AI/ML theory (CS 337)
Computer Science and Engineering
Indian Institute of Technology Bombay

Endsem
15-11-2022 Tuesday, Total time 3 hours

Instructions

1. It is an OPEN BOOK examination.
2. This paper has seven questions. The maximum marks is 50.
3. Write your name, roll number and the subject number (CS 337) on the top of each of your answer script.
4. There will be partial credits for subjective questions, if you have made substantial progress towards the answer. However there will be NO credit for rough work.
5. Please keep your answer sheets different from the rough work you have made. Do not attach the rough work with the answer sheet.
6. Total time for the examination is 3 hours.

1. Short questions

1.a In Gaussian Process regression with RBF kenel, for a test point $x^\star$, its predictive variance always decreases with increase in training dataset size. State True/ False.

1.a	/1	

True

1.b Is it true that if a function K satisfies $K(\mathbf{x}, \boldsymbol{y}) \geq 0$ for all $\mathbf{x}, \boldsymbol{y}$ then the kernel induced by it is positive semi definite. If yes, give a short proof else provide a counter example.

1.b	/1	

False. we can reuse the example in 2b.

1.c Recall that we saw in class that the set intersection kernel defined as $K_\cap(A, B) = \Pr(A \cap B)$ is a valid Kernel where A, B are subsets over the universe U. For both parts, give one to two lines proof.

i. Prove that the union complement kernel defined as $K_c(A, B) = 1 - \Pr(A \cup B)$ is a valid kernel.

1.c	/1	

$K_c(A, B) = 1 - \Pr(A \cup B) = \Pr(A^c \cap B^c)$. Thus a valid kernel.

ii. Prove that the agreement kernel defined as $K_a(A, B) = 1 - \Pr(A - B) - \Pr(B - A)$ where minus $(-)$ denotes the set difference operator is a valid kernel.

$K_a(A, B) = 1 - \Pr(A - B) - \Pr(B - A) = 1 - \Pr(A \cup B) + \Pr(A \cap B) = K_c(A, B) + K_\cap(A, B)$ and thus a valid kernel.

1.c	/1	

1.d Consider a linear regression problem with the training data: $\mathcal{D} : \{(2, 1), (4, 5)\}$ (usual convention of set input, label pair). Given a feature map function $\phi(x)$, the linear regression problem can be formulated as:

$$\mathbf{w}^* = \arg\min \sum_{i=1}^{N} (\mathbf{w}^T \phi(x_i) - y_i)^2$$

where N is the number of training samples, x_i is the input data point with label y_i and $\mathbf{w}$ represents the weight vector.

i. Consider the feature map $\phi(x) = x$. Analytically solve the linear regression problem and determine the value of $\mathbf{w}^*$. Note that for this part, $\mathbf{w}^*$ is a scalar.

$$\mathbf{w}^* = [1.1]$$

ii. Consider the feature map:

$$\phi(x) = \begin{bmatrix} x \\ 1 \end{bmatrix}$$

Analytically solve the linear regression problem and determine the value of $\mathbf{w}^*$.

$$\mathbf{w}^* = \begin{bmatrix} 2 \\ -3 \end{bmatrix}$$

iii. Consider the feature map:

because input don't have the corresponding feature for the last mapping var, thus weight corresponding to that is zero.

$$\phi(x) = \begin{bmatrix} x \\ 1 \\ -1 \end{bmatrix}$$

Analytically solve the linear regression problem and determine the value of $\mathbf{w}^*$.

$$\mathbf{w}^* = \begin{bmatrix} 2 \\ -3 \\ 0 \end{bmatrix}$$

Or any other solution where $w_2 + w_3 = -3$.

1.d	/1+2+1	

1.e Abir is training multilayer (deep) neural networks and notices that the training error is going down and converges to a local minimum. Then when he tests on the new data, the test error is abnormally high. What is probably going wrong and what do you recommend him to do? **Note:** There may be multiple correct options here. Marks will be awarded only if justifications are provided for each choice.

i. The training data size is not large enough. Collect a larger training data and retrain it.

ii. Play with learning rate and add regularization term to the objective function.

penalize the errors.

iii. Use a different initialization and train the network several times. Use the average of predictions from all nets to predict test data.

iv. Use the same training data but add two more hidden layers.

