CSCI 567 Fall 2021 Quiz Two

Problem	1	2	3	4	5	Total
Max	30	12	14	16	28	100
Points						

Please read the following instructions carefully:

- Use the same link for the lectures to join Zoom for the quiz (available on both the course website and the DEN website). You will be assigned to a breakout room. Each room has one proctor, and you have to **turn your camera on** during the entire quiz. If you finish early, you can submit your quiz and leave the meeting, but you cannot rejoin.
- This is an open-book/note quiz. Searching the Internet is technically not forbidden, but you are strongly encouraged to focus on solving the problems yourself. **Collaboration of any kind or consultation from others, however, is strictly prohibited.** Violation will lead to **0 score** for the entire quiz and be reported to SJACS.
- For all questions, write down your answers as you would normally do for the written assignments, and then submit a pdf/jpg/png for each of the 5 problems. Make sure that your writing is recognizable. Try to keep you solutions concise.
- Duration of the exam is 2.5 hours plus extra 10 minutes for scanning and uploading. **If you are x minutes late, your grade for this quiz will suffer x% penalty (no excuses please).** Questions are **not ordered by their difficulty**, so budget your time for each question carefully.
- If you have any **clarification** questions during the quiz, make a private post on Piazza and it will be answered shortly.

1 Multiple-Choice Questions (30 points)

IMPORTANT: *Select ALL answers that you think are correct. You get 0.5 point for selecting each correct answer and similarly 0.5 point for not selecting each incorrect answer.*

(1) Which of the following about decision trees is correct?

(A) Good interpretability is a key advantage of decision trees.
(B) Decision tree algorithms are usually implemented using recursion.
(C) Entropy is the only way to measure the uncertainty of a node when building a decision tree.
(D) Regularization is not applicable to decision trees since they do not minimize a certain loss function.

Answer: AB

(2) Consider a binary dataset with 50 positive examples 50 negative examples. Decision stump $\mathcal{T}_1$ splits this dataset into two children where the left one has 20 positive examples and 40 negative examples, while another decision stump $\mathcal{T}_2$ results in a left child with 25 positive examples and 25 negative examples. Which of the following is correct?

(A) The entropy of the left child of $\mathcal{T}_1$ is $\frac{1}{3}\ln 3 + \frac{2}{3}\ln\frac{3}{2}$.
(B) The entropy of the right child of $\mathcal{T}_1$ is $\frac{1}{4}\ln 4 + \frac{3}{4}\ln\frac{4}{3}$.
(C) The entropy of either child of $\mathcal{T}_2$ is the maximum possible entropy for two classes.
(D) Based on conditional entropy, $\mathcal{T}_2$ is a better split than $\mathcal{T}_1$.

Answer: ABC. (D) is wrong simply based on (C) — no need to actually compute and compare their conditional entropy.

(3) Which of the following about boosting is correct?

(A) Boosting is guaranteed to achieve zero training error.
(B) AdaBoost never overfits.
(C) AdaBoost is greedily minimizing the exponential loss.
(D) The distribution D_{t+1} computed by AdaBoost at the end of round t is such that the weak classifier h_t is only as good as random guessing with respect to it.

Answer: CD

(4) Which of the following is a possible application of clustering?

(A) Compressing images.
(B) Discovering communities in a social network.
(C) Accelerating downstream algorithms.
(D) Finding similar customers in market research.

Answer: ABCD

(5) Which of the following about the convergence of K-means is correct?

(A) K-means always finds the global minimum of the K-means objective, but it might take very long time.

(B) K-means always finds a local minimum of the K-means objective.
(C) K-means++ always finds the global minimum of the K-means objective, but it might take very long time.
(D) In expectation K-means++ finds a good approximation of the global minimum of the K-means objective.

Answer: BD

(6) Which of the following about Gaussian Mixture Model (GMM) is correct?

(A) GMM assumes that the data are generated stochastically in a particular way, and thus can only be applied if we know the training data are indeed generated from this model.
(B) The MLE of a GMM model can be found using the EM algorithm.
(C) Learning a GMM via EM gives not only the cluster (soft) assignments and centers, but also other parameters such as mixture weights and covariance matrices.
(D) GMM for clustering always performs better than K-means since it learns more parameters.

Answer: C

(7) Which of the following about density estimation is correct?

