Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018

Instructions: *Total:* **120 marks**

1. This question paper contains a total of 10 pages (10 sides of paper). Please verify.
2. Please write your name, roll number, department on **every side of every sheet** of this booklet.
3. You may write your answers using pencil but your handwriting should be bold and prominently visible.
4. **Important:** Please do not give derivations/elaborate steps unless specifically asked for it. Feel free to use standard results (e.g., solution of least squares regression) without deriving them from scratch.
5. The last page of the question paper lists some formulae if you need them.

Section 1 (True or False: 12 X 1 = 12 marks). For each of the following simply write **T** or **F** in the box.

1. F The kernel SVM weight vector $\boldsymbol{w}$ can be written explicitly as a finite dimensional vector only when using a linear kernel (assuming we aren't using any approximations, such as landmarks).
2. F Learning a single hidden layer neural network with infinite many hidden units is equivalent to learning a kernelized model with an RBF kernel.
3. T Both alternating optimization (ALT-OPT), as well as the expectation maximization (EM) algorithm, are sensitive to initialization.
4. T It is possible to get closed form solutions for all the parameters of a fully supervised generative classification model with Gaussian class-conditionals.
5. F A K-nearest neighbors classifier that uses Euclidean distances can only learn a linear decision boundary regardless of the value of K.
6. T A depth-1 decision tree will usually have a higher bias than a depth-5 decision tree (here "bias" is used in the sense of this word as in the bias-variance trade-off).
7. F If the training inputs and test inputs for a classification problem are drawn from the same distribution then the test error of the learned model will be zero.
8. F Iteration $t+1$ of Adaboost is faster than iteration t because iteration $t+1$ trains only using the misclassified examples from iteration t.
9. F MAP estimation for a parameter, when using a Gaussian prior with zero mean and spherical covariance, is equivalent to doing MLE for the parameter.
10. T If the gap between training and test error is large for a model then retraining the model with larger training set may reduce the gap.
11. F Probabilistic PCA with noise variance equal to zero and classic PCA will give the same solution for the projection matrix, assuming mean-centered data for both the methods.
12. F A feedforward neural network's output layer computes a convex combination of the outputs of the last hidden layer nodes.

Section 2 (MCQ: 12 x 2 = 24 marks). **Tick-mark** ☑ all the options that you think are correct. **Important:** Marks for a question will be awarded only when all correct options (and only those) are selected.

1. Which of these can be used for regression? A Decision Tree, B K-nearest neighbor, C Perceptron, D Feedforward neural network, E Logistic regression.
2. Which of these can be kernelized? A K-nearest neighbors, B Decision trees, C K-means clustering, D Principal Component Analysis, E Prototype based classification.
3. Which of these are linear dimensionality reduction methods: A Probabilistic PCA, B Standard PCA, C Fisher Discriminant Analysis, D, Stochastic Neighbor Embedding, E Locally linear embedding.
4. Which of these objectives are non-differentiable? A Squared loss with ℓ_1 regularizer, B Hinge loss with ℓ_2 reg., C Hinge loss with ℓ_1 reg., D Huber loss with ℓ_2 reg., E Huber loss with ℓ_1 reg.

Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018

---

5. Which of these can only learn linear decision boundaries? $\boxed{\text{A}}$ SVM with quadratic kernel, $\boxed{\text{B}}$ Decision tree classifier, $\boxed{\text{C}}$ Prototype based classification with Euclidean distances, $\boxed{\text{D}}$ Single hidden layer neural net with ReLU activations, $\boxed{\text{E}}$ Logistic regression with score being linear combination of the features.

6. Which of these learning problems/sub-problems require constrained optimization? $\boxed{\text{A}}$ Solving for the mixing proportion weights in a mixture model, $\boxed{\text{B}}$ Learning the standard Perceptron, $\boxed{\text{C}}$ Learning PPCA, $\boxed{\text{D}}$ Learning the kernel SVM, $\boxed{\text{E}}$ Value-iteration based policy learning in reinforcement learning.

