

Introduction to Machine Learning

Final Exam

Feb 2, 2019

Time limit: 120 minutes

Number of pages: 22

Total points: 89

You can use the back of the pages if you run out of space. Collaboration on the exam is strictly forbidden. Please show *all* of your work and always *justify* your answers. The questions are not necessarily ordered by difficulty; if you cannot solve a question, move on to the next one.

Please write your answers with a *pen*.

(1 point) Please fill in your student ID and full name (LASTNAME, FIRSTNAME) in capital letters.

Please leave the table below empty.

Problem	Maximum points	Obtained
----------------	-----------------------	-----------------

1. Regression

(12 points)

This questions is about weighted linear regression. You are given a dataset consisting of n labeled training points $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$.

In addition, you are given a set of non-negative weights $\{\lambda_1, \dots, \lambda_n\}$, where $\sum_{i=1}^n \lambda_i = 1$. Each weight $\lambda_i \in \mathbb{R}_+$ reflects the importance of correctly estimating the label of a specific training point (\mathbf{x}_i, y_i) .

A common approach towards this task is to find a solution $\mathbf{w} \in \mathbb{R}^d$ which minimizes the *weighted empirical risk* $\hat{R}(\mathbf{w})$, which is defined as follows:

$$\hat{R}(\mathbf{w}) = \sum_{i=1}^n \lambda_i (\mathbf{w}^\top \mathbf{x}_i - y_i)^2.$$

(3 points) (i) *Analytic Solution (MC)*

Let us denote by $\mathbf{X} \in \mathbb{R}^{n \times d}$ the matrix whose rows are $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, $\mathbf{y} \in \mathbb{R}^n$ a row vector whose entries are $\{y_1, \dots, y_n\}$, and let $\mathbf{\Lambda} \in \mathbb{R}^{n \times n}$ be a diagonal matrix such $\mathbf{\Lambda}_{ii} = \lambda_i$

What is the closed form solution for the minimizer $\hat{\mathbf{w}} := \arg \min_{\mathbf{w} \in \mathbb{R}^d} \hat{R}(\mathbf{w})$?

Comment: You may assume that the matrices $\mathbf{X}^\top \mathbf{X}$ and $\mathbf{X}^\top \mathbf{\Lambda} \mathbf{X}$ are invertible.

- $\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{\Lambda} \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{\Lambda} \mathbf{y}$
- $\hat{\mathbf{w}} = \mathbf{\Lambda} (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{\Lambda} \mathbf{y}$
- $\hat{\mathbf{w}} = \mathbf{\Lambda}^{1/2} (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{\Lambda}^{1/2} \mathbf{y}$
- $\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{\Lambda} \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{\Lambda}^{1/2} \mathbf{y}$
- $\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{\Lambda}^{1/2} \mathbf{y}$
- $\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{\Lambda} \mathbf{y}$

(3 points) (ii) *Probabilistic Interpretation (MC)*

Consider the following probabilistic model. Assume that for all i ,

$$y_i = \mathbf{w}^\top \mathbf{x}_i + \epsilon_i,$$

where $\mathbf{w} \in \mathbb{R}^d$ is a fixed (unknown) vector, and $\{\epsilon_1, \dots, \epsilon_n\}$ are statistically independent Gaussian random variables such that

$$\epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$$

where, $\sigma_i > 0$ is the standard deviation. The Maximum Likelihood Estimate (MLE) for this model is defined as follows,

$$\mathbf{w}_{\text{MLE}} := \arg \max_{\mathbf{w}} P(y_1, \dots, y_n | x_1, \dots, x_n, \sigma_1, \dots, \sigma_n, \mathbf{w}).$$

Recall that in class you have shown that if all σ_i 's are the same then solving the above MLE problem is equivalent to minimizing the empirical risk $\arg \min_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^n (\mathbf{w}^\top \mathbf{x}_i - y_i)^2$.

It can be shown that minimizing the weighted empirical risk appearing in the previous problem is equivalent to finding the MLE solution for an appropriate choice of $\sigma_1, \dots, \sigma_n$. What should

the relation between σ_i and λ_i for this equivalence to hold?

- $\lambda_i \propto \sigma_i^{-2}$
- $\lambda_i \propto \sigma_i^{-1}$
- $\lambda_i \propto \sigma_i^{-1/2}$
- $\lambda_i \propto \sigma_i$
- $\lambda_i \propto \sigma_i^{1/2}$
- $\lambda_i \propto \sigma_i^2$
- $\lambda_i \propto \log(1 + \sqrt{\sigma_i})$

In order to improve generalization properties of our model, we introduce a regularization term to the training objective (same weights). This is especially beneficial when you have little data. The cost function becomes,

$$\hat{R}_\eta(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n (\mathbf{w}^\top \mathbf{x}_i - y_i)^2 + \eta C(\mathbf{w}).$$

Two common candidates seen in the course are L_1 (Lasso) and L_2 (Ridge) regularization. These correspond to $C_1(w) = \|\mathbf{w}\|_1$, and $C_2(w) = \|\mathbf{w}\|_2^2$ in the above formula (in place of C) respectively.

(2 points) (iii) *Analytic solution for L_1 (MC)*

Please choose which of the following formulas corresponds to the closed form of the minimizer of the $\hat{R}_\eta(\mathbf{w})$ with $C(\mathbf{w}) = \|\mathbf{w}\|_1$,

- $\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X} + \eta \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$
- $\hat{\mathbf{w}} = (\eta \mathbf{I})^{1/2} (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top (\eta \mathbf{I})^{1/2} \mathbf{y}$
- $\hat{\mathbf{w}} = (\mathbf{X}^\top (\mathbf{I} + \eta \mathbf{I}) \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$
- In general, there is no closed form.

(2 points) (iv) *Regularization limits (T/F)*

Decide whether the following statements are true or false when $\eta \rightarrow \infty$:

True False

- When $C(\mathbf{w}) = \|\mathbf{w}\|_1$, then the solution $\|\hat{\mathbf{w}}\|_2 \rightarrow 0$.
- When $C(\mathbf{w}) = \|\mathbf{w}\|_1$, the regularization has no longer any effect on \hat{w} .
- When $C(\mathbf{w}) = \|\mathbf{w}\|_1$ or $C(\mathbf{w}) = \|\mathbf{w}\|_2^2$ the solution $\|\hat{\mathbf{w}}\|_2 \rightarrow \infty$.
- When $C(\mathbf{w}) = \|\mathbf{w}\|_2^2$ the regularization has no longer any effect on \hat{w} .