(A) Density estimation is an unsupervised learning problem.
(B) Density estimation always requires assuming a generative model to start with.
(C) Kernel density estimation has no hyperparameters.
(D) Kernel density estimation is a non-parametric method.

Answer: AD

(8) Which of the following about Naive Bayes is correct?

(A) Naive Bayes assumes that different features are all independent of each other.
(B) Naive Bayes with a Gaussian model learns a linear classifier if the variance is fixed.
(C) Naive Bayes with a Gaussian model learns a quadratic classifier if the variance is also a parameter.
(D) Naive Bayes and logistic regression learn the same model via different methods.

Answer: BC

(9) Which of the following about Principal Component Analysis (PCA) is correct?

(A) PCA can be used to compress a dataset.
(B) PCA is useful for visualizing a dataset.
(C) PCA requires finding all eigenvectors of the covariance matrix.
(D) PCA requires finding all eigenvalues if we want to find enough principal components to cover a certain percentage of the spectrum.

Answer: AB

(10) Which of the following about kernel PCA is correct?

(A) Kernel PCA requires centering the original dataset.
(B) Kernel PCA requires rescaling the L2 norm of an eigenvector to $1/\lambda$ where λ is the corresponding eigenvalue.
(C) Running kernel PCA is always slower than running standard PCA.

(D) The output of kernel PCA is not a linear transformation of the original dataset.

Answer: D

(11) In a Hidden Markov Model (HMM), what can we infer using only the forward and backward messages computed from a certain sequence of observations $x_1, \ldots, x_T$?

(A) The distribution of the state at a particular time step given $x_1, \ldots, x_T$.
(B) The most likely last state given $x_1, \ldots, x_T$.
(C) The most likely state sequence given $x_1, \ldots, x_T$.
(D) The probability of seeing $x_1, \ldots, x_T$.

Answer: ABD. Note that (B) can be found by $\text{argmax}_s \, \alpha_s(T)$.

(12) In an HMM, which of the following quantities is equal to $P(X_{1:T} = x_{1:T})$? (α and β are forward and backward messages respectively, and a and b are model parameters as discussed in the lecture.)

(A) $\sum_s \alpha_s(1)\beta_s(1)$
(B) $\sum_s \alpha_s(T)\beta_s(T)$
(C) $\sum_{s,s'} \alpha_s(1)a_{s,s'}b_{s',x_2}\beta_{s'}(2)$
(D) $\sum_{s,s'} \alpha_s(T-1)a_{s,s'}b_{s',x_T}$

Answer: ABCD

(13) Suppose that $z_1^*, \ldots, z_T^*$ is the output of the Viterbi algorithm given a sequence of observations $x_1, \ldots, x_T$. Which of the following is NOT correct?

(A) z_1^* is the most likely first state given $x_1, \ldots, x_T$.
(B) z_1^* is the most likely first state given x_1.
(C) z_T^* is the most likely last state given $x_1, \ldots, x_T$.
(D) z_T^* is the most likely last state given x_T.

Answer: ABCD

(14) In a multi-armed bandit problem with two arms and binary rewards, suppose that we have selected the first arm 6 times, 3 of which yield reward 1, and the second arms 3 times, 2 of which yield reward 1. Which of the following about the behavior of the UCB algorithm in the next round is correct?

(A) UCB selects the second arm because $\frac{1}{2} + 2\sqrt{\frac{\ln 10}{6}} < \frac{2}{3} + 2\sqrt{\frac{\ln 10}{3}}$.
(B) UCB selects the second arm because $\frac{1}{2} + 2\sqrt{\frac{\ln 10}{3}} < \frac{2}{3} + 2\sqrt{\frac{\ln 10}{2}}$.
(C) UCB selects the second arm because $\frac{1}{2} + 2\sqrt{\frac{\ln 5}{3}} < \frac{2}{3} + 2\sqrt{\frac{\ln 5}{2}}$.
(D) UCB selects the second arm with probability $\frac{2}{3}$.

Answer: A

(15) Which of the following about reinforcement learning is correct?

(A) Reinforcement learning with S states is simply a combination of S multi-armed bandit problems.
(B) Value iteration always converges, but there is no guarantee on how long it will take.
(C) Both model-based and model-free methods need to address the exploration-exploitation trade-off.
(D) Q-learning learns the Q function directly without estimating the model parameters.

