Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

Instructions: *Total:* **100 marks**

1. Total duration: **3 hours**. Please write your name, roll number, department on **all pages**.
2. This booklet has 10 pages (8 pages + 2 pages for rough work). No part of your answers should be on pages designated for rough work. Additional rough sheets may be provided if needed.
3. Write/mark your answers clearly in the provided space. Please keep your answers precise and concise.
4. Avoid showing very detailed derivations. You may do those on rought sheet and only show key steps.
5. **Answer each question using information provided in the question (no clarifications during the exam). If you want to make any assumptions, please state them in your answer.**

Section 1 (9 Descriptive Answer Questions: Total 100 marks). .

1. Consider a classification problem with K classes. Assume that for each class $c = 1, 2, \ldots, K$, we are given a class-attribute vector $\boldsymbol{a}_c \in \mathbb{R}^M$. We are giving training data $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$ with inputs $\boldsymbol{x}_n \in \mathbb{R}^D$, but the training examples are only from the first S classes and the remaining $U = K - S$ classes do not have any training examples. The test input $\boldsymbol{x}_*$ can be from any of the K classes. Explain (with all necessary equations) how you would learn a learning with prototypes (LwP) classifier for this problem. Your solution must not use anything other than simple vector operations like addition or dot products. (**12 marks**).

   For the first S seen classes, we can compute their means easily as $\mu_c = \frac{1}{N_c} \sum_{\boldsymbol{x}_n : y_n = c} \boldsymbol{x}_n$, $c = 1, 2, \ldots, S$.

   Given these S means, we can express the mean μ_c of each of the remaining classes $c = S+1, \ldots, K$, for which we do not have any training examples, as a weighted sum of the means of the first S classes.

   In equations, we can write it as $\mu_c = \sum_{i=1}^S s_{c,i} \mu_i$, $(c = S+1, \ldots, K)$ where $s_{c,i}$ is the similarity of class c with class i. We can define it in many ways. One simple way is to use the dot product of their class attribute vectors, i.e., $s_{c,i} = \boldsymbol{a}_c^\top \boldsymbol{a}_i$.

   Note that this simple method was already described in homework 1 programming problem.

   Another approach: We can also use a regression method to do this (also described in homework 1 programming problem). Basically, we can assume that each μ_c is defined as $\mu_c = \mathbf{W}\boldsymbol{a}_c$. Using the $(\mu_c, \boldsymbol{a}_c), c = 1, 2, \ldots, S$ as training data, we can estimate $\mathbf{W}$ by minimizing a squared loss function, i.e., $\hat{\mathbf{W}} = \text{argmin}_{\mathbf{W}} \sum_{c=1}^S ||\mu_c - \mathbf{W}\boldsymbol{a}_c||^2$. Once we have estimated $\hat{\mathbf{W}}$, we can get the mean μ_c for the remaning classes $c = S+1, \ldots, K$ simply as $\mu_c = \hat{\mathbf{W}}\boldsymbol{a}_c$ This method however will involve additional operations such as matrix inverse etc to solve for $\mathbf{W}$.

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

2. Consider the ridge regression problem:

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \frac{1}{2}\sum_{n=1}^{N}(y_n - \boldsymbol{w}^\top \boldsymbol{x}_n)^2 + \frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \frac{1}{2}(\boldsymbol{y} - \mathbf{X}\boldsymbol{w})^\top(\boldsymbol{y} - \mathbf{X}\boldsymbol{w}) + \frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w}$$

where $\mathbf{X}$ is the $N \times D$ feature matrix and $\boldsymbol{y}$ is the $N \times 1$ vector of labels of the N training examples. Note that the factor of $\frac{1}{2}$ has been used in the above expression just for convenience of derivations required for this problem and does not change the solution to the problem.

Derive the Newton's method's update equations for each iteration. For this model, how many iterations would the Newton's method will take to converge? (**14 marks**)

Newton's method requires the gradient and the Hessian matrix of the loss.

The gradient is $\boldsymbol{g} = -\mathbf{X}^\top(\boldsymbol{y} - \mathbf{X}\boldsymbol{w}) + \lambda\boldsymbol{w}$.