(2 points) (v) *Different L_2 regularization (T/F)*

Suppose we use the regularizer $C(\mathbf{w}) = \|\mathbf{w}\|_2^2$ and optimize \hat{R}_{η_1} with a regularization constant η_1 to get the minimizer $\hat{\mathbf{w}}_1$, and \hat{R}_{η_2} with a regularization constant η_2 to get the minimizer $\hat{\mathbf{w}}_2$.

We know that η_2 and η_1 are *arbitrary* and *positive*, and crucially,

$$\eta_2 > \eta_1.$$

Decide which of the following statements are true or false for all possible datasets $\{\mathbf{x}_i, y_i\}_{i=1}^n$:

True **False**

- $\|\hat{\mathbf{w}}_2\|_2 \leq \|\hat{\mathbf{w}}_1\|_2$
- The solution $\hat{\mathbf{w}}_2$ is sparser than $\hat{\mathbf{w}}_1$
- Solutions are the same, i.e. $\hat{\mathbf{w}}_1 = \hat{\mathbf{w}}_2$
- There always exist η_1, η_2 s.t. $\eta_1 \neq \eta_2$ and $\hat{\mathbf{w}}_1 = \hat{\mathbf{w}}_2$.

2. Kernels

(12 points)

(2 points) (i) *Kernelization (T/F)*

Which of the following learning algorithms can be kernelized?

True **False**

- Principal component analysis
- Logistic regression
- K-Means Clustering
- Nearest Neighbour Classification

(2 points) (ii) *Feature Maps (T/F)*

From the lectures, we know that every kernel admits a feature representation in an inner product space such that the kernel can be represented as inner product (for example; if the inner product is in the Euclidean space, $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^\top \phi(\mathbf{y})$). Decide whether the following statements are true or false.

True **False**

- The feature map ϕ induced by a kernel k is always one-to-one.
- The identity map $\phi(x) = x$ defines the linear kernel.
- The dimension of the Euclidean feature map ϕ induced by the cubic kernel $k(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x}^\top \mathbf{y})^3$ where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ grows at least at a polynomial rate in d .
- The radial basis function kernel $k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|_2^2)$ has an infinite-dimensional feature map ϕ .

(2 points) (iii) *Valid Kernels (T/F)*

Let $x, y \in \mathbb{R}$. Let $k_1(x, y)$ and $k_2(x, y)$ be any valid kernel functions on $\mathbb{R} \times \mathbb{R}$. Consider the definitions of the function $f(x, y)$ below. For which of these definitions is f always a valid kernel (True)?

Hint: $\cos(x + y) = \cos(x)\cos(y) - \sin(x)\sin(y)$

True **False**

- $f(x, y) = ck_1(x, y)^2 k_2(x, y)$ for any $c \in \mathbb{R}$
- $f(x, y) = \cos(x - y)$
- $f(x, y) = \frac{1}{k_1(x, y)}$ assuming $k_1(x, y) > 0$ for all $x, y \in \mathbb{R}$
- $f(x, y) = (k_1(x, y) + k_2(x, y))^2$

(2 points) (iv) *Separable space (T/F)*

Consider a dataset consisting of the following four points in \mathbb{R}^2 : $\mathbf{x}^1 = [-1, -1]^\top$, $\mathbf{x}^2 = [-1, 1]^\top$, $\mathbf{x}^3 = [1, -1]^\top$, $\mathbf{x}^4 = [1, 1]^\top$. Class labels for each point are unknown, but assume that each point \mathbf{x}^i may belong to either of only two classes. You apply a feature transformation $\Phi(\cdot)$ to each point. For which of the feature transformations below is the resulting dataset $\{\Phi(\mathbf{x}^1), \Phi(\mathbf{x}^2), \Phi(\mathbf{x}^3), \Phi(\mathbf{x}^4)\}$ guaranteed to be linearly separable (with no point lying exactly on the decision boundary) for every possible class labelling (True)?

Hint: Note that subscript denotes the coordinate in this question, and superscript identifies the datapoint in the dataset.

True	False	
<input type="checkbox"/>	<input type="checkbox"/>	$\Phi(\mathbf{x}) = [\mathbf{x}_1, \mathbf{x}_2, 1]$
<input type="checkbox"/>	<input type="checkbox"/>	$\Phi(\mathbf{x}) = [\mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_1, \mathbf{x}_2, 1]$
<input type="checkbox"/>	<input type="checkbox"/>	$\Phi(\mathbf{x}) = [\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_1^2 + \mathbf{x}_2^2, 1]$
<input type="checkbox"/>	<input type="checkbox"/>	$\Phi(\mathbf{x}) = [\mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_1\mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_2, 1]$

(2 points) (v) *Decision Boundaries (Matching Question)*

You have fitted the following four models to learn a classifier for a multi-class classification problem with three classes:

- A. SVM with kernel $k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$
- B. SVM with kernel $k(\mathbf{x}, \mathbf{y}) = (\gamma \mathbf{x}^\top \mathbf{y} + 1)^3$
- C. SVM with kernel $k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2)$
- D. Nearest neighbour classifier (with five neighbours and uniform weighting)

All SVMs use a *one-vs-one* approach for the multi-class classification and are fitted using the same value for γ . The four figures below show the samples used for fitting all models and the decision boundaries generated by each classifier. Match each model above with its corresponding figures.