7. Which of the following are true? $\boxed{\text{A}}$ When the regularization hyperparam. tends to infinity, regularization becomes ineffective, $\boxed{\text{B}}$ ℓ_1 norm is non-convex, $\boxed{\text{C}}$ ℓ_2 norm is convex, $\boxed{\text{D}}$, ℓ_1 norm promotes non-negativity, $\boxed{\text{E}}$ Using a Laplace prior is equivalent to using an ℓ_2 regularizer.

8. The output of a matrix factorization model learned using only user-item ratings matrix can be used to: $\boxed{\text{A}}$ Find other users similar to a given user, $\boxed{\text{B}}$ Find other items similar to a given item, $\boxed{\text{C}}$ Recommend existing items to new users, $\boxed{\text{D}}$ Learn clusters of items, $\boxed{\text{E}}$ Learn clusters of users.

9. How can we turn a linear classifier into a nonlinear one? $\boxed{\text{A}}$ First project the inputs to a low-dim space using PCA, $\boxed{\text{B}}$ First project the inputs to a low-dim space using Fisher Discriminant Analysis, $\boxed{\text{C}}$ Use it as a base learner in Adaboost, $\boxed{\text{D}}$ Use scores of $K > 1$ such classifiers to get K new features and learn another linear classifier on those features. $\boxed{\text{E}}$ Cluster inputs and learn a linear classifier for each cluster.

10. Posterior can be computed in closed form for: $\boxed{\text{A}}$ Linear regression with Gaussian likelihood, zero mean Gaussian prior, and fixed hyperparams, $\boxed{\text{B}}$ Linear regression with Gaussian likelihood, non-zero mean Gaussian prior, and fixed hyperparams $\boxed{\text{C}}$ Logistic regression with Gaussian prior, $\boxed{\text{D}}$ Bernoulli coin-toss model with Beta prior on coin's bias, $\boxed{\text{E}}$ Gaussian mean estimation with Gaussian prior on mean.

11. Which of the following are true about KNN: $\boxed{\text{A}}$ Very fast at test time, $\boxed{\text{B}}$ Tends to underfit as K increases, $\boxed{\text{C}}$ Have zero error on training data, $\boxed{\text{D}}$ Equivalent to prototype based classification for $K = 1$, $\boxed{\text{E}}$ Training them is computationally very expensive.

12. Which of the following is true about support vector machines (SVM)? $\boxed{\text{A}}$ They are faster than decision trees at test time, $\boxed{\text{B}}$ Multiclass SVMs are equivalent to softmax regression, $\boxed{\text{C}}$ For linear SVM, every training example is a support vector, $\boxed{\text{D}}$ Maximizing the SVM margin is equivalent to maximizing the ℓ_2 norm of SVM weight vector, $\boxed{\text{E}}$ Increasing margin leads to more number of misclassified training examples.

Section 3 (Short Answer: 8 x 4 = 32 marks). Write your answers precisely and concisely in the provided box.

1. Consider a generative model for binary classification. Suppose each input has 2 features, where the first feature takes one of 5 possible values and the second feature is binary. With naive Bayes assumption, how many parameters would we need to learn for this generative classification model. Justify your answer.

$X = [x_1, x_2]$

$P(x_1|y=0) = \text{multinoulli}(\pi_1^{(0)}, \pi_2^{(0)}, \pi_3^{(0)}, \pi_4^{(0)}, \pi_5^{(0)}) \Rightarrow 5$ params

$P(x_2|y=0) = \text{Bern}(\mu^{(0)}) \Rightarrow 1$ param

(actually 4 are sufficient since $\sum \pi_k = 1$)

Likewise for $P(x_1|y=1)$ and $P(x_2|y=0) \Rightarrow 5+1$

$P(y) = \text{Bern}(\theta) \Rightarrow 1$ param

Thus total $= 2 \times 6 + 1 = 13$

Also acceptable: $2 \times (4+1) + 1 = 11$

Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018

2. You are given N inputs $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$. Suppose, for each $\boldsymbol{x}_n \in \mathbb{R}^D$, you want to obtain a K dimensional and *non-sparse* feature vector $\boldsymbol{z}_n$, where the sum of the K features is one. Briefly describe how you would compute such feature vectors $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N\}$, using a K-means clustering algorithm on this data?