Answer: CD

2 SVM (12 points)

Consider a dataset consisting of points in the form of (x, y), where x is a real value, and $y \in \{-1, 1\}$ is the label. There are only three points $(x_1, y_1) = (-1, -1)$, $(x_2, y_2) = (1, -1)$, and $(x_3, y_3) = (0, 1)$.

(a) Given a kernel function $k(x, x') = (1 + xx')^2$, write down the dual formulation of SVM for this dataset with hyperparameter C set to $+\infty$ (you have to plug in the actual data instead of just showing the generic dual formulation). (3 points)

The general dual formulation (with $C = +\infty$) is:

$$\max_{\alpha} \quad \sum_n \alpha_n - \frac{1}{2} \sum_{m,n} y_m y_n \alpha_m \alpha_n k(\boldsymbol{x}_m, \boldsymbol{x}_n)$$
$$\text{s.t.} \quad \alpha_n \geq 0, \ \forall n \quad \text{and} \quad \sum_n \alpha_n y_n = 0.$$

Plugging in the specific dataset gives:

$$\max_{\alpha_1, \alpha_2, \alpha_3 \geq 0} \quad \alpha_1 + \alpha_2 + \alpha_3 - \tfrac{1}{2}\left(4\alpha_1^2 + 4\alpha_2^2 + \alpha_3^2 - 2\alpha_1\alpha_3 - 2\alpha_2\alpha_3\right)$$
$$\text{s.t.} \quad \alpha_1 + \alpha_2 = \alpha_3.$$

Rubrics: 2 points for the objective and 1 point for the constraints.

(b) Solve the dual formulation (show your derivation). (3 points)

Substituting α_3 with $\alpha_1 + \alpha_2$, the objective becomes

$$2\alpha_1 + 2\alpha_2 - \tfrac{1}{2}\left(4\alpha_1^2 + 4\alpha_2^2 + (\alpha_1 + \alpha_2)^2 - 2(\alpha_1 + \alpha_2)^2\right)$$
$$= 2\alpha_1 + 2\alpha_2 + \alpha_1\alpha_2 - \tfrac{3}{2}\alpha_1^2 - \tfrac{3}{2}\alpha_2^2.$$

Setting the gradient to zero gives $2 + \alpha_2 - 3\alpha_1 = 0$ and $2 + \alpha_1 - 3\alpha_2 = 0$. Solving this linear system gives $\alpha_1^* = \alpha_2^* = 1$, and thus $\alpha_3^* = \alpha_1^* + \alpha_2^* = 2$.

Rubrics: Give partial credits as appropriate. Do not deduct points if the mistake is solely due to the wrong formulation from the last question.

(c) Given a test point $x = 2$, what is the prediction of this SVM? Show your derivation, and note that to get full credits, you have to avoid directly operating in the feature space that corresponds to the kernel function. (6 points)

Based on Slide 39/43 of Lecture 6, we first figure out the primal solution b^*:

$$b^* = y_1 - \sum_{n=1}^{3} \alpha_n^* y_n k(x_n, x_1) = -1 + 4 + 0 - 2 = 1.$$

Then, the prediction can be found by $\text{SGN}\left(\sum_{n=1}^{3} \alpha_n^* y_n k(x_n, 2) + b^*\right) = \text{SGN}\left(-1 - 9 + 2 + 1\right) = -1$.

Rubrics:

- 2 points for finding the correct value of b^*, which can also be computed similarly using (x_2, y_2) or (x_3, y_3) since they are all support vectors.
- 2 points for deriving the correct final prediction.
- 2 points for doing everything without operating in the primal feature space.
- Similarly, do not deduct points if the mistake is solely due to the wrong dual solution from the last question.

