CS 189 Spring 2019 Introduction to Machine Learning Final Exam

- Please do not open the exam before you are instructed to do so.
- **Electronic devices are forbidden on your person**, including cell phones, iPods, headphones, and laptops. Turn your cell phone off and **leave all electronics at the front of the room**, or **risk getting a zero** on the exam.
- When you start, the **first thing you should do** is **check that you have all 12 pages and all 6 questions**. The second thing is to please **write your initials at the top right of every page after this one** (e.g., write "JS" if you are Jonathan Shewchuk).
- The exam is closed book, closed notes except your two cheat sheets.
- You have 3 hours.
- Mark your answers on the exam itself in the space provided. Do **not** attach any extra sheets.
- The total number of points is 150. There are 26 multiple choice questions worth 3 points each, and 5 written questions worth a total of 72 points.
- For multiple answer questions, fill in the bubbles for **ALL correct choices:** there may be more than one correct choice, but there is always at least one correct choice. **NO partial credit** on multiple answer questions: the set of all correct answers must be checked.

First name	
Last name	
SID	
First and last name of student to your left	
First and last name of student to your right	

Q1. [78 pts] Multiple Answer

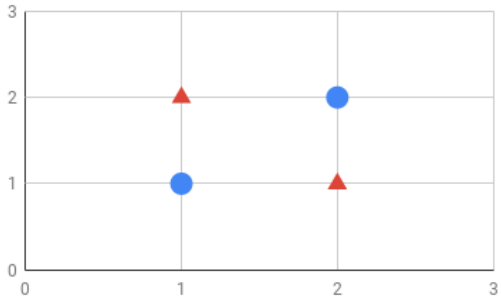
Fill in the bubbles for **ALL correct choices**: there may be more than one correct choice, but there is always at least one correct choice. **NO partial credit**: the set of all correct answers must be checked.

(a) [3 pts] Which of the following algorithms can learn nonlinear decision boundaries? The decision trees use only axis-aligned splits.

- ◯ A depth-five decision tree
- ◯ AdaBoost with depth-one decision trees
- ◯ Quadratic discriminant analysis (QDA)
- ◯ Perceptron

(b) [3 pts] Which of the following classifiers are capable of achieving 100% training accuracy on the data below? The decision trees use only axis-aligned splits.

- ◯ Logistic regression
- ◯ AdaBoost with depth-one decision trees
- ◯ A neural network with one hidden layer
- ◯ AdaBoost with depth-two decision trees



(c) [3 pts] Which of the following are true of support vector machines?

- ◯ Increasing the hyperparameter C tends to decrease the training error
- ◯ Increasing the hyperparameter C tends to decrease the margin
- ◯ The hard-margin SVM is a special case of the soft-margin with the hyperparameter C set to zero
- ◯ Increasing the hyperparameter C tends to decrease the sensitivity to outliers

(d) [3 pts] Let $r(x)$ be a decision rule that minimizes the risk for a three-class classifier with labels $y \in \{0, 1, 2\}$ and an asymmetric loss function. What is true about $r(\cdot)$?

- ◯ $\forall y \in \{0, 1, 2\}, \ \exists x : r(x) = y$
- ◯ $\forall x$, $r(x)$ is a class y that maximizes the posterior probability $P(Y = y|X = x)$
- ◯ If we don't have access to the underlying data distribution $P(X)$ or $P(Y|X)$, we cannot exactly compute the risk of $r(\cdot)$
- ◯ If $P(X = x)$ changes but $P(Y = y|X = x)$ remains the same for all x, y, $r(X)$ still minimizes the risk

(e) [3 pts] Which of the following are true about two-class Gaussian discriminant analysis? Assume you have estimated the parameters $\hat{\mu}_C, \hat{\Sigma}_C, \hat{\pi}_C$ for class C and $\hat{\mu}_D, \hat{\Sigma}_D, \hat{\pi}_D$ for class D.