Compute soft assignment for each x_n (just like soft K-means)
e.g. $$z_{nk} = \frac{\exp(-\|x_n-\mu_k\|^2)}{\sum_{l=1}^{K} \exp(-\|x_n-\mu_l\|^2)} \Rightarrow \sum z_{nk} = 1$$

3. Consider a linear model with a regularizer $R(\boldsymbol{w}) = ||\boldsymbol{w}||^2 + \sum_{d=1}^{D}\sum_{d'=d+1}^{D}(w_d - w_{d'})^2$. What will be the effect of such a regularizer on $\boldsymbol{w}$ when minimizing the objective $\sum_{n=1}^{N} \ell(y_n, \boldsymbol{w}^\top \boldsymbol{x}_n) + \lambda R(\boldsymbol{w})$ w.r.t. $\boldsymbol{w}$?

It will promote w's entries to not just be small but also similar to each other. Note: Suppose someone gave us a graph A (DXD binary matrix) with $A_{dd'} = 1$ denoting that features d and d' are similar then we can use $R(w) = \|w\|^2 + \sum_d \sum_{d'} A_{dd'} (w_d - w_{d'})^2$

4. In at most 1-3 sentences (preferably only words, no equations!), describe how additional unlabeled data can be utilized within an algorithm for learning the parameters of a generative classification model.

We can use something like EM or ALT-OPT to learn the parameters where, in each iteration, we compute a guess for the label of each unlabeled example and use these guesses when updating the parameters (we did it in one of the HW).

5. Can we compute the squared ℓ_2 norm $||\boldsymbol{w}||^2$ of the kernel ridge regression weights $\boldsymbol{w} = \sum_{n=1}^{N} \alpha_n \phi(\boldsymbol{x}_n)$, assuming ϕ to be the feature mapping of an RBF kernel? If yes, show how it can be done. If no, clearly state why it can't be done. Also answer the same question if we want to compute the ℓ_1 norm of $\boldsymbol{w}$.

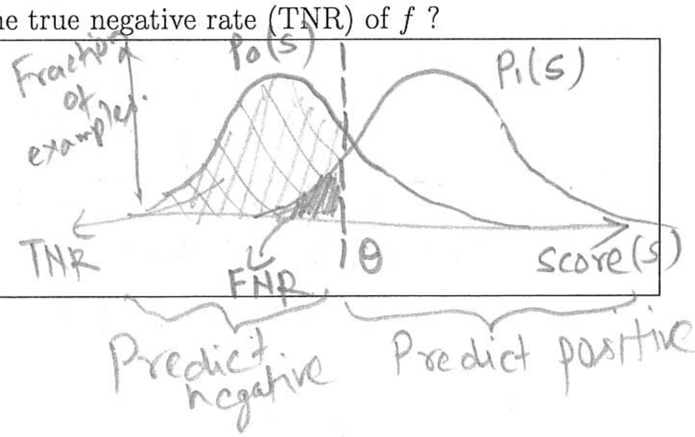
$$\|w\|^2 = w^T w = \left(\sum_{n=1}^{N} \alpha_n \phi(x_n)\right)^T \left(\sum_{m=1}^{N} \alpha_m \phi(x_m)\right) = \sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m \phi(x_n)^T \phi(x_m)$$

Thus $\|w\|^2 = \sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m K(x_n, x_m) = \alpha^T K \alpha$ (in matrix-vector form)

However, we can't compute ℓ_1 norm since $\phi(x_n)$ is infinite-dim

6. Assume a model f applied to a two-class data. Suppose the PDF of scores $s \in (-\infty, \infty)$ of f on positive examples is $p_1(s)$, whereas the PDF of f's scores on nagative examples is $p_0(s)$ (assume positive examples to be 1 and negative examples to be 0). Suppose F_0 denotes the CDF of p_0 and F_1 denotes the CDF of p_1. What's the false negative rate (FNR) and the true negative rate (TNR) of f ?