3 Naive Bayes and Boosting (14 points)

Consider the dataset of 5 examples shown in Table 1. Each example has two features "Height" and "Vocabulary", and one label "Age". They all have two possible values: short or tall for Height, small or large for Vocabulary, and young or old for Age.

n	Height	Vocabulary	Age
1	short	large	old
2	tall	small	old
3	short	large	young
4	short	small	young
5	tall	large	young

Table 1: Training set

$p(\text{A} = \text{young})$	3/5
$p(\text{A} = \text{old})$	2/5

	H = short	H = tall	V = small	V = large
A = young	2/3	1/3	1/3	2/3
A = old	1/2	1/2	1/2	1/2

Table 2: Naive Bayes parameters

(a) Suppose that we run Naive Bayes on this dataset. Write down all the 10 parameters learned by this algorithm in Table 2. Specifically, for the table on the top, fill in the unconditional probability of the label, and for the table on the bottom, fill in the probability of each feature conditioning on the label: for example, the first entry represents $p(\text{H} = \text{short}|\text{A} = \text{young})$ (H, V and A are shorthands for Height, Vocabulary, and Age respectively). No reasoning needed. (5 points)

Rubrics: 0.5 point for each parameter.

(b) For each of the 4 possible feature combinations $\boldsymbol{x}$, write down the joint probability $p(\boldsymbol{x}, A = \text{young/old})$ in Table 3 (still under the Naive Bayes assumption), as well as the prediction of the algorithm. No reasoning needed. (6 points)

$\boldsymbol{x}$	$p(\boldsymbol{x}, A = \text{young})$	$p(\boldsymbol{x}, A = \text{old})$	Prediction (young or old)
(short, small)	$\frac{3}{5} \cdot \frac{2}{3} \cdot \frac{1}{3} = \frac{2}{15}$	$\frac{2}{5} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{10}$	young
(short, large)	$\frac{3}{5} \cdot \frac{2}{3} \cdot \frac{2}{3} = \frac{4}{15}$	$\frac{2}{5} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{10}$	young
(tall, small)	$\frac{3}{5} \cdot \frac{1}{3} \cdot \frac{1}{3} = \frac{1}{15}$	$\frac{2}{5} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{10}$	old
(tall, large)	$\frac{3}{5} \cdot \frac{1}{3} \cdot \frac{2}{3} = \frac{2}{15}$	$\frac{2}{5} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{10}$	young

Table 3: Naive Bayes predictions

Rubrics: 0.5 point for each entry. A single number is enough for the first two columns. For each row of these columns, if conditional probabilities $p(A = \text{young/old} \mid \boldsymbol{x})$ are given instead, give 0.5 point.

(c) Suppose that we run AdaBoost with Naive Bayes as the base algorithm. What you have figured out in the last two questions shows you what this base algorithm will do in the first round of AdaBoost. Calculate the importance of this classifier β_1 and the distribution D_2 over the 5 training examples for the next round of AdaBoost (see Slide 30/47 of Lecture 7 for the meaning of these notations). Show your derivation. (3 points)

Comparing the labels in Table 1 and the predictions in Table 3, we see that Naive Bayes only misclassifies the first training point and thus has training error $\epsilon_1 = 0.2$. Therefore $\beta_1 = \frac{1}{2} \ln \left(\frac{1-\epsilon_1}{\epsilon_1} \right) = \frac{1}{2} \ln 4 = \ln 2$, and the next distribution is such that $D_2(1) \propto e^{\ln 2} = 2$ and $D_2(2) = D_2(3) = D_2(4) = D_2(5) \propto e^{-\ln 2} = \frac{1}{2}$, which after normalization becomes $D_2 = (\frac{1}{2}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8})$.

Rubrics: 1 point for β_1 and 2 points for D_2 (okay to skip the normalization step).

4 HMM (16 points)

Recall that a Hidden Markov Model is parameterized by:

- initial state distribution $P(Z_1 = s) = \pi_s$,
- transition distribution $P(Z_{t+1} = s' \mid Z_t = s) = a_{s,s'}$,
- emission distribution $P(X_t = o \mid Z_t = s) = b_{s,o}$.

(a) Define $\zeta_{s,s'}(\tau) = P(Z_{t+\tau} = s' \mid Z_t = s)$, the probability of transiting from state s to state s' after τ steps (note that this quantity is independent of the starting time step t). Derive a recursive form to compute $\zeta_{s,s'}(\tau)$. More specifically, for any $\tau \geq 1$, express $\zeta_{s,s'}(\tau)$ in terms of $\zeta_{\cdot,\cdot}(\tau - 1)$ and the model parameters (show your derivation). (3 points)