1.e	/2	

i,ii,iii

2. **2.a** Consider the kernel function $k : \mathbb{R}^2 \times \mathbb{R}^2 \longrightarrow \mathbb{R}$ defined as:

$$k(\mathbf{x}, \boldsymbol{z}) = (\mathbf{x}^\top \boldsymbol{z} + 1)^2$$

The above function is a valid kernel function. Hence, it can be represented in terms of a transformation function $\phi(.)$ such that $k(\mathbf{x}, \boldsymbol{z}) = \phi(\mathbf{x})^T \phi(\boldsymbol{z})$. Determine the function $\phi(.)$

2.a	/2	

$$\begin{aligned} k(x,z) &= (\mathbf{x}^\top \boldsymbol{z} + 1)^2 \\ &= (x_1 z_1 + x_2 z_2 + 1)^2 \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 1 + 2x_1x_2z_1z_2 + 2x_1z_1 + 2x_2z_2 \\ &= (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2).(1, \sqrt{2}z_1, \sqrt{2}z_2, \sqrt{2}z_1z_2, z_1^2, z_2^2) \\ &= \phi(\mathbf{x})^\top \phi(\boldsymbol{z}) \end{aligned}$$

2.b For vectors $\mathbf{x}, \boldsymbol{y} \in \mathbb{R}^d$, let us define the kernel function as $k(\mathbf{x}, \boldsymbol{y}) = \sum_{i=1}^{d} \max(\mathbf{x}[i], \boldsymbol{y}[i])$. Is k a valid kernel? If yes, identify the transformation ϕ, else provide a counter example.

Doubt.

2.b	/2	

We can assume the data $D = \{(1,2),(3,4)\}$. We can work out and check that the kernel matrix is not psd.

3. For this question, we will work with a hard margin SVM. Consider the 2-class classification problem in $\mathbb{R}^2$ with the following training examples:
Class +1: (1,0), (0,1)
Class -1: (0,0)

3.a What would be the optimal separating hyperplane? (No need to solve the optimization problem here.)

3.a /1

The hyperplane would be the line passing through the points (0,0.5), (0.5,0).

3.b Suppose we add one more training sample to Class +1, which is given by (1,1). Will the optimal separating hyperplane change? Explain.

3.b /1

No, the optimal separating hyperplane will not change because the point (1,1) does not affect the margin between the two classes.

3.c The separability constraint for this problem is defined as $y_i[\mathbf{w}^T\mathbf{x}_i + b] \geq 1$, where y_i is the class label, $\mathbf{x}_i$ is a training sample and $\mathbf{w}, b$ are parameters of the separating hyperplane. Now, suppose we change the separability constraint to $y_i[\mathbf{w}^T\mathbf{x}_i + b] \geq K$, where K is some positive number. Would the equation of the optimal separating hyperplane change? If yes, how would it change? If no, why not?

3.c /2

No, the optimal separating hyperplane will not change as the number K can be absorbed into the parameters $\mathbf{w}, b$, resulting in same results as the previous constraints.

Initial constraint was -1 here.

3.d Suppose we change the constraints to $\mathbf{w}^T\mathbf{x}_i + b \geq 1$ for Class +1 and $\mathbf{w}^T\mathbf{x}_i + b \leq -0.5$ for Class -1. What can you say about the new optimal separating hyperplane?.

Just put the values into the equation and find out the constraints.

With the standard SVM formulation, we see that the constraints on $\mathbf{w}, b$ w.r.t the given data are $\mathbf{w}_1 + b \geq 1, \mathbf{w}_2 + b \geq 1, b \leq -1$. With this we have that $w_1, w_2 \geq 2$
With the new formulation, we have the constraints as $\mathbf{w}_1 + b \geq 1, \mathbf{w}_2 + b \geq 1, b \leq -0.5$ which tells us that $w_1, w_2 \geq 1.5$. Because the objective function in hard margin SVM is $||\mathbf{w}||^2$, the optima will change.