For a threshold θ
FNR = $F_1(\theta)$
TNR = $F_0(\theta)$



Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018

7. Suppose you have two coins c_1 and c_2 with biases $\pi_1 \in (0,1)$ and $\pi_2 \in (0,1)$, respectively. You have another coin c_3 with bias $\mu \in (0,1)$. You do two coin tosses as follows: First, you toss coin c_3. If it shows heads, you toss coin c_1; otherwise you toss coin c_2. Denote the outcome of the second toss (i.e., when you toss c_1 or c_2) as $x \in \{0,1\}$. What is the marginal *distribution* $p(x)$? Clearly write down its expression.

Mixture of two Bernoulli distributions

$p(x) = \mu \times \text{Bern}(x|\pi_1) + (1-\mu)\text{Bern}(x|\pi_2)$

8. Taking the example of a single hidden layer feedforward neural network, show that it is necessary to have nonlinearities in the hidden nodes, in the absence of which the network would reduce to a linear model.

Without nonlinearity in hidden layer nodes: $h_n = Wx_n$

$y_n = v^T h_n = v^T W x_n = \hat{w}^T x_n$ where $\hat{w} = (v^T W)^T$

Therefore we get a linear model if there is no nonlinearity.

Section 4 (5 problems: 5 x 8 = 40 marks). Write your answers precisely and concisely in the provided box.

1. Derive and write down the SGD update (minibatch size = 1) for the linear regression model with squared loss $\sum_{n=1}^{N}(y_n - \boldsymbol{w}^\top \boldsymbol{x}_n)^2$. Using the SGD update equation, formally show that each SGD update does the *right* thing, i.e., it *improves* the model's prediction on the current example $(\boldsymbol{x}_n, y_n)$.

$\nabla L(w) = -2(y_n - w^T x_n)x_n$ (using a single example (x_n, y_n))

SGD update: $w^{(t+1)} = w^{(t)} + \eta(y_n - w^{(t)T}x_n)x_n$ (η subsumes the factor of 2 also $\eta > 0$)

Suppose $y_n > w^{(t)T}x_n$, then note that

$w^{(t+1)T}x_n = w^{(t)T}x_n + \eta\underbrace{(y_n - w^{(t)T}x_n)x_n^T x_n}_{>0}$

$\Rightarrow w^{(t+1)T}x_n > w^{(t)T}x_n$

which is an improvement since $w^T x_n$ value is moving closer to y_n.

can use a similar argument for the case when $y_n < w^{(t)T}x_n$. In this case, $w^{(t+1)T}x_n < w^{(t)T}x_n$, which is again an improvement since $w^T x_n$ value is moving closer to y_n.

Name:

Roll No.: Dept.:

2. Consider the prototype based classification model, given training data $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, where input $\boldsymbol{x}_n \in \mathbb{R}^D$ and label $y_n \in \{-1, +1\}$. Suppose we have mapped the inputs to a new feature space ϕ that has an associated kernel function $k(.,.)$. Show that the prediction for a new test input $\boldsymbol{x}_*$ can be written in form of $y_* = \text{sign}[f(\boldsymbol{x}_*)]$ and clearly write down the expression for $f(\boldsymbol{x}_*)$. The expression for $f(\boldsymbol{x}_*)$ must be only in terms of the kernel function k, and must not contain the feature mapping ϕ in it.

In prototype based classification (unkernelized case)

$$y_* = \text{Sign}\Big[\underbrace{\|x_* - \mu_-\|^2 - \|x_* - \mu_+\|^2}_{f(x_*)}\Big]$$

In the kernelized case $f(x_*) = \|\phi(x_*) - \phi(\mu_-)\|^2 - \|\phi(x_*) - \phi(\mu_+)\|^2$

where $\phi(\mu_-)$ and $\phi(\mu_+)$ are means of negative and positive examples in the ϕ space, e.g. $\boxed{\phi(\mu_-) = \frac{1}{N_-}\sum_{n:y_n=-1}\phi(x_n)}$

substituting and expanding, we get

$$f(x_*) = \Big\|\phi(x_*) - \frac{1}{N_-}\sum_{n:y_n=-1}\phi(x_n)\Big\|^2 - \Big\|\phi(x_*) - \frac{1}{N_+}\sum_{n:y_n=1}\phi(x_n)\Big\|^2$$

Common mistake many of you have made.