The Hessian matrix is the gradient of the gradient is equal to $\mathbf{H} = \mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}$

The Newton's method update for iteration t is $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \mathbf{H}_t^{-1}\boldsymbol{g}_t$ where $\boldsymbol{g}_t$ and $\mathbf{H}_t$ are the gradient and Hessian, respectively, evaluated at $\boldsymbol{w} = \boldsymbol{w}_t$. Note that the Hessian for this model doesn't depend on $\boldsymbol{w}$.

The Newton's update will be

$$\begin{aligned}
\boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \mathbf{H}_t^{-1}\boldsymbol{g}_t \\
&= \boldsymbol{w}_t + (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}[\mathbf{X}^\top(\boldsymbol{y} - \mathbf{X}\boldsymbol{w}_t) - \lambda\boldsymbol{w}_t] \\
&= \boldsymbol{w}_t - (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}(\mathbf{X}^\top\mathbf{X}\boldsymbol{w}_t + \lambda\boldsymbol{w}_t) + (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\boldsymbol{y} \\
&= \boldsymbol{w}_t - (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})\boldsymbol{w}_t + (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\boldsymbol{y} \\
&= (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\boldsymbol{y}
\end{aligned}$$

Since the update does not depend on the iteration number (it is always equal to $(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\boldsymbol{y}$), Newton's method will converge in just ONE iteration.

Note: It happened so because of the simple nature of the loss function (it is a strictly convex function and for such function, Newton's method gives the globally optimal solution, and for the particular case of ridge regression loss, it takes just one iteration to converge)

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

3. Consider K-means clustering where we are trying to learn K means $\mu_1, \ldots, \mu_K$, given N observations $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, with each $\boldsymbol{x}_n \in \mathbb{R}^D$. Suppose we have some *a priori* information that the K means are "close" to known vectors $\mu_1^*, \ldots, \mu_K^*$, respectively. Propose a suitable prior distribution for each mean μ_k that makes use of this information. Now derive the K-means algorithm updates for cluster assignments $\boldsymbol{z}_n$ and cluster means μ_k when we use this prior distribution as a regularizer. (**12 marks**)

We can choose a prior distribution such that μ_k^* is the mean of the prior distribution of μ_k. One such prior is a Gaussian distribution with mean equal to μ_k^*, i.e., $p(\mu_k) = \mathcal{N}(\mu_k | \mu_k^*, \Sigma)$. You can set Σ as some simple covariance matrix like identity or spherical ($\mathbf{I}$ or $\lambda^{-1}\mathbf{I}$). We will assume it to be the identity matrix.

The K-means loss function in this case will be the standard loss function of K-means, plus a regularizer which will be due to the use of the above prior distribution.

As we know, the regularizer corresponds to the (negative) log of the prior, which in this case will be

$$R(\mu_k) = -\log \mathcal{N}(\mu_k | \mu_k^*, \lambda^{-1}\mathbf{I}) = \lambda(\mu_k - \mu_k^*)^\top(\mu_k - \mu_k^*) = \lambda||\mu_k - \mu_k^*||^2 \quad \text{(ignoring constants)}$$

The K-means objective will thus be

$$\mathcal{L}(\mu_1, \mu_2, \ldots, \mu_K, \mathbf{Z}) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}||\boldsymbol{x}_n - \mu_k||^2 + \lambda \sum_{k=1}^{K} ||\mu_k - \mu_k^*||^2$$

We will use ALT-OPT to minimize the above objective. As far as $\mathbf{Z}$ is concerned, the objective is exactly the same as standard K-means objective so, as per ALT-OPT, $\hat{z}_n = \text{argmin}_k ||\boldsymbol{x}_n - \mu_k||^2$, i.e., assigning $\boldsymbol{x}_n$ to the cluster whose mean is the closest to $\boldsymbol{x}_n$. Given $\mathbf{Z}$ (i.e., knowing which data point is currently assigned to which cluster), finding the optimal μ_k boils down to solving the following problem:

$$\hat{\mu}_k = \underset{\mu_k}{\text{argmin}} \sum_{n:z_n=k} ||\boldsymbol{x}_n - \mu_k||^2 + \lambda||\mu_k - \mu_k^*||^2 = \frac{\sum_{n:z_n=k} \boldsymbol{x}_n + \lambda\mu_k^*}{N_k + \lambda}$$

Note the interesting role playing by μ_k^*. It is like having λ copies of an additional "pseudo" observation μ_k^* assigned to cluster k, so these pseudo observations also contribute to the mean of cluster k.