- ◯ If $\hat{\mu}_C = \hat{\mu}_D$ and $\hat{\pi}_C = \hat{\pi}_D$, then the LDA and QDA classifiers are identical
- ◯ If $\hat{\Sigma}_C = \hat{\Sigma}_D$, $\hat{\pi}_C = 1/6$, and $\hat{\pi}_D = 5/6$, then the LDA and QDA classifiers are identical
- ◯ If $\hat{\Sigma}_C = I$ (the identity matrix) and $\hat{\Sigma}_D = 5I$, then the LDA and QDA classifiers are identical
- ◯ If the LDA and QDA classifiers are identical, then the posterior probability $P(Y = C|X = x)$ is linear in x

(f) [3 pts] Consider an $n \times d$ design matrix X with labels $y \in \mathbb{R}^n$. What is true of fitting this data with dual ridge regression with the polynomial kernel $k(X_i, X_j) = (X_i^T X_j + 1)^p = \Phi(X_i)^\top \Phi(X_j)$ and regularization parameter $\lambda > 0$?

◯ If the polynomial degree is high enough, the polynomial will fit the data exactly

◯ The algorithm computes $\Phi(X_i)$ and $\Phi(X_j)$ in $O(d^p)$ time

◯ The algorithm solves an $n \times n$ linear system

◯ When n is very large, this dual algorithm is more likely to overfit than the primal algorithm with degree-p polynomial features

(g) [3 pts] Consider the kernel perceptron algorithm on an $n \times d$ design matrix X. We choose a matrix $M \in \mathbb{R}^{D \times d}$ and define the feature map $\Phi(x) = Mx \in \mathbb{R}^D$ and the kernel $k(x, z) = \Phi(x) \cdot \Phi(z)$. Which of the following are always true?

◯ The kernel matrix is $XM^\top MX^\top$

◯ If the primal perceptron algorithm terminates, then the kernel perceptron algorithm terminates

◯ The kernel matrix is $MX^\top XM^\top$

◯ If the kernel perceptron algorithm terminates, then the primal perceptron algorithm terminates

(h) [3 pts] Which of the following are true of decision trees? Assume splits are binary and are done so as to maximize the information gain.

◯ If there are at least two classes at a given node, there exists a split such that information gain is strictly positive

◯ As you go down any path from the root to a leaf, the information gain at each level is non-increasing

◯ The deeper the decision tree is, the more likely it is to overfit

◯ Random forests are less likely to overfit than decision trees

(i) [3 pts] While solving a classification problem, you use a pure, binary decision tree constructed by the standard greedy procedure we outlined in class. While your training accuracy is perfect, your validation accuracy is unexpectedly low. Which of the following, in isolation, is likely to improve your validation accuracy in most real-world applications?

◯ Lift your data into a quadratic feature space

◯ Select a random subset of the features and use only those in your tree

◯ Normalize each feature to have variance 1

◯ Prune the tree, using validation to decide how to prune

(j) [3 pts] For the sigmoid activation function and the ReLU activation function, which of the following are true in general?

◯ Both activation functions are monotonically non-decreasing

◯ Both functions have a monotonic first derivative

◯ Compared to the sigmoid, the ReLU is more computationally expensive

◯ The sigmoid derivative $s'(\gamma)$ is quadratic in $s(\gamma)$

(k) [3 pts] Which of the following are true in general for backpropagation?

◯ It is a dynamic programming algorithm

◯ Some of the derivatives cannot be fully computed until the backward pass

◯ The weights are initially set to zero

◯ Its running time grows exponentially in the number of layers

(l) [3 pts] Facets of neural networks that have (reasonable, though not perfect) analogs in human brains include

◯ backpropagation

◯ linear combinations of input values

◯ convolutional masks applied to many patches

◯ edge detectors

(m) [3 pts] Which of the following are true of the vanishing gradient problem for sigmoid units?

◯ Deeper neural networks tend to be more susceptible to vanishing gradients

◯ Using ReLU units instead of sigmoid units can reduce this problem

◯ If a unit has the vanishing gradient problem for one training point, it has the problem for every training point

◯ Networks with sigmoid units don't have this problem if they're trained with the cross-entropy loss function

(n) [3 pts] Suppose our input is two-dimensional sample points, with ten non-exclusive classes those points may belong to (i.e., a point can belong to more than one class). To train a classifier, we build a fully-connected neural network (with bias terms) that has a single hidden layer of twenty units and an output layer of ten units (one for each class). Which statements apply?

◯ For the output units, softmax activations are more appropriate than sigmoid activations

◯ For the hidden units, ReLU activations are more appropriate than linear activations

◯ This network will have 240 trainable parameters

◯ This network will have 270 trainable parameters

(o) [3 pts] Which of the following can lead to valid derivations of PCA?