$$
\begin{aligned}
\zeta_{s,s'}(\tau) &= P(Z_{t+\tau} = s' \mid Z_t = s) && \text{(definition)} \\
&= \sum_{s''} P(Z_{t+\tau} = s', Z_{t+\tau-1} = s'' \mid Z_t = s) && \text{(marginalizing, 1 point)} \\
&= \sum_{s''} P(Z_{t+\tau-1} = s'' \mid Z_t = s) P(Z_{t+\tau} = s' \mid Z_{t+\tau-1} = s'', Z_t = s) \\
&= \sum_{s''} P(Z_{t+\tau-1} = s'' \mid Z_t = s) P(Z_{t+\tau} = s' \mid Z_{t+\tau-1} = s'') && \text{(Markov property, 1 point)} \\
&= \sum_{s''} \zeta_{s,s''}(\tau - 1) a_{s'',s'} && \text{(definition, 1 point)}
\end{aligned}
$$

(b) Based on the recursive form you derived from the last question, write down an algorithm that calculates $\zeta_{s,s'}(\tau)$ for all state pairs $(s, s') \in [S] \times [S]$ and all $\tau \in \{0, 1, \ldots, T\}$. (4 points)

Algorithm 1: Algorithm for computing ζ

Initialization: set $\zeta_{s,s'}(0) = \begin{cases} 1 & \text{if } s' = s \\ 0 & \text{else} \end{cases}$ for all $(s, s') \in [S] \times [S]$

for $\tau = 1, \ldots, T$ **do**
  for *all* $(s, s') \in [S] \times [S]$ **do**
    compute

$$\zeta_{s,s'}(\tau) = \sum_{s'' \in [S]} \zeta_{s,s''}(\tau - 1) a_{s'',s'}$$

Rubrics: 2 points for the initialization and another 2 points for the rest. Deduct 1 point if the case $\tau = 0$ is missing.

(c) Suppose that we collect only a subsequence of M observations out of T steps: $x_{t_1}, x_{t_2}, \ldots, x_{t_M}$, where $1 \le t_1 < t_2 < \cdots < t_M \le T$. Write down a variant of the Viterbi algorithm that outputs the corresponding most likely hidden state subsequence $z^*_{t_1}, \ldots, z^*_{t_M}$. No reasoning is needed, and feel free to directly use quantities computed from the last question (Algorithm 1) if needed. (9 points)

Algorithm 2: Viterbi algorithm for a subsequence

Input: observations $x_{t_1}, x_{t_2}, \ldots, x_{t_M}$, where $1 \le t_1 < t_2 < \cdots < t_M \le T$

Output: the most likely state subsequence $z^*_{t_1}, \ldots, z^*_{t_M}$.

Initialize: for each $s \in [S]$, compute $\delta_s(1) = b_{s,x_{t_1}} \sum_{s'} \pi_{s'} \zeta_{s',s}(t_1 - 1)$

for $m = 2, \ldots, M$ **do**

  for *each* $s \in [S]$ **do**

    Compute

$$\delta_s(m) = b_{s,x_{t_m}} \max_{s'} \zeta_{s',s}(t_m - t_{m-1}) \delta_{s'}(m-1)$$

$$\Delta_s(m) = \operatorname*{argmax}_{s'} \zeta_{s',s}(t_m - t_{m-1}) \delta_{s'}(m-1).$$

Backtracking: let $z^*_{t_M} = \operatorname{argmax}_s \delta_s(M)$.

For $m = M, \ldots, 2$, set $z^*_{t_{m-1}} = \Delta_{z^*_{t_m}}(m)$.

Reasoning (NOT required): this can be thought of as another Markov chain of length M where the transition between the $(m-1)$-th and m-th step is governed by $\zeta_{s,s'}(t_m - t_{m-1})$. With this view, the algorithm is essentially the original Viterbi algorithm applied to this modified Markov chain, with $a_{s,s'}$ replaced by $\zeta_{s,s'}(t_m - t_{m-1})$ for the appropriate m. The only additional detail is to figure out the modified initial distribution, which is $\sum_{s'} \pi_{s'} \zeta_{s',s}(t_1 - 1)$ for state s.

Rubrics: 3 points each for the initialization, the procedure of computing δ and Δ, and the backtracking. Give partial credits as appropriate.

5 Clustering and EM (28 points)

Suppose that we are given a set of black-and-white 32×32 images, each showing a single hand-written digit. In this problem, you need to apply two different methods to cluster this dataset (note that these two methods are independent so you can solve one without having solved the other).