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

---

4. Suppose we have collected N observations $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ using a sensor. Let us assume each $\boldsymbol{x} \in \mathbb{R}^D$ as generated from a Gaussian distribution $\mathcal{N}(\mu, \Sigma)$. We would like to estimate the mean and covariance of this Gaussian. However, suppose the sensor was faulty and each $\boldsymbol{x}_n$ could only have part it as observed (think of a blacked out image). Denote $\boldsymbol{x}_n = [\boldsymbol{x}_n^{obs}, \boldsymbol{x}_n^{miss}]$ where $\boldsymbol{x}_n^{obs}$ and $\boldsymbol{x}_n^{miss}$ denote the observed and missing parts, respectively, of $\boldsymbol{x}_n$. We only get to see $\boldsymbol{x}_n^{obs}$. Note that different observations could have different parts as missing (e.g., different images may have different sets of pixels as missing), so the indices of the observed/missing entries of the vector $\boldsymbol{x}_n$ may be different for different n.

   We can use EM to get maximum likelihood estimates of μ and Σ given this partially observed data. To do so, you will treat each $\boldsymbol{x}_n^{miss}$ as a latent variable and estimate its conditional posterior $p(\boldsymbol{x}_n^{miss}|\boldsymbol{x}_n^{obs}, \mu, \Sigma)$, given the current estimates μ and Σ of the parameters. In the M step, you will re-estimate μ and Σ. Clearly write down the following: (1) The expression for $p(\boldsymbol{x}_n^{miss}|\boldsymbol{x}_n^{obs}, \mu, \Sigma)$; (2) The expected CLL for this model; (3) The M step update equations for μ and Σ. (**14 marks**)

   (1) It is given that $\boldsymbol{x}_n = [\boldsymbol{x}_n^{obs}, \boldsymbol{x}_n^{miss}]$ is drawn from a Gaussian $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. For each $\boldsymbol{x}_n$, let us assume that the mean and covariance matrix of this Gaussian are partitioned as $\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_n^o \\ \boldsymbol{\mu}_n^m \end{pmatrix}$ and covariance matrix $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_n^{oo} & \boldsymbol{\Sigma}_n^{om} \\ \boldsymbol{\Sigma}_n^{mo} & \boldsymbol{\Sigma}_n^{mm} \end{bmatrix}$, where 'o' and 'm' denotes the set of indices for the observed and missing entries.

   Using the Gaussian conditional formula, the CP is given by $p(\boldsymbol{x}_n^{miss}|\boldsymbol{x}_n^{obs}, \mu, \Sigma) = \mathcal{N}(\boldsymbol{x}_n^{miss}|\boldsymbol{\mu}_n^{m|o}, \boldsymbol{\Sigma}_n^{m|o})$, where $\boldsymbol{\Sigma}_n^{m|o} = \boldsymbol{\Sigma}_n^{mm} - \boldsymbol{\Sigma}_n^{mo}(\boldsymbol{\Sigma}_n^{oo})^{-1}\boldsymbol{\Sigma}_n^{om}$ and $\boldsymbol{\mu}_n^{m|o} = \boldsymbol{\mu}_n^m + \boldsymbol{\Sigma}_n^{mo}(\boldsymbol{\Sigma}_n^{oo})^{-1}(\boldsymbol{x}_n^{obs} - \boldsymbol{\mu}_n^o)$

   (2) The complete data log-likelihood (CLL) is $\ell_n = \log \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. It's "complete" since the missing part of $\boldsymbol{x}_n$ (latent variable) is also within $\boldsymbol{x}_n = [\boldsymbol{x}_n^{obs}, \boldsymbol{x}_n^{miss}]$. Substituting and simplifying, we get $\ell_n = \log \frac{1}{\sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}} \exp\{-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu})\}$. Simplifying further and ignoring the constants, we get $\ell_n = -\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}) - \frac{1}{2}\log|\boldsymbol{\Sigma}|$. The overall expected CLL will be $\sum_{n=1}^N \mathbb{E}[\ell_n]$. To get this expectation, linear terms in the CLL, like $\boldsymbol{x}_n^\top \boldsymbol{\mu}$ will need to be replaced by $\mathbb{E}[\boldsymbol{x}_n]^\top \boldsymbol{\mu}$ and quadratic terms like $\boldsymbol{x}_n^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{x}_n$ will need to be replaced by $\mathbb{E}[\boldsymbol{x}_n^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{x}_n] = \text{trace}[\boldsymbol{\Sigma}^{-1}\mathbb{E}[\boldsymbol{x}_n \boldsymbol{x}_n^\top]]$ using cyclic property of trace.