◯ Fit the mean and covariance matrix of a Gaussian distribution to the sample data with maximum likelihood estimation

◯ Find the direction w that minimizes the sum of projection distances

◯ Find the direction w that minimizes the sample variance of the projected data

◯ Find the direction w that minimizes the sum of squares of projection distances

(p) [3 pts] Write the SVD of an $n \times d$ design matrix X (with $n \geq d$) as $X = UDV^T$. Which of the following are true?

◯ The components of D are all nonnegative

◯ The columns of V all have unit length and are orthogonal to each other

◯ If X is a real, symmetric matrix, the SVD is always the same as the eigendecomposition

◯ The columns of D are orthogonal to each other

(q) [3 pts] Which of the following is true about Lloyd's algorithm for k-means clustering?

◯ It is a supervised learning algorithm

◯ If run for long enough, it will always terminate

◯ It never returns to a particular assignment of classes to sample points after changing to another one

◯ No algorithm (Lloyd's or any other) can always find the optimal solution

(r) [3 pts] Which of the following are advantages of using k-medoid clustering instead of k-means?

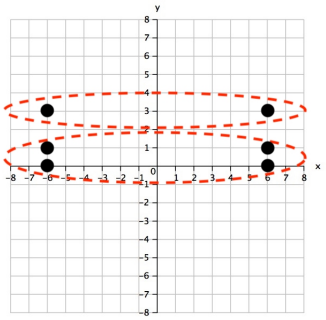
◯ k-medoids is less sensitive to outliers

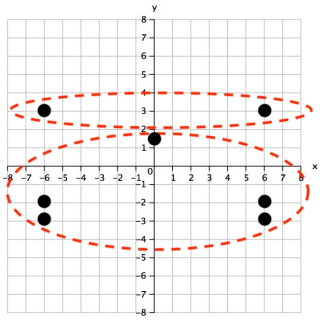
◯ Medoids are faster to compute than means

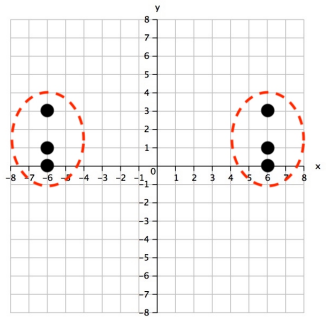
◯ Medoids make more sense than means for non-Euclidean distance metrics

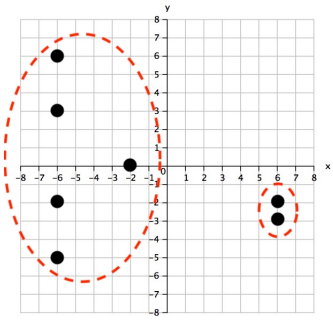
◯ The k-medoids algorithm with the Euclidean distance metric has no hyperparameters, unlike k-means

(s) [3 pts] We wish to cluster 2-dimensional points into two clusters, so we run Lloyd's algorithm for k-means clustering until convergence. Which of the following clusters could it produce? (The points inside an oval belong to the same cluster).

○ 

○ 

○ 

○ 

(t) [3 pts] Which of the following are true of hierarchical clustering?

○ The number k of clusters is a hyperparameter

○ The greedy agglomerative clustering algorithm repeatedly fuses the two clusters that minimize the distance between clusters

○ Complete linkage works only with the Euclidean distance metric

○ During agglomerative clustering, single linkage is more sensitive to outliers than complete linkage

(u) [3 pts] Which of the following are true of spectral clustering?

○ The Fiedler vector is the eigenvector associated with the second largest eigenvalue of the Laplacian matrix

○ Nobody knows how to find the sparsest cut in polynomial time

○ The relaxed optimization problem for partitioning a graph involves minimizing the Rayleigh quotient of the Laplacian matrix and an indicator vector (subject to a constraint)

○ The Laplacian matrix of a graph is invertible

(v) [3 pts] For binary classification, which of the following statements are true of AdaBoost?

○ It can be applied to neural networks

○ It uses the majority vote of learners to predict the class of a data point

○ The metalearner provides not just a classification, but also an estimate of the posterior probability

○ The paper on AdaBoost won a Gödel Prize

(w) [3 pts] For binary classification, which of the following statements are true of AdaBoost with decision trees?

