that for every design matrix $X \in R^{n \times d}$, the corresponding kernel matrix is positive semidefinite.

(c) Suppose that a regression algorithm contains the following line of code.

$$\mathbf{w} \leftarrow \mathbf{w} + X^\top MXX^\top \mathbf{u}$$

Here, $X \in \mathbb{R}^{n \times d}$ is the design matrix, $\mathbf{w} \in \mathbb{R}^d$ is the weight vector, $M \in \mathbb{R}^{n \times n}$ is a matrix unrelated to $X$, and $\mathbf{u} \in \mathbb{R}^n$ is a vector unrelated to $X$. We want to derive a dual version of the algorithm in which we express the weights $\mathbf{w}$ as a linear combination of samples $X_i$ (rows of $X$) and a dual weight vector $\mathbf{a}$ contains the coefficients of that linear combination. Rewrite the line of code in its dual form so that it updates $\mathbf{a}$ correctly (and so that $\mathbf{w}$ does not appear).

(d) Can this line of code for updating $\mathbf{a}$ be kernelized? If so, show how. If not, explain why.

# 7 Decision Trees

Consider the design matrix

$$\begin{bmatrix} 4 & 6 & 9 & 1 & 7 & 5 \\ 1 & 6 & 5 & 2 & 3 & 4 \end{bmatrix}^\top$$

representing 6 sample points, each with two features $f_1$ and $f_2$.

The labels for the data are

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}^\top$$

In this question, we build a decision tree of depth 2 by hand to classify the data.

(a) What is the entropy at the root of the tree?

(b) What is the rule for the first split? Write your answer in a form like $f_1 \geq 4$ or $f_2 \geq 3$. Hint: you should be able to eyeball the best split without calculating the entropies.

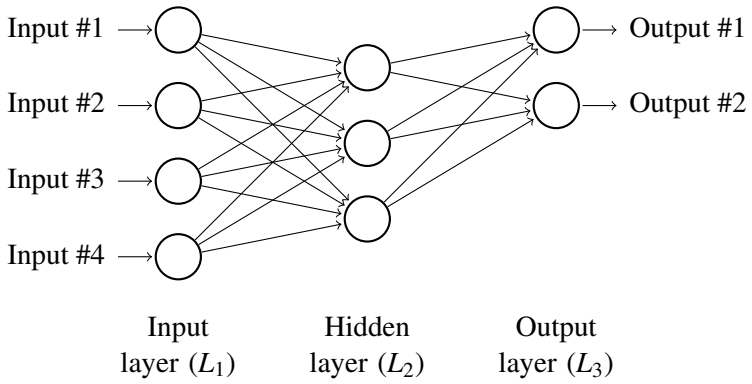
(c) For each of the two treenodes after the first split, what is the rule for the second split?

(d) Let's return to the root of the tree, and suppose we're incompetent tree builders. Is there a (not trivial) split at the root that would have given us an information gain of zero? Explain your answer.

# 8 Neural Networks

Consider a three layer fully-connected network with $n_1, n_2, n_3$ neurons in three layers respectively. Inputs are fed into the first layer. The loss is mean squared error $E$, and the non-linearity is a sigmoid function. Let the label vector be $t$ of size $n_3$. Let each layer output vector be $y_i$ and input vector be $z_i$, both of size $n_i$. Let the weight between layer $i$ and layer $i+1$ be $W_{ii+1}$. The $j$-th element in $y_i$ is defined by $y_i^j$, same for $z_i^j$. The weight connecting $k$-th and $l$-th neuron in $i, i+1$ layers is defined by $W_{ii+1}^{kl}$ (You don't need to consider bias in this problem).

Here is a summary of our notation:




- $\sigma$ denotes the activation function for $L_2$ and $L_3$, $\sigma(x) = \frac{1}{1+e^{-x}}$. There is no activation applied to the input layer.
- $z_i^{(j)} = \sum_{k=1}^{P} W_{i-1i}^{kj} x_{i-1}^{(k)}$
- $y_i^{(j)} = \sigma\left(\sum_{k=1}^{P} W_{i-1i}^{kj} x_{i-1}^{(k)}\right)$

Now solve the following problems.

(a) Find $\frac{\partial E}{\partial z_3^j}$ in terms of $y_3^j$.

(b) Find $\frac{\partial E}{\partial y_2^k}$ in terms of elements in $W_{23}$ and $\frac{\partial E}{\partial z_3^j}$.

(c) Find $\frac{\partial E}{\partial W_{23}^{kj}}$ in terms of $y_2^k$, $y_3^j$ and $t^j$.

(d) If the input to a neuron in max-pooling layer is $x$ and the output is $y = max(x)$, derive $\frac{\partial y}{\partial x_i}$.

The PDF of a multivariate $n$-dimensional Gaussian with mean $\mu$ and covariance matrix $\Sigma$ is given by

$$f(x) = \frac{1}{(2\pi)^{n/2}\sqrt{\det(\Sigma)}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right).$$

Doodle page! Draw us something if you want or give us suggestions or complaints. You can also use this page to report anything suspicious that you might have noticed.