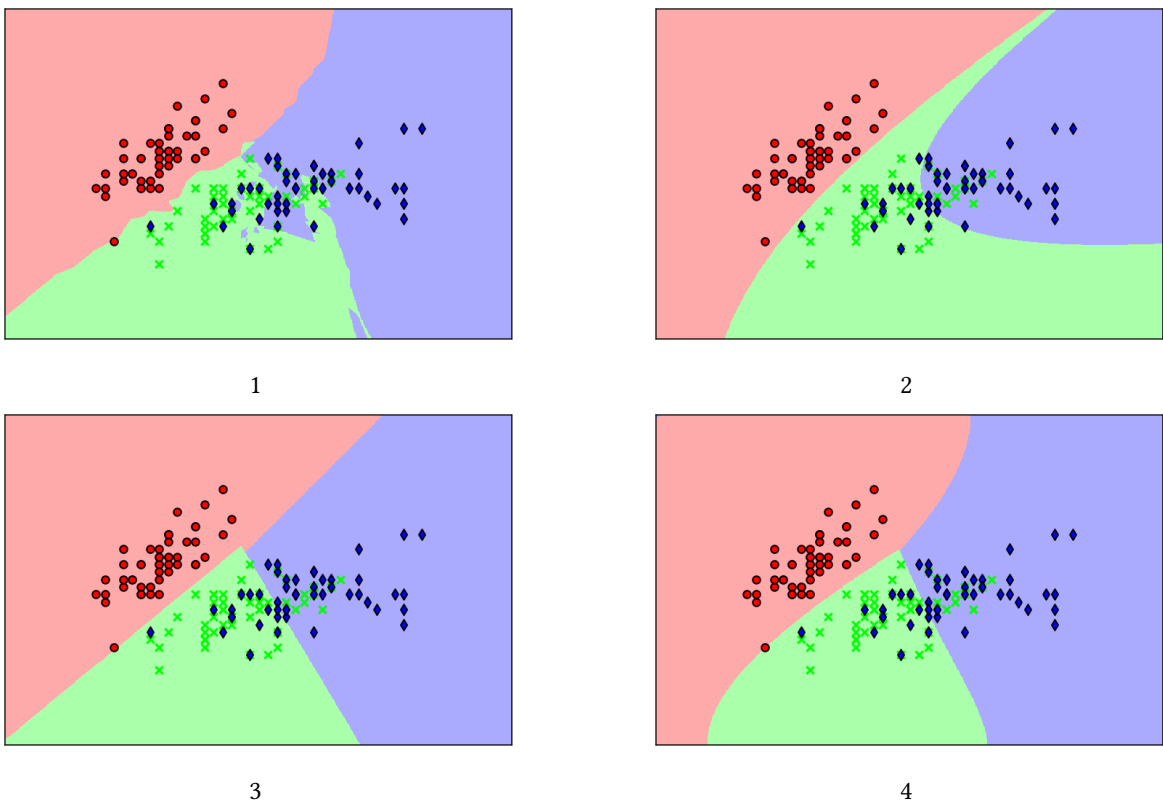
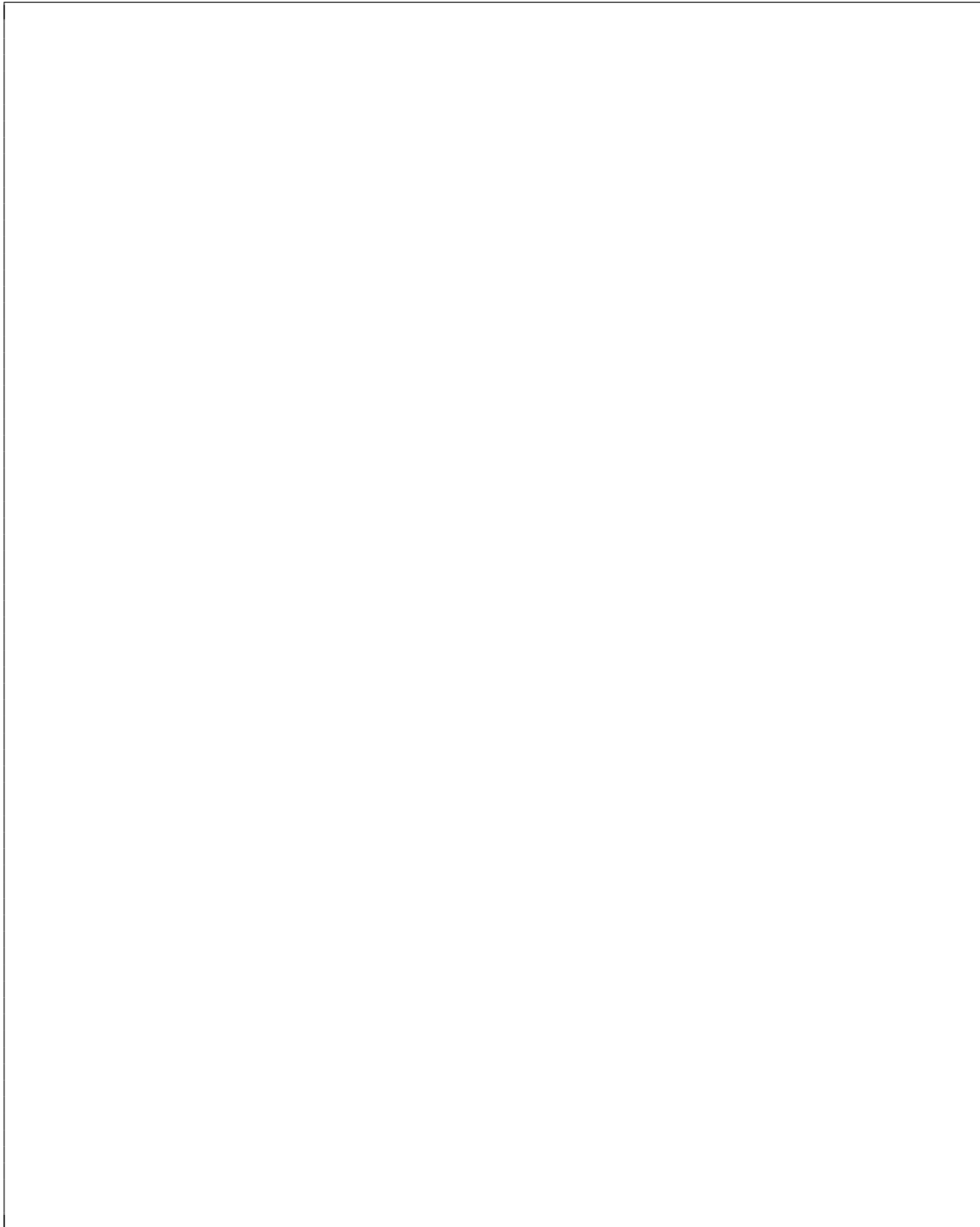


Figure 1: Decision Boundaries

(2 points) (vi) Consider the following function over real-valued scalars x and y :

$$k(x, y) = (1 + cxy)^2,$$

where c is a positive constant. The basis function of this kernel represent the kernel as $k(x, y) = \phi(x)^\top \phi(y)$, where $\phi(x) \in \mathbb{R}^3$. Given that $\phi(x) = [1, \star, cx^2]$, derive the expression that falls under the star.