◯ It usually has lower bias than a single decision tree

◯ It is popular because it usually works well even before any hyperparameter tuning

◯ To use the weight w_i of a sample point X_i when training a decision tree G, we scale the loss function $L(G(X_i), y_i)$ by w_i

◯ It can train multiple decision trees in parallel

(x) [3 pts] Which of the following are reasons one might choose latent factor analysis (LFA) over k-means clustering to group together n data points in $\mathbb{R}^d$?

◯ LFA is not sensitive to how you initialize it, whereas Lloyd's algorithm is

◯ LFA allows us to consider points as belonging to multiple "overlapping" clusters, whereas in k-means, each point belongs to only one cluster

◯ In market research, LFA can distinguish different consumer types, whereas k-means cannot

◯ k-means requires you to guess k in advance, whereas LFA makes it easier to infer the right number of clusters after the computation

(y) [3 pts] Which of the following are true for k-nearest neighbor classification?

◯ It is more likely to overfit with $k = 1$ (1-NN) than with $k = 1{,}000$ (1,000-NN)

◯ In very high dimensions, exhaustively checking every training point is often faster than any widely used competing exact k-NN query algorithm

◯ If you have enough training points drawn from the same distribution as the test points, k-NN can achieve accuracy almost as good as the Bayes decision rule

◯ The optimal running time to classify a point with k-NN grows linearly with k

(z) [3 pts] Suppose we use the k-d tree construction and query algorithms described in class to find the *approximate* nearest neighbor of a query point among n sample points. Select the true statements.

◯ It is possible to guarantee that the tree has $O(\log n)$ depth by our choice of splitting rule at each treenode

◯ Sometimes we permit the k-d tree to be unbalanced so we can choose splits with better information gain

◯ Querying the k-d tree is faster than querying a Voronoi diagram for sample points in $\mathbb{R}^2$

◯ Sometimes the query algorithm declines to search inside a box that's closer to the query point than the nearest neighbor it's found so far

Q2. [17 pts] Getting Down(hill) with the Funk Function

The Netflix Prize was an open competition for the best *collaborative filtering* algorithm to predict user ratings for films. Competitors were given an $n \times d$ *ratings matrix* R; entry R_{jk} is user j's rating of movie k. Because users only watch a small fraction of the movies, most entries in R are unobserved, hence filled with a default value of zero. Latent factor analysis attempts to predict missing ratings by replacing R with a low-rank approximation, which is a truncated singular value decomposition (SVD).

(a) [4 pts] Given the SVD $R = UDV^\top$, write a formula for the rank-r truncated SVD R' for comparison; make sure you explain your notation. Then write the standard restrictions (imposed by the definition of SVD) on U, D, and V.

LFA leaves plenty of room for improvement. Simon Funk (a pseudonym, but a real person), who at one point was ranked third in the competition, developed a method called "Funk SVD." Recall that the rank-r truncated SVD R' minimizes the Frobenius norm $\|R - R'\|_F$, subject to the constraint that R' has rank r. Mr. Funk modified this approach to learn two matrices $A \in \mathbb{R}^{n \times r}$ and $B \in \mathbb{R}^{r \times d}$ such that $AB \approx R$. The rank of AB cannot exceed r. Let a_j be the jth row of A, let b_k be the kth column of B, and observe that $(AB)_{jk} = a_j \cdot b_k$. Mr. Funk solves the problem of finding matrices A and B that minimize the objective function

$$L(A, B) = \sum_{j,k\,:\,R_{jk} \neq 0} (R_{jk} - a_j \cdot b_k)^2.$$

The key difference between this objective function and the one optimized by the truncated SVD is that the summation is over **only nonzero** components of R. Instead of computing an SVD, Mr. Funk minimizes this objective with gradient descent.

(b) [2 pts] Explain why the optimal solution is not unique; that is, there is more than one pair of optimal matrices (A, B).

(c) [5 pts] Although Mr. Funk uses stochastic gradient descent, we will derive a batch gradient descent algorithm. It turns out to be easiest to write the update rule for A one row at a time. State the gradient descent rule for updating row a_j during the minimization of Mr. Funk's objective function $L(A, B)$. Use some step size $\epsilon > 0$. (Be careful that you sum only the correct terms!) (Note: there is a symmetric rule for updating b_k; the algorithm must update both A and B.)