(a) **(The Basic)** We start by formalizing the problem with the notations we learned from the lectures. Since each pixel is either black or white, we can represent an image as a vector $\boldsymbol{x} \in \{0,1\}^D$ (0 stands for white and 1 stands for black). Further let N be the total number of images and K be the number of clusters we want. Write down the appropriate value of D and K for this application based on the information above (no reasoning needed). (2 points)

$D = 32 \times 32 = 1024$, $K = 10$ (as there are 10 possible digits)

Rubrics: 1 point for each.

(b) **(K-means)** Next, we considering applying K-means++ to cluster this dataset. Instead of doing this directly to the original dataset, we apply a feature map $\boldsymbol{\phi} : \{0,1\}^D \to \mathbb{R}^M$ to each image, and then apply K-means++ to the new features. With a corresponding kernel function $k(\cdot,\cdot)$ for this feature map, you need to follow the steps below to show that this algorithm can be efficiently implemented (that is, without operating in space $\mathbb{R}^M$).

(1) First, recall that in the initialization step of K-means++, we need to randomly select K centers based on squared L2 distances, which requires calculating $\|\boldsymbol{\phi}(\boldsymbol{x}_n) - \boldsymbol{\phi}(\boldsymbol{x}_m)\|_2^2$ for some $n, m \in [N]$. Express this quantity using the kernel function only (show your derivation). (2 points)

$$\begin{aligned} \|\boldsymbol{\phi}(\boldsymbol{x}_n) - \boldsymbol{\phi}(\boldsymbol{x}_m)\|_2^2 &= \boldsymbol{\phi}(\boldsymbol{x}_n)^\top \boldsymbol{\phi}(\boldsymbol{x}_n) - 2\boldsymbol{\phi}(\boldsymbol{x}_n)^\top \boldsymbol{\phi}(\boldsymbol{x}_m) + \boldsymbol{\phi}(\boldsymbol{x}_m)^\top \boldsymbol{\phi}(\boldsymbol{x}_m) && \text{(1 point)} \\ &= k(\boldsymbol{x}_n, \boldsymbol{x}_n) - 2k(\boldsymbol{x}_n, \boldsymbol{x}_m) + k(\boldsymbol{x}_m, \boldsymbol{x}_m) && \text{(1 point)} \end{aligned}$$

(2) Suppose that $\mathcal{S} \subset [N]$ contains a nonempty subset of images belonging to the same cluster in some iteration of K-means++. In the next iteration, one needs to compute the squared distance between an arbitrary image $\boldsymbol{\phi}(\boldsymbol{x}_n)$ and the center of this cluster $\boldsymbol{\mu} = \frac{1}{|\mathcal{S}|} \sum_{m \in \mathcal{S}} \boldsymbol{\phi}(\boldsymbol{x}_m)$, that is, $\|\boldsymbol{\phi}(\boldsymbol{x}_n) - \boldsymbol{\mu}\|_2^2$. Once again, express this quantity using the kernel function only (show your derivation). (4 points)

$$\begin{aligned} \|\boldsymbol{\phi}(\boldsymbol{x}_n) - \boldsymbol{\mu}\|_2^2 &= \boldsymbol{\phi}(\boldsymbol{x}_n)^\top \boldsymbol{\phi}(\boldsymbol{x}_n) - 2\boldsymbol{\phi}(\boldsymbol{x}_n)^\top \boldsymbol{\mu} + \boldsymbol{\mu}^\top \boldsymbol{\mu} && \text{(1 point)} \\ &= k(\boldsymbol{x}_n, \boldsymbol{x}_n) - \frac{2}{|\mathcal{S}|} \sum_{m \in \mathcal{S}} \boldsymbol{\phi}(\boldsymbol{x}_n)^\top \boldsymbol{\phi}(\boldsymbol{x}_m) + \frac{1}{|\mathcal{S}|^2} \sum_{m,m' \in \mathcal{S}} \boldsymbol{\phi}(\boldsymbol{x}_m)^\top \boldsymbol{\phi}(\boldsymbol{x}_{m'}) && \text{(2 points)} \\ &= k(\boldsymbol{x}_n, \boldsymbol{x}_n) - \frac{2}{|\mathcal{S}|} \sum_{m \in \mathcal{S}} k(\boldsymbol{x}_n, \boldsymbol{x}_m) + \frac{1}{|\mathcal{S}|^2} \sum_{m,m' \in \mathcal{S}} k(\boldsymbol{x}_m, \boldsymbol{x}_{m'}) && \text{(1 point)} \end{aligned}$$