3.d /3

4. Background: Self Attention

Given an input sequence of N query vectors $\boldsymbol{Q} := [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_N]^\top \in \mathbb{R}^{N\times D}$, N key vectors $\boldsymbol{K} := [\boldsymbol{k}_1, \cdots, \boldsymbol{k}_N]^\top \in \mathbb{R}^{N\times D}$ and N value vectors $\mathbf{V} := [\mathbf{v}_1, \cdots, \mathbf{v}_N]^\top \in \mathbb{R}^{N\times D}$, self-attention computes the output sequence $\mathbf{H} := [\boldsymbol{h}_1, \cdots, \boldsymbol{h}_N]^\top \in \mathbb{R}^{N\times D}$, as a weighted average, as follows:

$$\boldsymbol{h}_i = \sum_{j=1}^{N} \text{softmax}\Big(\boldsymbol{q}_i^\top \boldsymbol{k}_j / \sqrt{D}\Big) \mathbf{v}_j \tag{1}$$

Recall

Softmax: For any vector $\boldsymbol{z} \in \mathbb{R}^D$, the softmax function: $\mathbb{R}^D \to \mathbb{R}^D$ is applied as:

$$\text{softmax}(\boldsymbol{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{D} e^{z_j}}$$

4.a Write the matrix notation for Eqn. (1)

The output sequence $\mathbf{H} := [\boldsymbol{h}_1, \cdots, \boldsymbol{h}_N]^\top$ is then given by

$$\mathbf{H} = \text{softmax}\Big(\underline{\qquad ? \qquad}\Big)\underline{\quad ? \quad} \tag{2}$$

where the softmax function is applied to each row of the matrix.

4.a /2

$$\mathbf{H} = \text{softmax}\Big(\mathbf{Q}\mathbf{K}^\top/\sqrt{D}\Big)\mathbf{V}$$

4.b Nonparametric Regression Interpretation of Self-attention

We consider the key and value vectors as the training set $\{\boldsymbol{k}_j, \mathbf{v}_j\}$ for $j \in [N]$ to come from the following *nonparametric regression model*:

$$\mathbf{v}_j = f(\boldsymbol{k}_j) + \varepsilon_j, \tag{3}$$

where $\varepsilon_1, \ldots, \varepsilon_N$ are independent noises such that $\mathbb{E}(\varepsilon_j) = 0$. We assume that the key value pairs in the dataset $(\mathbf{v}_1, \boldsymbol{k}_1), \ldots, (\mathbf{v}_N, \boldsymbol{k}_N)$ are sampled i.i.d. from a joint distribution $p(\boldsymbol{k}, \mathbf{v})$. The goal is to estimate the function f using kernel regression.

We estimate $f(\boldsymbol{k}_j)$ as a conditional expectation $\mathbb{E}[\mathbf{v}|\boldsymbol{k}_j]$ where:

$$\mathbb{E}[\mathbf{v}|\boldsymbol{k}] = \int_{\mathbb{R}^D} \mathbf{v} \cdot p(\mathbf{v}|\boldsymbol{k})d\mathbf{v} = \int \frac{\mathbf{v} \cdot p(\mathbf{v}, \boldsymbol{k})}{p(\boldsymbol{k})}d\mathbf{v}. \tag{4}$$

Kernel density estimator: To estimate $p(\mathbf{v}, \boldsymbol{k})$ and $p(\boldsymbol{k})$, we employ the kernel density estimation approach. By using the isotropic Gaussian kernel with variance σ, we have the following estimators of $p(\mathbf{v}, \boldsymbol{k})$ and $p(\boldsymbol{k})$:

$$\hat{p}_\sigma(\mathbf{v}, \boldsymbol{k}) = \frac{1}{N}\sum_{j=1}^{N} \varphi_\sigma(\mathbf{v} - \mathbf{v}_j)\varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j), \qquad \hat{p}_\sigma(\boldsymbol{k}) = \frac{1}{N}\sum_{j=1}^{N} \varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j), \tag{5}$$

where $\varphi_\sigma(.)$ is the multivariate Isotropic Gaussian density function with diagonal covariance matrix $\sigma^2\mathbf{I}_D$. Recall the multivariate Gaussian density formula $\varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(\frac{-1}{2\sigma^2}||\boldsymbol{k} - \boldsymbol{k}_j||^2\right)$ Show that $\hat{p}_\sigma(\mathbf{v}, \boldsymbol{k})$ is a valid probability density.