(d) [3 pts] What will happen if you initialize Funk SVD by setting $A \leftarrow 0$ and $B \leftarrow 0$? Suggest a better initialization.

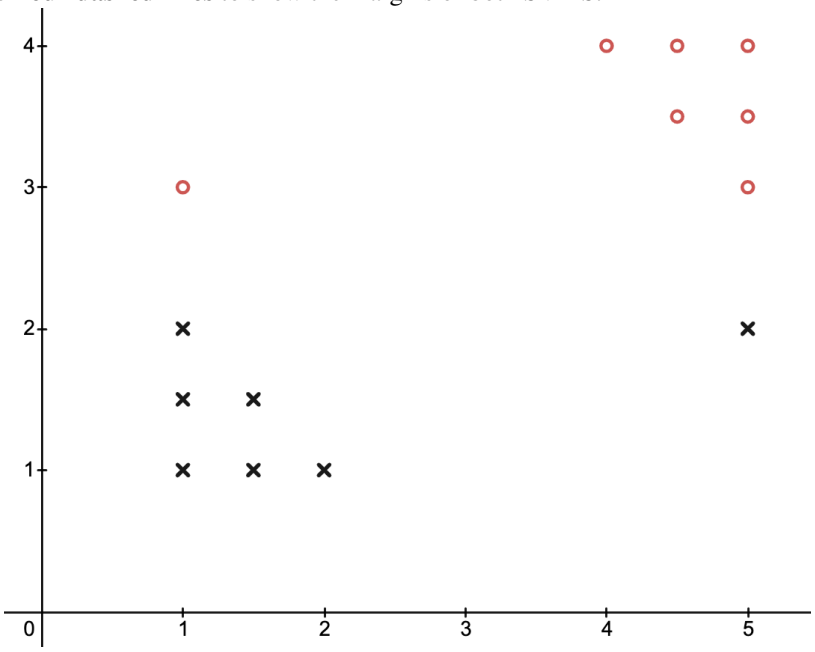
(e) [3 pts] Consider the special case where $r = 1$ and the matrix R has no zero entries. In this case, what is the relationship between an optimal solution A, B and the rank-one truncated singular value decomposition?

Q3. [10 pts] Decision Boundaries

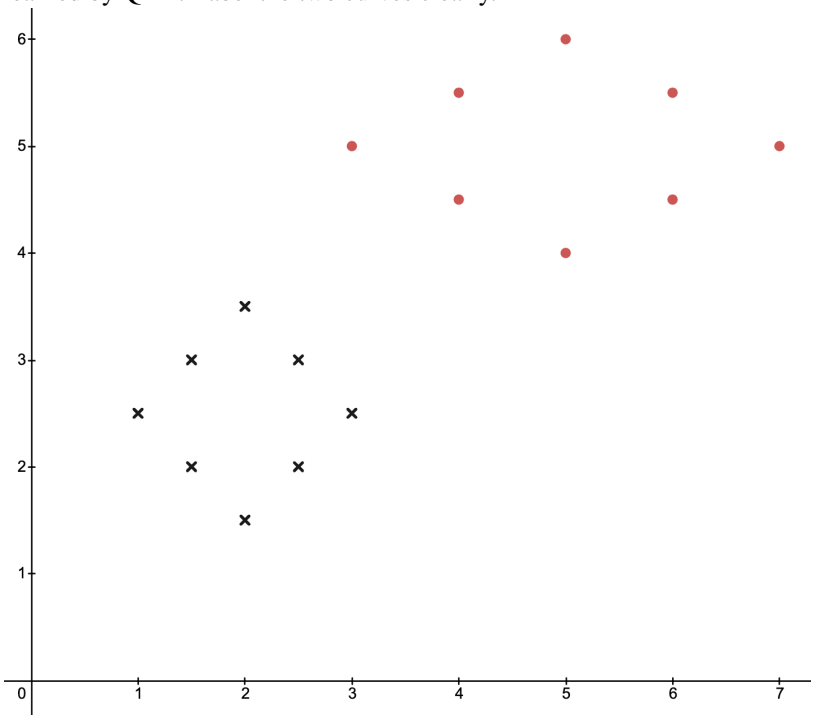
In the question, you will draw the decision boundaries that classifiers would learn.