$$= \frac{2}{N_+}\sum_{n:y_n=1}\underbrace{\phi(x_*)^T\phi(x_n)}_{k(x_*,x_n)} - \frac{2}{N_-}\sum_{n:y_n=-1}\underbrace{\phi(x_*)^T\phi(x_n)}_{k(x_*,x_n)} + \frac{1}{N_-^2}\sum_{\substack{n,m:\\ y_n=y_m=-1}}\underbrace{\phi(x_n)^T\phi(x_m)}_{k(x_n,x_m)} - \frac{1}{N_+^2}\sum_{\substack{n,m:\,y_n=y_m\\ =+1}}\underbrace{\phi(x_n)^T\phi(x_m)}_{k(x_n,x_m)}$$

Important: $\phi(\mu_-)$ is not ϕ simply applied to $\frac{1}{N_-}\sum_{n:y_n=-1}x_n$ (likewise for $\phi(\mu_+)$) but

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 for each mean μ_k that makes use of this information. For any iteration of K-means, given the current observation-to-cluster assignments $\{z_1, \ldots, z_N\}$, and your proposed prior distribution, derive the update equation for each mean.

$P(\mu_K) = \mathcal{N}(\mu_K^*, \lambda^{-1} I_D) \Rightarrow$ corresponds to a regularizer of the form $\lambda\|\mu_K - \mu_K^*\|$ for each μ_K.

The K-mean objective using this additional regularizer can be written as (assuming Z given)

$$L(\mu_1 .. , \mu_K) = \sum_{K=1}^{K}\Big[\sum_{n:z_n=K}\|x_n - \mu_K\|^2 + \lambda\|\mu_K - \mu_K^*\|^2\Big]$$

For each μ_K, we need to solve

$$\hat{\mu}_K = \arg\min_{\mu_K} \sum_{n:z_n=K}\|x_n - \mu_K\|^2 + \lambda\|\mu_K - \mu_K^*\|^2$$

so μ_K^* is like an additional data point with weight λ.

Taking derivative w.r.t μ_K and setting to zero:

$$\sum_{n:z_n=K}\big[-2(x_n - \mu_K) + 2\lambda(\mu_K - \mu_K^*)\big] = 0 \Rightarrow \boxed{\hat{\mu}_K = \frac{\sum_{n:z_n=K} x_n + \lambda\mu_K^*}{N_K + \lambda}}$$

Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018

4. Consider learning a linear regression model by minimizing the squared loss functio $\sum_{n=1}^{N}(y_n - \boldsymbol{w}^\top \boldsymbol{x}_n)^2$. Suppose we decide to mask out or "drop" each feature x_{nd} of each input $\boldsymbol{x}_n \in \mathbb{R}^D$, independently, with probability $1-p$ (equivalently, retaining the feature with probability p). Masking or dropping out basically means that we will set the feature x_{nd} to 0 with probability $1-p$. Essentially, it would be equivalent to replacing each input $\boldsymbol{x}_n$ by $\tilde{\boldsymbol{x}}_n = \boldsymbol{x}_n \circ \boldsymbol{m}_n$, where $\circ$ denotes elementwise product and $\boldsymbol{m}_n$ denotes the $D \times 1$ binary mask vector with $m_{nd} \sim \text{Bernoulli}(p)$ ($m_{nd} = 1$ means the feature x_{nd} was retained; $m_{nd} = 0$ means the feature x_{nd} was masked/zeroed).

Let us now define a new loss function using these masked inputs as follows: $\sum_{n=1}^{N}(y_n - \boldsymbol{w}^\top \tilde{\boldsymbol{x}}_n)^2$. Show that minimizing the *expected* value of this new loss function (where the expectation is used since the mask vectors $\boldsymbol{m}_n$ are random) is equivalent to minimizing a **regularized** loss function. Clearly write down the expression of this regularized loss function. (PS: You did something like this in Practice Set 1).