(3) Based on your answers from the last two questions, fill in the missing details in Algorithm 3. More specifically,

- complete the for loop in Line 2 which finds the initial center indices $n_2, \dots, n_K \in [N]$;
- complete the for loop in Line 5 which finds the new partition $\mathcal{S}'_1, \dots, \mathcal{S}'_K$ based on $\mathcal{S}_1, \dots, \mathcal{S}_K$.

Note that we have pre-computed the Gram matrix $\boldsymbol{M}$ as an input, so you can directly use $M_{n,m}$ whenever you need $k(\boldsymbol{x}_n, \boldsymbol{x}_m)$. Also note that the only required output is the partition $\mathcal{S}_1, \dots, \mathcal{S}_K \subset [N]$, but not their corresponding centers since they live in $\mathbb{R}^M$ (make sure that your algorithm does not use any vectors in $\mathbb{R}^M$). (6 points)

Algorithm 3: K-means++ with kernel

Input: dataset $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$ with Gram matrix $\boldsymbol{M}$
Output: a partition of the dataset $\mathcal{S}_1, \dots, \mathcal{S}_K \subset [N]$
Initialize:
1 Uniformly at random select an index $n_1 \in [N]$ as the first center.
2 **for** $k = 2, \dots, K$ **do**
  Randomly select the k-th center's index n_k such that

$$\mathbb{P}[n_k = n] \propto \min_{j=1,\dots,k-1} \left(M_{n,n} - 2M_{n,n_j} + M_{n_j,n_j}\right)$$

3 Set $\mathcal{S}_k = \{n_k\}$ for all $k \in [K]$
Repeat until convergence:
4 Set $\mathcal{S}'_k = \emptyset$ for all $k \in [K]$ ▷ initialize the partition as empty sets
5 **for** $n = 1, \dots, N$ **do**
    $k = \operatorname{argmin}_{j \in [K]} \left(M_{n,n} - \frac{2}{|\mathcal{S}_j|} \sum_{m \in \mathcal{S}_j} M_{n,m} + \frac{1}{|\mathcal{S}_j|^2} \sum_{m,m' \in \mathcal{S}_j} M_{m,m'}\right)$
    $\mathcal{S}'_k \leftarrow \mathcal{S}'_k \cup \{n\}$
6 Set $\mathcal{S}_k = \mathcal{S}'_k$ for all $k \in [K]$ ▷ overwrite $\mathcal{S}_1, \dots, \mathcal{S}_K$ with the new partition.

Rubrics: 2 points for the first loop and 4 points for the second loop. Note that dropping $M_{n,n}$ in both places results in the same algorithm. Do not deduct points for mistakes inherited from the last two questions.

(c) **(Mixture Model and EM)** In the lectures we discussed the Gaussian mixture model for clustering, which is unfortunately not applicable here since we have binary features. Instead, we can replace each Gaussian component with a Bernoulli component. More specifically, an image $\boldsymbol{x}_n$ is assumed to be generated in the following manner:

- just like GMM, we first randomly select a hidden variable $z_n \in [K]$ such that $p(z_n = k) = \omega_k$, where $\omega_1, \dots, \omega_K$ are K mixture weights;
- supposing $z_n = k$, for each coordinate $d \in [D]$ of $\boldsymbol{x}_n$, we then independently sample $x_{nd} \in \{0, 1\}$ from a Bernoulli distribution $\text{Ber}(\mu_{kd})$ for some mean parameter $\mu_{kd} \in [0, 1]$.

In other words, the marginal distribution of $\boldsymbol{x}_n$ is

$$p(\boldsymbol{x}_n) = \sum_{k=1}^{K} p(z_n = k) p(\boldsymbol{x}_n | z_n = k) = \sum_{k=1}^{K} \omega_k \prod_{d=1}^{D} (1 - \mu_{kd})^{1 - x_{nd}} \mu_{kd}^{x_{nd}}.$$

Now follow the steps below to derive the EM algorithm for this model.