(a) [6 pts] Given the sample points below, draw and label **two lines**: the decision boundary learned by a hard-margin SVM and the decision boundary learned by a soft-margin SVM. We are not specifying the hyperparameter C, but don't make C too extreme. (We are looking for a qualitative difference between hard- and soft-margin SVMs.) **Label the two lines clearly.**

Also draw and label **four dashed lines** to show the margins of both SVMS.



(b) [4 pts] Given the sample points below, draw and label **two curves**: the decision boundary learned by LDA and the decision boundary learned by QDA. Label the two curves clearly.



Q4. [16 pts] Kernel Principal Components Analysis

Let X be an $n \times d$ design matrix. Suppose that X has been centered, so the sample points in X have mean zero. In this problem we consider kernel PCA and show that it equates to solving a *generalized Rayleigh quotient problem*.

(a) [1 pt] Fill in the blank: every principal component direction for X is an eigenvector of _____.

(b) [1 pt] Fill in the blank: an optimization problem can be kernelized only if its solution w is always a linear combination of the sample points. In other words, we can write it in the form $w =$ _____.

(c) [4 pts] Show that every principal component direction w with a nonzero eigenvalue is a linear combination of the sample points (even when $n < d$).

(d) [4 pts] Let $\Phi(z)$ be a feature map that takes a point $z \in \mathbb{R}^d$ and maps it to a point $\Phi(z) \in \mathbb{R}^D$, where D might be extremely large or even infinite. But suppose that we can compute the kernel function $k(x, z) = \Phi(x) \cdot \Phi(z)$ much more quickly than we can compute $\Phi(x)$ directly. Let $\Phi(X)$ be the $n \times D$ matrix in which each sample point is replaced by a featurized point. By our usual convention, row i of X is $X_i^\top$, and row i of $\Phi(X)$ is $\Phi(X_i)^\top$.

Remind us: what is the kernel matrix K? Answer this two ways: explain the relationship between K and the kernel function $k(\cdot, \cdot)$; then write the relationship between K and $\Phi(X)$. Lastly, show that these two definitions are equivalent.

(e) [2 pts] Fill in the space: the first principle component direction of the *featurized* design matrix $\Phi(X)$ is any nonzero vector $w \in \mathbb{R}^D$ that maximizes the *Rayleigh quotient*, which is .

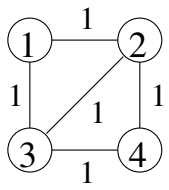
(f) [4 pts] Show that the problem of maximizing this Rayleigh quotient is equivalent to maximizing

$$\frac{a^\top B a}{a^\top C a}$$

for some positive semidefinite matrices $B, C \in \mathbb{R}^{n \times n}$, where $a \in \mathbb{R}^n$ is a vector of dual weights. This expression is called a *generalized Rayleigh quotient*. What are the matrices B and C? For full points, express them in a form that does not require any direct computation of the feature vectors Φ, which could be extremely long.

Q5. [12 pts] Spectral Graph Clustering

Let's apply spectral graph clustering to this graph.



(a) [4 pts] Write the Laplacian matrix L for this graph. All the edges have weight 1.

(b) [2 pts] Consider the minimum bisection problem, where we find an indicator vector y that minimizes $y^\top Ly$, subject to the balance constraint $1^\top y = 0$ and the strict binary constraint $\forall i, y_i = 1$ or $y_i = -1$. Write an indicator vector y that represents a minimum bisection of this graph.

(c) [4 pts] Suppose we relax (discard) the binary constraint and replace it with the weaker constraint $y^\top y =$ constant, permitting y to have real-valued components. (We keep the balance constraint.) What indicator vector is a solution to the relaxed optimization problem? What is its eigenvalue?

Hint: Look at the symmetries of the graph. Given that the continuous values of the y_i's permit some of the vertices to be at or near zero, what symmetry do you think would minimize the continuous-valued cut? Guess and then check whether it's an eigenvector.

(d) [2 pts] If we apply the sweep cut to find a cut with good sparsity, what two clusters do we get? Is it a bisection?