3. Neural Networks

(12 points)

Consider a one dimensional convolutional neural network given in the figure below. The input to the network has three features x_1 to x_3 each in \mathbb{R} . The network consists of the following layers,

1. a convolution layer using a kernel of size two with weights k_1 and k_2 . The convolution uses no padding and stride of length one,
2. afterwards ReLU ($\text{ReLU}(x) = \max(x, 0)$) is applied to get outputs y_1 and y_2
3. a fully connected layer applied to y_1 and y_2 with the weights w_1 and w_2 to get a scalar output r .

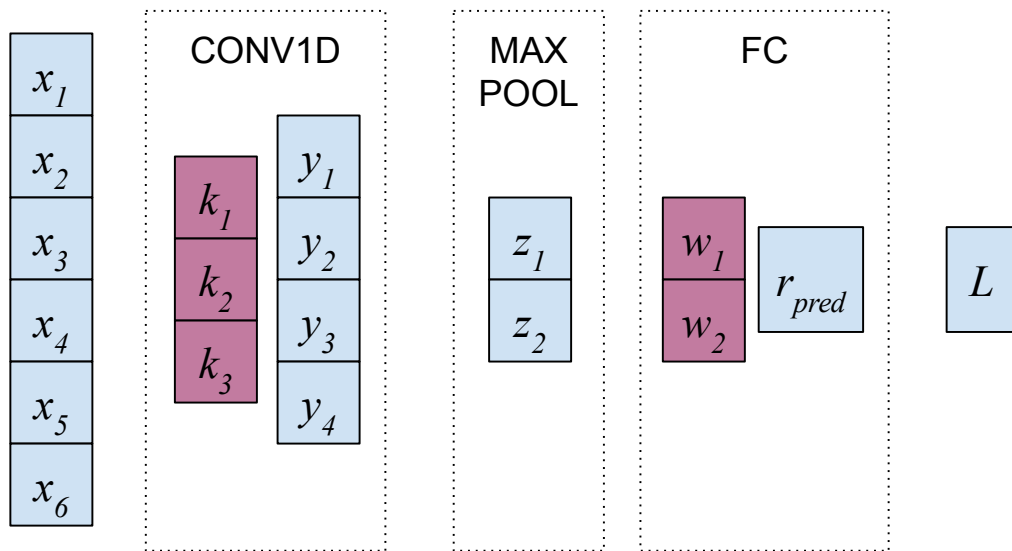


Figure 2: Simple convolutional neural network.

(2 points) (i) *Forward Propagation (Num)*

Suppose all weights (k_1, k_2, w_1, w_2) of the network are initialized to be 0.5. Consider an input example $\mathbf{x} = ([1.0, 1.0, -1.0])^\top$. What is the value of r after forward propagating this input example?

(2 points) (ii) *Backward Propagation (Num)*

Suppose you are given, the same training example and the same initial weights as in the previous question. You want to backpropagate the loss to the input layer. Suppose $\frac{\partial L}{\partial r} = 2$ for the example point, what is $\frac{\partial L}{\partial x_1}$ for the input example.

(2 points) (iii) *Activation Functions (T/F)*

Let $x \in \mathbb{R}$, and $\sigma(x) = \frac{1}{1+\exp(-x)}$ be the sigmoid activation function. Determine which of the following statements are true or false:

True **False**

- ReLU(x) is a non-linear activation function
- The sigmoid activation can be represented as a finite combination of argument scaled ReLU(x). In other words, $\sigma(x) = \sum_{i=1}^k \text{ReLU}(a_i x)$, where $a_i \in \mathbb{R}$.
- The absolute value function $|x|$ can be represented exactly as finite combination of argument scaled ReLU(x). In other words, $|x| = \sum_{i=1}^k \text{ReLU}(a_i x)$, where $a_i \in \mathbb{R}$.
- $\lim_{y \rightarrow \infty} \frac{d\sigma(x)}{dx} |_{x=y} = \infty$, where the vertical rule implies that the function is evaluated at this point.

Having defined the neural network, we would like to optimize the weights of the network. Using the shorthand $\Theta = [k_1, k_2, w_1, w_2]^\top$, we can achieve our goal by minimizing the loss function $L(\Theta)$. The loss function L is assumed to be the mean squared error as follows,

$$L(\Theta) = \frac{1}{n} \sum_{i=1}^n (r(\mathbf{x}_i, \Theta) - y_i)^2.$$

One could use the Gradient Descent (GD) algorithm defined as follows,

$$\Theta_{t+1} = \Theta_t - \eta \nabla L(\Theta_t)$$

where $\eta > 0$ is a fixed learning rate for minimization. As we have seen in the lectures, it is often beneficial to apply the Stochastic Gradient Descent (SGD) update rule instead. With SGD, in each round we pick a single training point, (\mathbf{x}_t, y_t) among the dataset \mathcal{D} , and based on it we compute a gradient estimate, which is an **unbiased** estimate of the true gradient $\nabla L(\Theta_t)$. In order to ensure convergence the parameter (learning rate) η_t in SGD varies.

Comment: The derivative operator is with respect to the variable Θ .

(2 points) (iv) *Learning Rate (T/F)*

Decide which of the following choices for η_t defines a converging SGD method in general (assume the losses are bounded).

True **False**

- $\eta_t \propto \log(t)$
- $\eta_t \propto \frac{1}{t}$
- $\eta_t \propto \min(0.1, \frac{1}{t})$
- $\eta_t \propto e^t$

(2 points) (v) *SGD vs GD (T/F)*

Decide which of these statements are true or false about the comparison of SGD and GD.

True **False**

- One step of SGD is more computationally costly than GD.
- The learning rate for SGD needs to be always bigger than the learning rate for GD.