4.b /1

4.c Show that Eqn. (4) and Eqn. (5) gives us the following estimator for the function f:

$$\widehat{f}_\sigma(\boldsymbol{k}) = \frac{\sum_{j=1}^{N} v_j \varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j)}{\sum_{j=1}^{N} \varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j)}. \tag{6}$$

4.c /2

$$\begin{aligned}\widehat{f}_\sigma(\boldsymbol{k}) &= \int_{\mathbb{R}^D} \frac{\mathbf{v} \cdot \hat{p}_\sigma(\mathbf{v}, \boldsymbol{k})}{\hat{p}_\sigma(\boldsymbol{k})}d\mathbf{v} = \int_{\mathbb{R}^D} \frac{\mathbf{v} \cdot \sum_{j=1}^{N} \varphi_\sigma(\mathbf{v} - \mathbf{v}_j)\varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j)}{\sum_{j=1}^{N} \varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j)}d\mathbf{v} \\ &= \frac{\sum_{j=1}^{N} \phi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j) \int \mathbf{v} \cdot \varphi_\sigma(\mathbf{v} - \mathbf{v}_j)d\mathbf{v}}{\sum_{j=1}^{N} \varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j)} = \frac{\sum_{j=1}^{N} v_j \varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j)}{\sum_{j=1}^{N} \varphi_\sigma(\boldsymbol{k} - \boldsymbol{k}_j)}.\end{aligned} \tag{7}$$

4.d Under the assumption that the keys $\boldsymbol{k}_j$ are normalized, *i.e.* $\|\boldsymbol{k}_j\|^2 = 1$, and by choosing $\sigma^2 = \sqrt{D}$, show that

$$\widehat{f}_\sigma(\boldsymbol{q}_i) = \sum_{j=1}^{N} \text{softmax}\Big(\boldsymbol{q}_i^\top \boldsymbol{k}_j/\sqrt{D}\Big)\mathbf{v}_j. \tag{8}$$

4.d /5

$$\begin{aligned}\widehat{f}_\sigma(\boldsymbol{q}_i) &= \frac{\sum_j^N \mathbf{v}_j \exp\left(-\|\boldsymbol{q}_i - \boldsymbol{k}_j\|^2/2\sigma^2\right)}{\sum_j^N \exp\left(-\|\boldsymbol{q}_i - \boldsymbol{k}_j\|^2/2\sigma^2\right)} \\ &= \frac{\sum_j^N \mathbf{v}_j \exp\left[-\left(\|\boldsymbol{q}_i\|^2 + \|\boldsymbol{k}_j\|^2\right)/2\sigma^2\right] \exp\left(\boldsymbol{q}_i \boldsymbol{k}_j^\top/\sigma^2\right)}{\sum_j^N \exp\left[-\left(\|\boldsymbol{q}_i\|^2 + \|\boldsymbol{k}_{j'}\|^2\right)/2\sigma^2\right] \exp\left(\boldsymbol{q}_i \boldsymbol{k}_j^\top/\sigma^2\right)}. \end{aligned} \tag{9}$$

$$\widehat{f}_\sigma(\boldsymbol{q}_i) = \frac{\sum_j^N \mathbf{v}_j \exp\left(\boldsymbol{q}_i \boldsymbol{k}_j^\top/\sigma^2\right)}{\sum_{j'}^N \exp\left(\boldsymbol{q}_i \boldsymbol{k}_{j'}^\top/\sigma^2\right)} = \sum_{j=1}^N \mathrm{softmax}\Big(\boldsymbol{q}_i^\top \boldsymbol{k}_j/\sigma^2\Big)\mathbf{v}_j. \tag{10}$$

5. **Gaussian Process** Consider a Gaussian Process $f(x) \sim GP(0, K)$ where $K(x, x') = cos\left(|x - x'|\frac{2\pi}{\rho}\right)$.

5.a Assume a training dataset with two points $\{(\rho, 1), (\frac{3\rho}{4}, 2)\}$ and the irreducible noise is $\sigma^2 = 0.25$. Compute the predictive distribution for a test point at $x^\star = 2\rho$.