   (3) The MLE solution can be obtained by optimizing the expected CLL w.r.t. $\boldsymbol{\mu}, \boldsymbol{\Sigma}$. However, we can also get it by easily. Note that if there were no missing values in any $\boldsymbol{x}_n$, the MLE solution for mean and covariance matrix are $\hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{n=1}^N \boldsymbol{x}_n$ and $\hat{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{n=1}^N (\boldsymbol{x}_n - \hat{\boldsymbol{\mu}})(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}})^\top$, respectively. However, there is a missing part in $\boldsymbol{x}_n$ so we need to replace each $\boldsymbol{x}_n$ and $\boldsymbol{x}_n \boldsymbol{x}_n^\top$ term by the corresponding expectations. For example, in the expression of $\hat{\boldsymbol{\mu}}$, we replace $\boldsymbol{x}_n$ by $\mathbb{E}[\boldsymbol{x}_n] = \mathbb{E}\begin{bmatrix} \boldsymbol{x}_n^{obs} \\ \boldsymbol{x}_n^{miss} \end{bmatrix}$. Note that when computing expectations/covariance, you need to treat the observed part $\boldsymbol{x}_n^{obs}$ as a constant (not a random variable). E.g., its expectation will simply be $\mathbb{E}[\boldsymbol{x}_n^{obs}] = \boldsymbol{x}_n^{obs}$, whereas since $\boldsymbol{x}_n^{miss}$ is a random variable with distribution given by the CP, we will need to compute its expectation as $\mathbb{E}[\boldsymbol{x}_n^{miss}] = \boldsymbol{\mu}_n^{m|o}$ (mean of CP, and likewise its covariance as well be computed using the CP's covariance matrix).

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

5. Suppose we have data from two classes whose inputs are assumed to be generated from two Gaussian distributions $\mathcal{N}\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{bmatrix}2 & 0\\0 & 2\end{bmatrix}\right)$ and $\mathcal{N}\left(\begin{pmatrix}2\\2\end{pmatrix}, \begin{bmatrix}5 & 0\\0 & 5\end{bmatrix}\right)$, respectively. Suppose we have learned a logistic regression model on this data but the model isn't performing on the test set.

   - If we have infinite amount of training data, will logistic regression model achieve zero training error? Briefly justify your answer. (**3 marks**)

     Since the two Gaussians overlap significantly, the data will not be linearly separable.

   - If we also add a regularizer to the above logistic regression model (infinite training data), will it achieve zero training error? Briefly justify your answer. (**3 marks**)

     Even with the regularizer, the model remains a linear model and thus the training error can't become zero.

   - If we switch to a kernel SVM (but keep the training set size as finite), is it possible to achieve zero training error? Briefly justify your answer. (**3 marks**)

     Yes because we can now learn nonlinear boundaries and with kernel SVM with an appropriate kernel like RBF, any nonlinear function can be learned in principle.

   - If we switch to a deep neural network (but keep the training set size as finite), is it possible to achieve zero training error? Briefly justify your answer. (**3 marks**)

     Yes because deep neural networks can also learn any nonlinear function in principle.

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

---

6. Given query, key, and value matrices $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, respectively (and all being of size $N \times d$), one way to define the $N \times d$ output of the self-attention mechanism is $\text{ATT}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\mathbf{Q}\mathbf{K}^\top/\sqrt{d}\right)\mathbf{V}$, where the softmax function is assumed to be applied row-wise on the matrix $\mathbf{Q}\mathbf{K}^\top/\sqrt{d}$.

   Note that we can also write the same as $\text{ATT}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{D}^{-1}\mathbf{A}\mathbf{V}$ where $\mathbf{A} = \exp(\mathbf{Q}\mathbf{K}^\top/\sqrt{d})$, and $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1}_N)$ and $\mathbf{1}_N$ denotes a column vector of all 1s. Denoting $\mathbf{A}$ as $\text{SM}(\mathbf{Q}, \mathbf{K})$, we can define a "softmax" kernel function $\text{SM}(\boldsymbol{q}_i, \mathbf{k}_j) = \exp(\boldsymbol{q}_i^\top \mathbf{k}_j/\sqrt{d})$, such that the $(i, j)^{th}$ entry of $\mathbf{A}$ equals $\text{SM}(\boldsymbol{q}_i, \mathbf{k}_j)$.