- The learning rate for GD can be chosen to be any constant for it to converge whereas for SGD we need to change it over time.
- If we pick the points uniformly at random, after n iterations (N.B. n is the number of data points) of SGD we would have always picked all data points at least once.

(1 point) (vi) *SGD uniform (MC)*

Assume that in each round we draw a sample (\mathbf{x}_t, y_t) *uniformly at random* from the dataset \mathcal{D} . What is the appropriate SGD update rule that preserves the above mentioned unbiasedness property?

- $\Theta_{t+1} = \Theta_t - \eta_t (\nabla L(\Theta_t) - \nabla(r(\mathbf{x}_t, \Theta_t) - y_t)^2)$
- $\Theta_{t+1} = \Theta_t - 2\eta_t(r(\mathbf{x}_t, \Theta_t) - y_t)$
- $\Theta_{t+1} = \Theta_t - \eta_t \nabla(r(\mathbf{x}_t, \Theta_t) - y_t)^2$
- $\Theta_{t+1} = \Theta_t - \eta_t \nabla(L(\Theta_t) - r(\mathbf{x}_t, \Theta_t) - y_t)^2$

(1 point) (vii) *SGD sampling (MC)*

Assume that in each round we draw a sample (\mathbf{x}_t, y_t) such that

$$P((\mathbf{x}_t, y_t) \text{ is sampled}) = \lambda_t, \quad \forall t \in \{1 \dots, N\}$$

What is the appropriate SGD update rule that preserves the above mentioned unbiasedness property?

- $\Theta_{t+1} = \Theta_t - \eta_t \lambda_t (\nabla L(\Theta_t) - \nabla(r(\mathbf{x}_t, \Theta_t) - y_t)^2)$
- $\Theta_{t+1} = \Theta_t - \frac{2}{\lambda_t} \eta_t (r(\mathbf{x}_t, \Theta_t) - y_t)$
- $\Theta_{t+1} = \Theta_t - \frac{\eta_t}{\lambda_t} \nabla(r(\mathbf{x}_t, \Theta_t) - y_t)^2$
- $\Theta_{t+1} = \Theta_t - \frac{\eta_t}{\lambda_t} \nabla(L(\Theta_t) - r(\mathbf{x}_t, \Theta_t) - y_t)^2$

4. Classification

(11 points)

In this task we explore a simple classification problem where we would like to create a classifier that can classify whether something is a chocolate or not. In order to do so, we obtained a small training set $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{20}$. In other words, we have 20 data points with feature vectors \mathbf{x}_i in two dimensions (color and sweetness). The label associated with being a chocolate is $y = 1$ (dot in the figure) and the label associated with not being chocolate is $y = -1$ (cross in the figure).

We choose to perform SVM classification with the following objective (where we allow for slack variables):

$$\min_{w \in \mathbb{R}^2, \{\xi_i \geq 0\}_{i=1}^N} \|w\|_2^2 + C \sum_{i=1}^N \xi_i$$

subject to $y_i(w^\top x_i) \geq 1 - \xi_i \forall i \in \{1, \dots, N\}$.

(2 points) (i) *Optimal slack variables (MC)*

In the following figure you can see the optimal solution for the separating hyperplane w^* for a particular choice of C . In the figure the dashed lines are defined such that $x^\top w^* = \pm 1$ respectively.

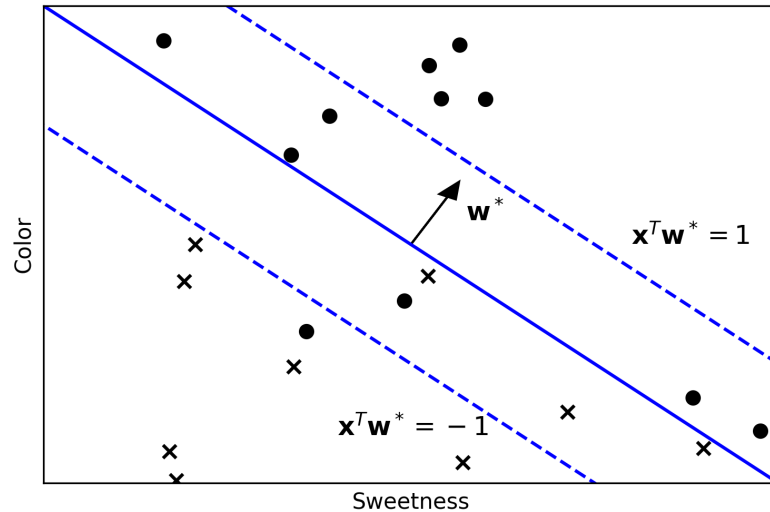


Figure 3

According to the figure, how many of the optimized slack variables $\{\xi_i^*\}_{i=1}^N$ are strictly positive?

- 2
- 9
- 10
- 3

(2 points) (ii) *Soft SVM (TF)*

For the dataset visualized in the above figure, determine which of the following are true or false.

True **False**

- For $C = 0$ the above optimization problem has more than one solution.
 - For $C = \infty$ the above optimization problem is infeasible.
 - For $0 < C < \infty$ the optimization problem has a unique solution.
 - The dataset is linearly separable for a specific value of C .
-

We have sent the classifier trained on the set of 20 points to the test facility. They tested it on a test set with 150 examples. Test facility summarized its findings in the following confusion matrix:

		True	
		chocolate	not chocolate
Predicted	chocolate	60	30
	not chocolate	20	40

(1 point) (iii) *Accuracy (Num)*

Calculate the accuracy given the above confusion matrix. (Tolerance ± 0.01 , i.e. you can round up or down.)

(1 point) (iv) *Precision (Num)*

Calculate the precision given the above confusion matrix. (Tolerance ± 0.01 , i.e. you can round up or down.)

(1 point) (v) *Recall (Num)*

Calculate the recall given the above confusion matrix. (Tolerance ± 0.01 , i.e. you can round up or down.)

(1 point) (vi) *TPR (Num)*

Calculate the True Positive rate (TPR) given the above confusion matrix. (Tolerance ± 0.01 , i.e. you can round up or down.)

(1 point) (vii) *FPR (Num)* Calculate the False Positive rate (FPR) given the above confusion matrix. (Tolerance ± 0.01 , i.e. you can round up or down.)