This is basically what the DROPOUT technique of neural nets does! REGULARIZATION (in each layer)

Let's denote $\tilde{L}(w) = \sum_{n=1}^{N}(y_n - w^T\tilde{x}_n)^2$. Its expectation will be

$$\mathbb{E}[\tilde{L}(w)] = \sum_{n=1}^{N}\mathbb{E}\left[y_n^2 + (w^T\tilde{x}_n)^2 - 2y_n w^T\tilde{x}_n\right] = \sum_{n=1}^{N} y_n^2 + \mathbb{E}[(w^T\tilde{x}_n)^2] - 2y_n\mathbb{E}[w^T\tilde{x}_n]$$

Note that $\mathbb{E}[(w^T\tilde{x}_n)^2] = \text{Var}[w^T\tilde{x}_n] + \mathbb{E}[w^T\tilde{x}_n]^2$. Plugging it in expression above,

$$\mathbb{E}[\tilde{L}(w)] = \sum_{n=1}^{N} \underbrace{y_n^2 - 2y_n\mathbb{E}[w^T\tilde{x}_n] + \mathbb{E}[(w^T\tilde{x}_n)^2]}\, + \text{Var}[w^T\tilde{x}_n]$$

$$= \sum_{n=1}^{N}(y_n - \mathbb{E}[w^T\tilde{x}_n])^2 + \text{Var}[w^T\tilde{x}_n] \leftarrow \text{We're pretty much at the answer now!}$$

Note that $\mathbb{E}[w^T\tilde{x}_n] = w^T\mathbb{E}[\tilde{x}_n] = pw^Tx_n$ (Since $\tilde{x}_n = x_n \odot m_n$ and $\mathbb{E}[m_{nd}] = p$)

Also, $\text{Var}[w^T\tilde{x}_n] = w^T\text{Cov}(\tilde{x}_n)w = w^T\begin{bmatrix} p(1-p)x_{n1}^2 & & 0 \\ & \ddots & \\ 0 & & p(1-p)x_{nD}^2 \end{bmatrix}w = \underline{w^T\Sigma_n w}$

Like an L2 regularizer

Thus $\mathbb{E}[\tilde{L}(w)] = \sum_{n=1}^{N}(y_n - pw^Tx_n)^2 + w^T\Sigma w$ where Σ is the above diagonal matrix summed over all n

5. Consider the full (not truncated) singular value decomposition (SVD) of an $N \times D$ matrix $\mathbf{X}$. Denote it as $\mathbf{X} = \mathbf{U}\Lambda\mathbf{V}^\top$. Show that the left singular vectors $\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N]$ are also the eigenvectors of $\mathbf{X}\mathbf{X}^\top$. Also show that the the right singular vectors $\mathbf{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_D]$ are also the eigenvectors of $\mathbf{X}^\top\mathbf{X}$.

Note that $XX^T = U\Lambda V^T(U\Lambda V^T)^T = U\Lambda V^T V\Lambda U^T = U\Lambda^2U^T$ (using $V^TV = I$)

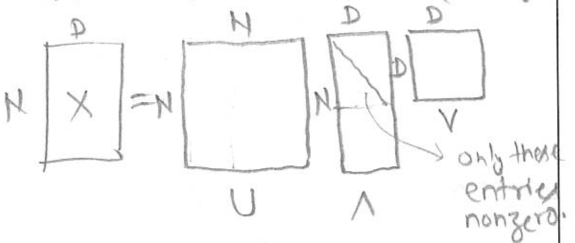
$\underline{(XX^T)u_n} = (U\Lambda^2U^T)u_n = \underline{\lambda_n^2 u_n}$ (using orthonormality of $U = [u_1, u_2, \ldots, u_N]$)

$\Rightarrow$ u_n is an eigenvector of XX^T

Likewise $X^TX = (U\Lambda V^T)^T(U\Lambda V^T) = V\Lambda U^TU\Lambda V^T = V\Lambda^2V^T$ (using $U^TU = I$)

$\underline{(X^TX)v_d} = (V\Lambda^2V^T)v_d = \underline{\lambda_d^2 v_d}$

$\Rightarrow$ v_d is an eigenvector of X^TX.



Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018

Section 5 (1 problem: 12 marks). Write your answers precisely and concisely in the provided box.