5.a /2

The covariance vector for $x^\star$ with the training data is $k = [1, 0]$ and the covariance matrix over the training data is

$$K = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{11}$$

$$(K + \sigma^2)^{-1} = \begin{pmatrix} 0.8 & 0 \\ 0 & 0.8 \end{pmatrix} \tag{12}$$

$$\tag{13}$$

The mean of the predictive $k^\top(K + \sigma^2)^{-1}y = 0.8$. The predictive variance is $k(x^\star, x^\star) - k^\top(K + \sigma^2)^{-1}k = 0.2$

5.b Write the formula for the joint distribution for $Q_1 \sim f(x)$ and $Q_2 \sim f(x - 1)$

$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & cos(\frac{2\pi}{\rho}) \\ cos(\frac{2\pi}{\rho}) & 1 \end{bmatrix}\right) \tag{14}$$

5.b /3

6. Kmeans in 1 dimension *i.e.* $x \in \mathbb{R}$

In the following we assume a dataset of size n that is *sorted* $x_1 \leq \cdots \leq x_n \in \mathbb{R}$. The task at hand is to find an optimal solution for 1D k-Means and assign them to $k \in \mathbb{N}$ clusters. If the points $x_i, \cdots, x_j$ are clustered together, let us define its cluster cost as $CC(i,j) = \sum_{\ell=i}^{j}(x_\ell - \mu_{i,j})^2$ where $\mu_{i,j} = \frac{1}{j-i+1}\sum_{\ell=i}^{j} x_\ell$. Then k-Means cost is the sum of cluster costs over all the k clusters.

- Only for this part, we will partition the n points into $k = 2$ clusters. Prove that in the optimal cluster assignment, it is not possible to have $x_1, \cdots x_a, x_c$ in one cluster and $x_b, x_{c+1}, \cdots, x_n$ in the other cluster where $a < b < c$.

We compare the costs of the two partition schemes. The k-Means cost of the given scheme is $CC(1,\ldots,a,c) + CC(b,c+1,\ldots n)$ and for the second partition scheme is $CC(1,\ldots,a,b) + CC(c,\ldots n)$. We can see that the new scheme gives lesser cost and thus the given partition in the question is not optimal.

b /2

- It turns out that 1D Kmeans can be solved efficiently using Dynamic Programming. Formulate the DP recurrence relation and explain the algorithm.
Hint: Create a 2D DP matrix $D \in \mathbb{R}^{n\times k}$ where each entry $D[i][j]$ records the k-Means cost of clustering the points $x_1, \cdots, x_i$ $(i \leq n)$ into $j \leq k$ clusters.

The recurrence relation is

$$DD[i,j] = \min_{\ell\in[i-1]} DD[\ell, j-1] + CC(\ell,\ldots i) \tag{15}$$

b /4

- Show how to find the optimal cluster ids using the DP formulation above.

To find the optimal cluster assignment, we backtrack from $DD[n,k]$ finding the argmin_ℓ in the above recurrence relation. We would get a set of $k-1$ $\arg\min$ gives us the boundaries of the k clusters. It is trivial to find the clustrer assignments given these boundaries.

b /2

7. Vectorize the Code Assume that we have a data matrix $\boldsymbol{X}$ of size $\mathbb{R}^{N\times d}$ where N is the number of samples and d is the dimension of each sample. For $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^{d\times 1}$, we define its outer product as $\mathbf{x}_i\mathbf{x}_j^\top \in \mathbb{R}^{d\times d}$.

The task is to vectorize the following code under different cases. Explain your vectorization logic along with code for partial credit.
Hint: You may use the following notation: For a Tensor **A** *of shape (i, j, k),*
A*[None, :, :, None, :] gives the unsqueezed tensor of shape (1, i, j, 1, k)*

```
def naive_implementation(X):
    """
    Input X : Tensor of shape (N x d)
    Output: Tensor of shape (d x d)
    """
    N, d = X.shape
    output = torch.zeros(d, d)
    for i in range(N):
        for j in range(N):
            # A is of shape (d, 1)
            A = X[i,:].T
            # B is of shape (1, d)
            B = X[j,:]
            output += torch.matmul(A, B)
    return output
```

7.a **Case 1:** $d = 1, n > 1$ This is an easy case where the feature dimension is 1. Observe that the output is a tensor of shape $(1, 1)$

7.a	/1	

7.b **case 2:** $n = 1, d > 1$ In this case, we just have one sample in the data matrix. The output will be of shape (d, d).

7.b	/1	

7.c **Case 3:** $n > 1, d > 1$ This is a more involved case where we have many high dimensional data points. The output will be of shape (d, d).

7.c	/4	

Total: 50