(2 points)(viii) *ROC curve (T/F)*

To assess the performance of our algorithm, one can compute a Receiver Operator Characteristic (ROC). Determine the validity of the following statements related to the ROC curve of the above algorithm (not shown).

True False

- The test dataset is balanced.
- The presented algorithm is below the ROC curve of an algorithm that would perform random guessing.
- We could influence the position of the learned classifier on the plot by changing the value of C .
- The classifier can achieve the accuracy of 1 for a specific value of C on the test dataset.

5. Clustering

(11 points)

The *k-means problem* is the problem of clustering data represented as a matrix $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$ into k different clusters.

The clusters are defined by their so-called *centroids* $\{\mu_1, \dots, \mu_k\}$ where each $\mu_i \in \mathbb{R}^d$. Let $\mathbf{z} \in \{1 \dots k\}^n$ be a vector representing the assignments of the data points to clusters. In other words, a data point \mathbf{x}_i is assigned to cluster z_i .

(1 point) (i) *Basics (MC)*

What is the objective function of the k-means optimization problem?

- $\sum_{i=1}^N \|\mathbf{x}_i - \mu_{z_i}\|_2^2$
- $\sum_{i=1}^k \|\mathbf{x}_i - \mu_{z_i}\|_2^2$
- $\sum_{i=1}^N \|\mathbf{x}_{z_i} - \mu_i\|_2^2$
- $\sum_{i=1}^k \|\mathbf{x}_{z_i} - \mu_i\|_2^2$

(2 points) (ii) *Lloyd's heuristic (T/F)*

Which of these statements about Lloyd's heuristic are true or false?

True **False**

- It always converges in polynomial time.
- It always converges.
- It always converges to a local optimum.
- The solution can be arbitrarily bad compared to the global optimum.

(2 points) (iii) *k-means++ (T/F)*

Which of these statements about the k-means++ initialization are true or false?

True **False**

- It chooses the initial centroids deterministically.
- It chooses the initial centroids all at the same time but refines them sequentially if they are the same.
- It chooses the initial centroids sequentially, where every new centroid is dependent on all the other already chosen ones.
- The solution can be arbitrarily bad compared to the optimal centroids in expectation.

(2 points) (iv) *Criteria (T/F)*

Which of these methods can be used to choose the number of clusters k ?

True **False**

- Prior knowledge
- The "elbow criterion"
- Minimizing the k-means objective w.r.t. k on the training set
- Minimizing the k-means objective w.r.t. k on a held-out validation set

Assume we have five one-dimensional points: $x_1 = 0, x_2 = 1, x_3 = 1, x_4 = 3, x_5 = 5$. We want to cluster these points using the k-means algorithm (Lloyd's heuristic).

- (4 points)** (i) For $k = 2$, if we initialize the centroids as $\mu_1 = 0$ and $\mu_2 = 5$, which solution (in terms of μ_1, μ_2 , and \mathbf{z}) will the algorithm converge to?

Hint: For example, $z_1 = 1$. Use the decimal point format. (Tolerance ± 0.01 , i.e. you can round up or down.)

6. Principal Component Analysis

(10 points)

In dimension reduction, we want to embed high dimensional data from a space \mathbb{R}^d to a space \mathbb{R}^k with $k \ll d$. We therefore want to represent the data $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$ with the respective lower-dimensional embeddings $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_n]^\top \in \mathbb{R}^{n \times k}$. One particular method for dimensionality reduction is the *Principal Component Analysis* (PCA), where the embeddings are given by $\mathbf{z}_i = \mathbf{W}^\top \mathbf{x}_i$, where $\mathbf{W} \in \mathbb{R}^{d \times k}$.

(2 points) (i) *PCA general (T/F)*

Determine whether the following statements about PCA are true or false.

True **False**

- PCA is a linear dimension reduction method.
- The first principal component is the eigenvector of the data covariance matrix with the smallest eigenvalue.
- All principal components are mutually orthogonal.
- PCA is regarded as a supervised learning technique.

(2 points) (ii) *PCA general II (T/F)*

Suppose $k = 1$, and hence \mathbf{W} becomes a column vector \mathbf{w} . Which of these objective functions is a valid PCA objective:

True **False**

- $\arg \min_{\mathbf{w}, \|\mathbf{w}\|_2=1, \{z_i\}_{i=1}^n} \sum_{i=1}^n \|\mathbf{w}z_i - \mathbf{x}_i\|_2^2$
- $\arg \min_{\mathbf{w}, \|\mathbf{w}\|_2=1} \sum_{i=1}^n \|\mathbf{w}\mathbf{w}^\top \mathbf{x}_i - \mathbf{x}_i\|_2^2$
- $\arg \min_{\mathbf{w}, \|\mathbf{w}\|_2=1} \sum_{i=1}^n \|\mathbf{w}^\top \mathbf{x}_i - \mathbf{w}^\top (\frac{1}{n} \sum_{i=1}^n \mathbf{x}_i)\|_2^2$
- $\arg \min_{\mathbf{w}, \{z_i\}_{i=1}^n} \frac{1}{n} \sum_{i=1}^n \|\mathbf{w}z_i - \mathbf{x}_i\|_2^2$

(2 points) (iii) *PCA general III (T/F)*

When $k > 1$, the matrix \mathbf{W}

True **False**

- $\mathbf{W}^\top \mathbf{W}$ is the identity
- \mathbf{W} is symmetric
- \mathbf{W} is diagonal
- $\mathbf{W}\mathbf{W}^\top$ is the identity

(2 points) (iv) *Short questions on kernel PCA (T/F)*

Determine whether the following statements about kernel PCA are true or false.

True **False**

- Kernel PCA is equivalent to PCA when used with linear kernels.
- Kernel PCA can only identify invariant linear subspaces.
- The complexity of kernel PCA is independent of the number of data points.
- The kernel is often centered as a preprocessing step.

(2 points) (v) *Short questions on neural network autoencoders (T/F)*

Determine whether the following statements about neural network autoencoders are true or false.

True False

- A neural network autoencoder cannot model nonlinear manifold structures.
- In general, the performance of a neural network autoencoder depends on the initialization of the weights before the optimization.
- At the global optimum, a neural network autoencoder with linear activations and squared loss is equivalent to PCA.
- Training of an autoencoder with linear activations is a convex optimization problem.