   - Show that the above softmax" kernel function $\text{SM}(\boldsymbol{q}_i, \mathbf{k}_j)$ can be written as a scalar multiplied with well-known kernel function. You must show this with precise mathematical expressions. (**6 marks**)
   - Let's define $\text{NEW\_ATT}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \tilde{\mathbf{D}}^{-1}\tilde{\mathbf{A}}\mathbf{V}$ where $\tilde{\mathbf{A}} = \text{tril}(\mathbf{A})$, $\tilde{\mathbf{D}} = \text{diag}(\tilde{\mathbf{A}}\mathbf{1}_N)$, and tril(.) extracts the lower-triangular part of the matrix including its diagonal. Briefly explain the difference between the outputs of the two attention functions $\text{ATT}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ and $\text{NEW\_ATT}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$. (**6 marks**)

   - $\text{SM}(\boldsymbol{q}_i, \mathbf{k}_j) = \exp(\boldsymbol{q}_i^\top \mathbf{k}_j/\sqrt{d})$. Atfer some rearranging of the terms, we can write as

$$\begin{aligned}\text{SM}(\boldsymbol{q}_i, \mathbf{k}_j) &= \exp(2\boldsymbol{q}_i^\top \mathbf{k}_j/2\sqrt{d}) = \exp(-(||\boldsymbol{q}_i - \mathbf{k}_j||^2 + ||\boldsymbol{q}_i||^2 + ||\mathbf{k}_j||^2)/2\sqrt{d}) \\ &= \exp(||\boldsymbol{q}_i||^2/2\sqrt{d})\exp(||\mathbf{k}_j||^2/2\sqrt{d})\exp(-(||\boldsymbol{q}_i - \mathbf{k}_j||^2\sqrt{d})\end{aligned}$$

     The part in red above is a scalar and the part in green in an RBF kernel with inputs $\boldsymbol{q}_i$ and $\mathbf{k}_j$.
   - $\mathbf{H} = \text{ATT}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{D}^{-1}\mathbf{A}\mathbf{V}$ where $\mathbf{A} = \exp(\mathbf{Q}\mathbf{K}^\top/\sqrt{d})$ computes each output $\boldsymbol{h}_n$ (row N of matrix $\mathbf{H}$) as a weighted combination of the value vectors of all the keys $\mathbf{k}_1, \ldots, \mathbf{k}_N$.
     On the other hand, consider $\mathbf{H} = \text{NEW\_ATT}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \tilde{\mathbf{D}}^{-1}\tilde{\mathbf{A}}\mathbf{V}$ where $\tilde{\mathbf{A}} = \text{tril}(\mathbf{A})$, $\tilde{\mathbf{D}} = \text{diag}(\tilde{\mathbf{A}}\mathbf{1}_N)$. Since the upper diagonal part of $\tilde{\mathbf{A}}$ contains all zeros, when computing each output $\boldsymbol{h}_n$, the query $\boldsymbol{q}_n$ will not pay any attention paid to keys $\mathbf{k}_i$ with $i > n$. Thus $\boldsymbol{h}_n$ will be computed using a weighted combination of the value vectors of only the keys $\mathbf{k}_1, \ldots, \mathbf{k}_n$. Note that this is similar to the "masked attention" mechanism in the decoder part of the transformer.

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

7. Consider modeling some data $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \{0, 1\}$, using a mixture of logistic regression models, where we model each binary label y_n by first picking one of the K logistic regression models, based on the value of a latent variable $z_n \sim$ multinoulli$(\pi_1, \ldots, \pi_K)$, and then generating y_n *conditioned* on z_n as $y_n \sim$ Bernoulli$[\sigma(\boldsymbol{w}_{z_n}^\top \boldsymbol{x}_n)]$. Now consider the *marginal* probability of the label $y_n = 1$, given $\boldsymbol{x}_n$, i.e., $p(y_n = 1|\boldsymbol{x}_n)$, and show that this can be quantity can also be thought of as the output of a neural network. Clearly specify what is the input layer, hidden layer(s), activations, the output layer, and the connection weights of this neural network. (**5 marks**)

The marginal probability $p(y_n = 1|\boldsymbol{x}_n)$ can be obtained by taking the joint probability $p(y_n = 1, \boldsymbol{z}_n|\boldsymbol{x}_n) = p(y_n = 1|\boldsymbol{z}_n, \boldsymbol{x}_n)p(\boldsymbol{z}_n)$ and summing it over all possible values of $\boldsymbol{z}_n$.