Q6. [17 pts] Learning Mixtures of Gaussians with k-Means

Let $X_1, \ldots, X_n \in \mathbb{R}^d$ be independent, identically distributed points sampled from a mixture of two normal (Gaussian) distributions. They are drawn independently from the probability distribution function (PDF)

$$p(x) = \theta\, N_1(x) + (1-\theta)\, N_2(x), \quad \text{where } N_1(x) = \frac{1}{(\sqrt{2\pi})^d} e^{-\|x-\mu_1\|^2/2} \text{ and } N_2(x) = \frac{1}{(\sqrt{2\pi})^d} e^{-\|x-\mu_2\|^2/2}$$

are the PDFs for the isotropic multivariate normal distributions $\mathcal{N}(\mu_1, 1)$ and $\mathcal{N}(\mu_2, 1)$, respectively. The parameter $\theta \in (0, 1)$ is called the *mixture proportion*. In essence, we flip a biased coin to decide whether to draw a point from the first Gaussian (with probability θ) or the second (with probability $1 - \theta$).

Each data point is generated as follows. First draw a random Z_i, which has value 1 with probability θ, and has value 2 with probability $1 - \theta$. Then, draw $X_i \sim \mathcal{N}(\mu_{Z_i}, 1)$. Our learning algorithm gets X_i as an input, but does not know Z_i.

Our goal is to find the maximum likelihood estimates of the three unknown distribution parameters $\theta \in (0, 1)$, $\mu_1 \in \mathbb{R}^d$, and $\mu_2 \in \mathbb{R}^d$ from the sample points $X_1, \ldots, X_n$. Unlike MLE for one Gaussian, it is **not** possible to give explicit analytic formulas for these estimates. Instead, we develop a variant of k-means clustering which (often) converges to the correct maximum likelihood estimates of θ, μ_1, and μ_2. This variant doesn't assign each point entirely to one cluster; rather, each point is assigned an estimated posterior probability of coming from normal distribution 1.

(a) [4 pts] Let $\tau_i = P(Z_i = 1|X_i)$. That is, τ_i is the posterior probability that point X_i has $Z_i = 1$. Use Bayes' Theorem to express τ_i in terms of X_i, θ, μ_1, μ_2, and the Gaussian PDFs $N_1(x)$ and $N_2(x)$. To help you with part (c), also write down a similar formula for $1 - \tau_i$, which is the posterior probability that $Z_i = 2$.

(b) [3 pts] Write down the log-likelihood function, $\ell(\theta, \mu_1, \mu_2; X_1, \ldots, X_n) = \ln p(X_1, \ldots, X_n)$, as a summation. Note: it doesn't simplify much.

(c) [3 pts] Express $\frac{\partial \ell}{\partial \theta}$ in terms of θ and $\tau_i, i \in \{1, \ldots, n\}$ and simplify as much as possible. There should be no normal PDFs explicitly in your solution, though the τ_i's may implicitly use them. Hint: Recall that $(\ln f(x))' = \frac{f'(x)}{f(x)}$.

(d) [4 pts] Express $\nabla_{\mu_1}\ell$ in terms of μ_1 and $\tau_i, X_i, i \in \{1,\ldots,n\}$. Do the same for $\nabla_{\mu_2}\ell$ (but in terms of μ_2 rather than μ_1). Again, there should be no normal PDFs explicitly in your solution, though the τ_i's may implicitly use them.
Hint: It will help (and get you part marks) to first write $\nabla_{\mu_1} N_1(x)$ as a function of $N_1(x)$, x, and μ_1.

(e) [3 pts] We conclude: if we know μ_1, μ_2, and θ, we can compute the posteriors τ_i. On the other hand, if we know the τ_i's, we can estimate μ_1, μ_2, and θ by using the derivatives in parts (c) and (d) to find the maximum likelihood estimates. This leads to the following k-means-like algorithm.

- Initialize $\tau_1, \tau_2, \ldots, \tau_n$ to arbitrary values in the range $[0, 1]$.
- Repeat the following two steps.
  1. Update the Gaussian cluster parameters: for fixed values of $\tau_1, \tau_2, \ldots, \tau_n$, update μ_1, μ_2, and θ.
  2. Update the posterior probabilities: for fixed values of μ_1, μ_2 and θ, update $\tau_1, \tau_2, \ldots, \tau_n$.

In part (a), you wrote the update rule for step 2. Using your results from parts (c) and (d), write down the explicit update formulas for step 1.