7. Linear Discriminant Analysis

(10 points)

In this problem we would like to perform classification using linear discriminant analysis (LDA). We assume the following simplified model,

$$P(X|Y = j) = \mathcal{N}(\mu_j, \mathbf{I})$$

where $\mathbf{I} \in \mathbb{R}^{d \times d}$ is an identity matrix and $j \in \{1, 2\}$. Furthermore, we parametrize $P(Y = 1) = p$, and naturally $P(Y = 2) = 1 - p$.

Consider a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where n_j represents the number of samples such that $y = j$.

(1 point) (i) *MLE Probability (Num)*

Assuming that $n_1 = 1$ and $n_2 = 3$ calculate the MLE estimate of p , \hat{p} .

(1 point) (ii) *MAP Probability (Num)*

An expert on the problem at hand has told us that he is sure that $p = 0.5$ or $p = 0.25$ with equal probability. Assuming this as a prior belief, calculate the maximum a-posteriori estimate of p , \hat{p} :

(1 point) (iii) *MLE II (MC)*

Which of the following expresses the MLE estimate $\hat{\mu}_1$ of μ_1 ?

- $\hat{\mu}_1 = \frac{1}{n_1} \sum_{i, \text{s.t. } y_i=1} \mathbf{x}_i$
- $\hat{\mu}_1 = \frac{n_1}{n_1+n_2} \sum_{i, \text{s.t. } y_i=1} \mathbf{x}_i$
- $\hat{\mu}_1 = \frac{n_1}{n_2} \sum_{i, \text{s.t. } y_i=1} \mathbf{x}_i$
- Cannot be derived from the information given.

(2 points) (iv) *Decision rule (MC)*

Suppose that we have the estimates of \hat{p}_j and $\hat{\mu}_j$ for all j . The LDA provides a decision rule that given an \mathbf{x} , it can predict its class. Which of the following is a valid procedure to predict the class label of \mathbf{x} using LDA.

Comment: $\hat{p}_1 = \hat{p}$ and $\hat{p}_2 = 1 - \hat{p}$.

- $\hat{y} = \arg \max_{j \in \{1,2\}} \exp\left(-\frac{\hat{p}_j \|\mathbf{x} - \hat{\mu}_j\|_2^2}{2}\right)$
- $\hat{y} = \arg \max_{j \in \{1,2\}} (\mathbf{x} - \hat{\mu}_j)^\top \hat{\mu}_j + \hat{p}_j$
- $\hat{y} = \arg \max_{j \in \{1,2\}} (2\mathbf{x} - \hat{\mu}_j)^\top \hat{\mu}_j + 2 \log(\hat{p}_j)$
- $\hat{y} = \arg \max_{j \in \{1,2\}} \exp\left(-\frac{\|\mathbf{x} - \hat{\mu}_j\|_2^2}{2\hat{p}_j}\right)$
- $\hat{y} = \arg \max_{j \in \{1,2\}} \exp\left(-\frac{\|\mathbf{x} - \hat{\mu}_j\|_2^2}{\hat{p}_j}\right)$
- $\hat{y} = \arg \max_{j \in \{1,2\}} (\mathbf{x} - \hat{\mu}_j)^\top \hat{\mu}_j + \log(\hat{p}_j)$

(1 point) (v) *Decision rule II (MC)*

If $\mathbf{x}^\top(\hat{\mu}_1 - \hat{\mu}_2) > \frac{1}{2}(\hat{\mu}_1^\top \hat{\mu}_1 - \hat{\mu}_2^\top \hat{\mu}_2)$, \mathbf{x} is classified as

- 1 for any \hat{p}
- 2 for any \hat{p}
- 2 if $\hat{p} = 0.5$
- 1 if $\hat{p} = 0.5$

(2 points) (vi) *Decision Theory (MC)*

Suppose that there is a cost associated with predicting class 1 wrong which is α times more than the cost associated with predicting the class 2 wrong. Naturally, predicting correctly incurs no cost.

Assuming that we want to minimize the expected cost of the decision policy and $\hat{p} = 0.5$, which of the following is the right decision boundary for the problem when using LDA from the previous question.

- $0 = \mathbf{x}^\top(\hat{\mu}_1 - \hat{\mu}_2) - \frac{1}{2}(\hat{\mu}_1^\top \hat{\mu}_1 - \hat{\mu}_2^\top \hat{\mu}_2) + \log(\alpha)$
- $0 = \mathbf{x}^\top(\hat{\mu}_1 - \hat{\mu}_2) - \frac{1}{2}(\hat{\mu}_1^\top \hat{\mu}_1 - \hat{\mu}_2^\top \hat{\mu}_2) + \alpha$
- $0 = \mathbf{x}^\top(\hat{\mu}_1 - \hat{\mu}_2) - \frac{1}{2}(\hat{\mu}_1^\top \hat{\mu}_1 - \hat{\mu}_2^\top \hat{\mu}_2)$
- It cannot be decided without the knowledge of the cost of predicting the class 2 incorrectly.
- $0 = \mathbf{x}^\top(\hat{\mu}_1 - \hat{\mu}_2) - \frac{1}{2}(\hat{\mu}_1^\top \hat{\mu}_1 - \hat{\mu}_2^\top \hat{\mu}_2) + \log(\hat{p}_1 - \alpha\hat{p}_2)$
- $0 = \mathbf{x}^\top(\hat{\mu}_1 - \hat{\mu}_2) - \frac{1}{2}(\hat{\mu}_1^\top \hat{\mu}_1 - \hat{\mu}_2^\top \hat{\mu}_2) + 2(1 - \alpha)$

(2 points) (vii) *LDA vs QDA (TF)*

In the lectures, you have compared LDA with quadratic discriminant analysis (QDA) and logistic regression. Determine which of the following statements are true or false:

True False

- Had we assumed a model $P(X|Y = j) = \mathcal{N}(\mu_j, \Sigma)$ for each j , where Σ is not diagonal, LDA would arrive at a quadratic decision boundary.
- LDA and logistic regression have the same assumptions.
- When the assumptions of logistic regression and LDA are met they share the same decision boundary.
- Assuming a two class problem, QDA needs to estimate at least $\frac{(d+1)d}{2}$ parameters while LDA only d .