1. Assume you are given N examples $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, with each $\boldsymbol{x}_n \in \mathbb{R}^D$ and $y_n \in \mathbb{R}$. Assume the following generative story for each $(\boldsymbol{x}_n, y_n)$: (1) Generate $z_n \sim \text{multinoulli}(\pi_1, \ldots, \pi_K)$, (2) Generate the inputs $\boldsymbol{x}_n \sim \mathcal{N}(\mu_{z_n}, \Sigma_{z_n})$, and (3) Generate the outputs as $y_n \sim \mathcal{N}(\boldsymbol{w}_{z_n}^\top \boldsymbol{x}_n, \beta^{-1})$.

   Your goal is to estimate the parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k, \boldsymbol{w}_k\}_{k=1}^K$ of this model. Assume β to be fixed.

   - You have to derive an EM algorithm to compute the posterior distribution over unknowns $\mathbf{Z} = \{z_1, \ldots, z_N\}$ and point estimate (MLE) of unknowns Θ. To do so, first write down the expression for the complete-data log-likelihood (CLL) for the model, and simplify it (ignore the constants).
   - Now derive the necessary expressions that you would need for the EM algorithm for this model. If some of these derivations are obvious/familiar to you, you can skip those and directly write down the final expressions (but these expressions better be correct; no partial marks can be given for incorrect expressions in such a case :)). Also give a brief sketch of the overall EM algorithm.
   - Assuming $\pi_k = 1/K, \forall k$, derive the ALT-OPT algorithm for this model (you may use the results from the above EM algorithm to get the ALT-OPT algorithm directly, without deriving from scratch). The ALT-OPT algorithm will compute point estimates for both $\mathbf{Z}$ and Θ. Also give a brief sketch of the overall ALT-OPT algorithm.

This is basically a mixture of experts model where we are also modeling x_n

$$- \; CLL(\Theta) = \log \prod_{n=1}^N P(x_n, y_n, z_n | \Theta)$$

$$= \log \prod_{n=1}^N \prod_{k=1}^K \left[P(x_n | z_n = k) P(y_n | z_n = k, x_n) P(z_n = k) \right]^{z_{nk}}$$

(ignoring θ from the notation)

$$= \sum_{n=1}^N \sum_{k=1}^K z_{nk} \left[\log \underbrace{P(x_n | z_n = k)}_{N(x_n|\mu_k, \Sigma_k)} + \log \underbrace{P(y_n | z_n = k, x_n)}_{N(y_n | w_k^T x_n, \beta^{-1})} + \log \underbrace{P(z_n = k)}_{\pi_k} \right]$$

$$= \sum_{n=1}^N \sum_{k=1}^K z_{nk} \left[-\frac{1}{2} \log|\Sigma_k| - \frac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) - \frac{\beta}{2}(y_n - w_k^T x_n)^2 + \log \pi_k \right]$$

- We need expected CLL. The only expectation we need is $\mathbb{E}[z_{nk}]$ which is the same as the posterior probability of $z_{nk} = 1$,

$$P(z_{nk} = 1 | x_n, y_n, \theta) \propto P(z_{nk} = 1) P(x_n | \underbrace{z_{nk} = 1}_{\text{same as } z_n = k}) P(y_n | \underbrace{z_{nk} = 1}_{\text{same as } z_n = k}, x_n)$$

$\leftarrow$ Unlike GMM or standard mixture of experts we have two likelihood terms, one for x_n, one for y_n

Thus $\mathbb{E}[z_{nk}] \propto \pi_k N(x_n | \mu_k, \Sigma_k) N(y_n | w_k^T x_n, \beta^{-1})$
(which can be easily normalized).

suppose $\gamma_{nk} = \mathbb{E}[z_{nk}]$, $N_k = \sum_{n=1}^N \gamma_{nk}$

Given γ_{nk} from the E step, the M step is straightforward. μ_k, Σ_k, π_k updates are just like GMM updates of these, e.g.