So $p(y_n = 1|\boldsymbol{x}_n) = \sum_{k=1}^K p(y_n = 1|\boldsymbol{z}_n = k, \boldsymbol{x}_n)p(\boldsymbol{z}_n = k) = \sum_{k=1}^K \pi_k \sigma(\boldsymbol{w}_k^\top \boldsymbol{x}_n)$

The above output can be thought of as being computing using a neural network where

- Input layer taken as input $\boldsymbol{x}_n$.
- There is a single hidden layer with K hidden units whose outputs are $\sigma(\boldsymbol{w}_1^\top \boldsymbol{x}_n), \sigma(\boldsymbol{w}_2^\top \boldsymbol{x}_n), \ldots, \sigma(\boldsymbol{w}_K^\top \boldsymbol{x}_n)$.
- The output layer computing a weighted combination of the outputs of the previous layer's hidden units, i.e. $y_n = \sum_{k=1}^K \pi_k \sigma(\boldsymbol{w}_k^\top \boldsymbol{x}_n)$
- Input layer to hidden layer connection weights are given by the K weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K$ and the hidden layer to output layer connection weights are given by $\pi_1, \ldots, \pi_K$.

8. Consider the following activation function: $h(x) = x\sigma(\beta x)$ where σ denotes the sigmoid function $\sigma(z) = \frac{1}{1+\exp(-z)}$. Show that, for appropriately chosen values of β, this activation function can approximate (1) the linear activation function, and (2) the ReLU activation function. (**5 marks**)

$h(x) = x\sigma(\beta x) = \frac{x}{1+\exp(-\beta x)}$.

- As $\beta \to 0$, $h(x) = x/2$, i.e., a linear activation function.
- As $\beta \to \infty$, consider two scenarios:
  (1) When $x < 0$, $h(x) = \frac{x}{\text{a very large number}} \approx 0$.
  (2) When $x >= 0$, $h(x) = \frac{x}{1+\text{a number close to zero}} \approx x$.
  Thus when $\beta \to \infty$, $h(x)$ behaves line ReLU activation function.

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

9. You are given an $N \times M$ matrix $\mathbf{R}$ with binary entries. Your goal is to approximate $\mathbf{R}$ using a product of two matrices $\mathbf{U} \in \mathbb{R}^{N \times K}$ and $\mathbf{V} \in \mathbb{R}^{M \times K}$ where K is usually smaller than N and M. In particular, assume $p(\mathbf{R}|\mathbf{U}, \mathbf{V}) = \text{Bernoulli}(\mathbf{R}|\sigma(\mathbf{U}\mathbf{V}^\top))$ where the sigmoid operation and Bernoulli are applied elementwise on their respective input matrices. Note that this is also equivalent to $p(R_{nm}|\boldsymbol{u}_n, \mathbf{v}_m) = \text{Bernoulli}(R_{nm}|\sigma(\boldsymbol{u}_n^\top \mathbf{v}_m))$, $\boldsymbol{u}_n^\top$ and $\mathbf{v}_m$ denote the n^{th} row of $\mathbf{U}$ and m^{th} column of $\mathbf{V}$, respectively.

   Give an ALT-OPT algorithm to learn $\mathbf{U}$ and $\mathbf{V}$ and show that it reduces to solving $N + M$ logistic regression problems in each iteration. For each logistic regression problem, what is the corresponding input matrix, labels, and the weight vector? (**14 marks**)

   Consider the matrix $\mathbf{R}$. Its n-th row $\mathbf{R}_{n,:}$ is an $1 \times M$ vector of binary entries. Note that we can express it as $p(\mathbf{R}_{n,:}|\mathbf{U}, \mathbf{V}) = \text{Bernoulli}(\mathbf{R}|\sigma(\boldsymbol{u}_n \mathbf{V}^\top))$. Note that the sigmoid is operating element-wise on the vector $\boldsymbol{u}_n \mathbf{V}^\top$. Also, to follow the standard convention, let's use the column notation, and write the same as $p(\mathbf{R}_{n,:}^\top|\mathbf{U}, \mathbf{V}) = \text{Bernoulli}(\mathbf{R}|\sigma(\mathbf{V}\boldsymbol{u}_n))$. If $\mathbf{V}$ is fixed, this is a standard logistic regression problem with $\mathbf{V}$ acting as the $M \times K$ input (feature) matrix, $\mathbf{R}_{n,:}^\top$ as the $M \times 1$ vector of binary labels, and $\boldsymbol{u}_n \in \mathbb{R}^K$ as the unknown weight vector to be estimated.