8. Gaussian Mixture Models and EM Algorithm

(10 points)

Suppose you have a dataset $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, where each $\mathbf{x}_i \in \mathbb{R}^d$ is generated with a Gaussian mixture model (GMM) with k components with weights $\{w_1 \dots w_k\}$, means $\{\mu_1, \dots, \mu_k\}$, and covariance matrices $\{\Sigma_1, \dots, \Sigma_k\}$.

(1 point) (i) *Basics (MC)*

Which of the following independence assumptions is encoded in the Gaussian mixture model?

- The points \mathbf{x}_i are independent of each other.
- The points \mathbf{x}_i are dependent on each other when conditioned that they were sampled from the same mixture component.
- The points \mathbf{x}_i need to be dependent of each other.
- The points \mathbf{x}_i can be but not need to be dependent of each other.

(1 point) (ii) *Basics II (Form)*

What is the a priori probability that a point from the dataset belongs to the i -th center.

Hint: The answer is a symbol not a text and has already been introduced.

Suppose you do not know the weights, means, nor covariance matrices of the GMM but you know the number of mixture components k used to generate the dataset. To learn the parameters you decide to run the *soft*-EM algorithm.

(1 point) (iii) *E-step (MC)*

Let

$$\theta = (w_1 \dots w_k, \mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k)$$

be the parameters from the previous M-step and z_i be the assignment of point \mathbf{x}_i to a mixture component taking values in $\{1 \dots k\}$. Capital letters denote random variables. In the E-step of the algorithm you compute:

- $P(X = x_i | Z_i = j, \theta)$ for all $i \in \{1 \dots n\}$ and $j \in \{1 \dots k\}$.
- $P(Z_i = j | X = x_i, \theta)$ for all $i \in \{1 \dots n\}$ and $j \in \{1 \dots k\}$.
- $P(Z_i = j, \theta)$ for all $i \in \{1 \dots k\}$ and $j \in \{1 \dots k\}$.
- $P(X = x_i, \theta)$ for all $i \in \{1 \dots n\}$.

(1 point) (iv) *M-step (MC)*

In the M step of the EM algorithm applied to GMMs you:

- maximize the marginal data log-likelihood, where the marginalization is taken with respect to the distribution calculated in the E-step.
- maximize the expected joint data log-likelihood, where the expectation is taken with respect to the distribution calculated in the E-step.
- maximize the expected joint data log-likelihood, where the expectation is taken with respect to the true distribution $P(w|\mathbf{x})$.
- maximize the marginal data log-likelihood, where the marginalization is taken with respect to the true distribution $P(w|\mathbf{x})$.

Your friend comes up with an unfair four-sided die which takes values in $\{1, 2, 3, 4\}$ and asks your help to estimate its properties.

She knows that the die takes value 1 with probability $\frac{1}{4}$, 2 with probability ε , 3 with probability 2ε and 4 with probability $\frac{3}{4} - 3\varepsilon$. She already tried to estimate the probabilities and she recorded the outcomes of rolling the die. She observed that the side 4 appeared x_4 times and the side 3 appeared x_3 times. She also observed that the side 1 and side 2 appeared x_{12} times in total, unfortunately she did not note how many times 1 and 2 appeared separately. In other words, x_1 and x_2 are unknown values with $x_1 + x_2 = x_{12}$ and she only knows x_{12} . We can model the unknown occurrences of side 1 and 2 by two random variables X_1 and X_2 , respectively. These two random variables then satisfy the constraint $X_1 + X_2 = x_{12}$. You immediately realize that the EM algorithm can be used to get a MLE for ε and you decide to calculate it.

Comment: Simple MLE estimate can be used to solve this problem by ignoring some information, but due to data efficiency we choose to perform EM. Also, note that capital letters denote random variables.

(2 points) (i) *EM on a tetrahedron die - E-Step (MC)*

What is $r_1 = \mathbb{E}[X_1|\varepsilon]$ and $r_2 = \mathbb{E}[X_2|\varepsilon]$ respectively?

- $x_{12} \frac{1/4}{1/4+\varepsilon}, \quad x_{12} \frac{\varepsilon}{1/4+\varepsilon}$
- $x_{12} \frac{\varepsilon}{1/2+\varepsilon}, \quad x_{12} \frac{\varepsilon}{1/2+\varepsilon}$
- $x_{12} \frac{\varepsilon}{1+\varepsilon}, \quad x_{12} \frac{\varepsilon/4}{1+\varepsilon}$
- $\varepsilon \frac{x_{12}}{1/4+x_1}, \quad \varepsilon \frac{x_{12}}{1/4+x_2}$
- $\varepsilon \frac{x_{12}+\varepsilon}{1/4+\varepsilon}, \quad \varepsilon \frac{x_{12}+\varepsilon}{1/4+\varepsilon}$
- $\varepsilon \frac{x_{12}}{1/2+\varepsilon}, \quad \varepsilon \frac{x_{12}}{1/4+\varepsilon}$
- $\varepsilon \frac{x_{12}+1/4}{1/2+\varepsilon}, \quad \varepsilon \frac{x_{12}+1/4}{1/4+\varepsilon}$

(4 points) (ii) *EM on a tetrahedron die - M-Step (MC)*

What is the MLE of ε ? You can now use the expected values of X_1 and X_2 , r_1 and r_2 respectively.

- $\frac{x_{12}-r_2+x_3}{4(x_{12}-r_2+x_3+x_4)}$
- $\frac{x_{12}+r_1+x_3}{4(x_{12}+x_3+x_4)}$
- $\frac{x_{12}+x_4}{4(x_{12}+x_3+x_4)}$
- $\frac{r_1+r_2+x_4}{4(x_{12}-x_3+x_4)}$
- $\frac{r_1+x_4}{4(x_{12}+x_3-x_4)}$
- $\frac{r_1+r_2+x_4}{4(x_{12}+r_3+x_4)}$
- $\frac{r_1+r_2+x_4}{4(x_{12}+x_3+r_4)}$
- $\frac{x_4}{4(x_{12}+x_3+r_4)}$
- $\frac{r_1+r_2}{4(x_{12}+x_3+r_4)}$
- $\frac{x_{12}-r_1+x_3}{4(x_{12}-r_1+x_3+x_4)}$