(1) **(E-step)** Fixing the model parameters $\{\omega_k\}_{k\in[K]}$ and $\{\mu_{kd}\}_{k\in[K],d\in[D]}$, derive the posterior distribution of the hidden variable: $\gamma_{nk} = p(z_n = k|\boldsymbol{x}_n)$ for each n and k. Using the proportional sign is acceptable. (4 points)

$$\begin{aligned}
\gamma_{nk} &= p(z_n = k|\boldsymbol{x}_n) \\
&\propto p(z_n = k, \boldsymbol{x}_n) && \text{(1 point)} \\
&= p(z_n = k)p(\boldsymbol{x}_n|z_n = k) && \text{(1 point)} \\
&= \omega_k \prod_{d=1}^{D} (1-\mu_{kd})^{1-x_{nd}} \mu_{kd}^{x_{nd}}. && \text{(2 points)}
\end{aligned}$$

(2) **(E-step)** Fixing γ_{nk}, write down the expected complete log-likelihood function $Q(\boldsymbol{\omega}, \boldsymbol{\mu})$, where $\boldsymbol{\omega}$ and $\boldsymbol{\mu}$ denote the collections of model parameters $\{\omega_k\}_{k\in[K]}$ and $\{\mu_{kd}\}_{k\in[K],d\in[D]}$ respectively. Show your derivation. (4 points)

$$\begin{aligned}
Q(\boldsymbol{\omega}, \boldsymbol{\mu}) &= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk} \ln p(z_n = k, \boldsymbol{x}_n) && \text{(2 points)} \\
&= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk} \left(\ln \omega_k + \sum_{d=1}^{D} (1-x_{nd}) \ln(1-\mu_{kd}) + \sum_{d=1}^{D} x_{nd} \ln \mu_{kd} \right) && \text{(2 points)} \\
&= \sum_{k=1}^{K} \left(\sum_{n=1}^{N} \gamma_{nk} \right) \ln \omega_k + \sum_{k=1}^{K}\sum_{d=1}^{D}\sum_{n=1}^{N} \gamma_{nk} \left((1-x_{nd}) \ln(1-\mu_{kd}) + x_{nd} \ln \mu_{kd} \right)
\end{aligned}$$

Rubrics: The last step is not necessary (but the rearranging helps solve the next question). Other equivalent expressions are acceptable as well.

(3) **(M-step)** Fixing γ_{nk} again, find the maximizer of $Q(\boldsymbol{\omega}, \boldsymbol{\mu})$. Show your derivation, but feel free to use conclusions from the lectures directly if needed. (6 points)

Optimizing over $\boldsymbol{\omega}$ only requires considering the first term $\sum_{k=1}^{K} \left(\sum_{n=1}^{N} \gamma_{nk} \right) \ln \omega_k$, which based on HW4 Q2, is maximized when $\omega_k = \frac{1}{N} \sum_{n=1}^{N} \gamma_{nk}$.

For each μ_{kd}, the only relevant term is $\sum_{n=1}^{N} \gamma_{nk} \left((1-x_{nd}) \ln(1-\mu_{kd}) + x_{nd} \ln \mu_{kd} \right)$, whose derivative with respect to μ_{kd} is

$$\sum_{n=1}^{N} \gamma_{nk} \left(\frac{x_{nd}-1}{1-\mu_{kd}} + \frac{x_{nd}}{\mu_{kd}} \right) = \sum_{n=1}^{N} \frac{\gamma_{nk}(x_{nd}-\mu_{kd})}{(1-\mu_{kd})\mu_{kd}} = \frac{\sum_{n=1}^{N} \gamma_{nk} x_{nd} - \mu_{kd} \sum_{n=1}^{N} \gamma_{nk}}{(1-\mu_{kd})\mu_{kd}}.$$

Setting the above to zero and solving for μ_{kd} gives the maximizer

$$\mu_{kd} = \frac{\sum_{n=1}^{N} \gamma_{nk} x_{nd}}{\sum_{n=1}^{N} \gamma_{nk}}.$$

Rubrics: 2 points for finding the correct answer for $\boldsymbol{\omega}$ and 4 points for $\boldsymbol{\mu}$ (e.g. 2 points for the correct derivative/gradient and 2 points for the correct final answer).