(if needed, you may continue the answer in the box on the next page)

$$\pi_k = \frac{N_k}{N}, \quad \mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} x_n, \quad \Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} (x_n - \mu_k)(x_n - \mu_k)^T$$

Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018

The update of W_K is also like the standard mixture of experts with $\hat{W}_K = \left(\sum_{n=1}^{N} \gamma_{nk} x_n x_n^T\right)^{-1}\left(\sum_{n=1}^{N} \gamma_{nk} y_n x_n\right)$

ALT-OPT with $\pi_K = \frac{1}{K}$ will be identical, except that (1) we don't need to estimate π_K

(2) z_n will be computed as follows

$$z_n = \arg\min_{k\in\{1..K\}} \left[-\log p(x_n|z_n=k) - \log p(y_n|z_n=k, x_n)\right]$$

$$= \arg\min_{k\in\{1..K\}} \left[\frac{1}{2}\log|\Sigma_k| + \frac{1}{2}(x_n-\mu_k)^T\Sigma_k^{-1}(x_n-\mu_k) + \frac{\beta}{2}(y_n - W_k^T x_n)^2\right]$$

(Note that this is the same as

$$z_n = \arg\max_k p(z_n=k|x_n,y_n,\Theta)$$

$$= \arg\max_k p(x_n,y_n|z_n=k,\Theta)\, \underline{p(z_n=k|\Theta)} \quad \frac{1}{K}$$

$$= \arg\min_k -\log p(x_n,y_n|z_n=k,\Theta))$$

We can use the optimal value of z_n to create a one-hot vector $\gamma_n = [\gamma_{n1} .. \gamma_{nK}]$

(3) Updates of μ_K, Σ_K, W_K will have an identical form as in the EM case but since γ_n is a one-hot vector, only the points with $\gamma_{nk}=1$ will contribute to the update of μ_K, Σ_K, W_K. (Recall HW 3 problem on mixture of experts).

Skipping the ALT-OPT & EM sketch (Obvious!).

Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018

Some formulae you might need

- Bernoulli: $\text{Bernoulli}(x|p) = p^x(1-p)^{1-x}$. Expectation $\mathbb{E}[x] = p$, Variance $\text{var}[x] = p(1-p)$
- Univariate Gaussian PDF: $\mathcal{N}(x|\mu, \lambda^{-1}) = \sqrt{\frac{\lambda}{2\pi}}\exp(-\frac{\lambda}{2}(x-\mu)^2)$, $\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\{-\frac{(x-\mu)^2}{2\sigma^2}\}$
- Multivariate Gaussian PDF: $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}}\exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\}$. Trace-based representation: $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}}\exp\left\{-\frac{1}{2}\text{trace}\left[\boldsymbol{\Sigma}^{-1}\mathbf{S}\right]\right\}$, $\mathbf{S} = (\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top$.
- For $x_k \in \{0, N\}$ and $\sum_{k=1}^K x_k = N$, $\text{multinomial}(x_1, \ldots, x_K|N, \boldsymbol{\pi}) = \frac{N!}{x_1!\ldots,x_K!}\pi_1^{x_1}\ldots\pi_K^{x_K}$, where $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_K]$, s.t. $\sum_{k=1}^K \pi_k = 1$. The multinoulli is the same as multinomial with $N = 1$.
- $\frac{\partial \boldsymbol{x}^\top \boldsymbol{a}}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{a}^\top \boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{a}$, quadratic form: $\frac{\partial}{\partial \boldsymbol{x}}(\boldsymbol{x}-\boldsymbol{s})^\top\mathbf{W}(\boldsymbol{x}-\boldsymbol{s}) = 2\mathbf{W}(\boldsymbol{x}-\boldsymbol{s})$
- $\frac{\partial}{\partial \mu}[\boldsymbol{\mu}^\top\mathbf{A}\boldsymbol{\mu}] = [\mathbf{A}+\mathbf{A}^\top]\boldsymbol{\mu}$, $\frac{\partial}{\partial \mathbf{A}}\log|\mathbf{A}| = \mathbf{A}^{-\top}$, $\frac{\partial}{\partial \mathbf{A}}\text{trace}[\mathbf{AB}] = \mathbf{B}^\top$
- For a random variable vector $\boldsymbol{x}$, $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top] = \mathbb{E}[\boldsymbol{x}]\mathbb{E}[\boldsymbol{x}]^\top + \text{cov}[\boldsymbol{x}]$
- For a random scalar x, $\text{var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$

Name:

Roll No.: Dept.:

IIT Kanpur
CS771 Intro to ML
End-semester Examination
Date: November 29, 2018