   Likewise, the m-th column of $\mathbf{R}$, i.e., $\mathbf{R}_{:,m}$ is an $N \times 1$ vector of binary entries. Note that we can express it as $p(\mathbf{R}_{:,m}|\mathbf{U}, \mathbf{V}) = \text{Bernoulli}(\mathbf{R}|\sigma(\mathbf{U}\mathbf{v}_m))$. If $\mathbf{U}$ is fixed, this is a standard logistic regression problem with $\mathbf{V}$ acting as the $M \times K$ input (feature) matrix, $\mathbf{R}_{:,n}$ as the $M \times 1$ vector of binary labels, and $\boldsymbol{u}_n \in \mathbb{R}^K$ as the unknown weight vector to be estimated.

   So we now know how to estimate each $\boldsymbol{u}_n$ $(n = 1, \ldots, N)$ given $\mathbf{V}$ and how to estimate each $\mathbf{v}_m$ $(m = 1, \ldots, M)$ given $\mathbf{U}$. We will use the ALT-OPT algorithm to estimate each of these and in each iteration of this ALT-OPT algorithm, we will be learning $N + M$ logistic regression models, with weight vectors given by $\{\boldsymbol{u}_n\}_{n=1}^N$ and $\{\mathbf{v}_m\}_{m=1}^M$.

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023

Some distributions and their properties:

- For $x \in \mathbb{R}$, the PDF of univariate Gaussian: $\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\{-\frac{(x-\mu)^2}{2\sigma^2}\}$. If using precision $\beta = 1/\sigma^2$, the PDF is $\mathcal{N}(x|\mu,\beta^{-1}) = \sqrt{\frac{\beta}{2\pi}}\exp\{-\frac{\beta}{2}(x-\mu)^2\}$.
- For $x \in \mathbb{R}^D$, D-dimensional Gaussian: $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}}\exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\}$.
- PDF of a Bernoulli random variable x is $p(x) = \text{Bernoulli}(x|\mu) = \mu^x(1-\mu)^{1-x}$ where $\mu \in (0,1)$ is the success probability.
- Covariance of a vector random variable X is $\text{cov}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$ where $\mathbb{E}$ denotes expectation.

Some other useful results:

- Given a joint distribution of two groups of random variables $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ which is Gaussian with mean vector $\boldsymbol{\mu} = \begin{pmatrix}\boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b\end{pmatrix}$ and covariance matrix $\boldsymbol{\Sigma} = \begin{bmatrix}\boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb}\end{bmatrix}$, the marginal and conditional distributions for Gaussians are:
  $p(\boldsymbol{x}_a) = \mathcal{N}(\boldsymbol{x}_a|\boldsymbol{\mu}_a,\boldsymbol{\Sigma}_{aa})$,
  $p(\boldsymbol{x}_a|\boldsymbol{x}_b) = \mathcal{N}(\boldsymbol{x}_a|\boldsymbol{\mu}_{a|b},\boldsymbol{\Sigma}_{a|b})$
  where $\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}$, and $\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\boldsymbol{x}_b - \boldsymbol{\mu}_b)$
- $\frac{\partial}{\partial\boldsymbol{\mu}}[\boldsymbol{\mu}^\top\mathbf{A}\boldsymbol{\mu}] = [\mathbf{A}+\mathbf{A}^\top]\boldsymbol{\mu}$, $\frac{\partial}{\partial\mathbf{A}}\log|\mathbf{A}| = \mathbf{A}^{-\top}$, $\frac{\partial}{\partial\mathbf{A}}\text{trace}[\mathbf{AB}] = \mathbf{B}^\top$

Name:

Roll No.: **Dept.**:

IIT Kanpur
CS771A (IML)
End-sem Exam
Date: November 22, 2023