Introduction to Machine Learning (CS 419M) **Quiz-2**
Computer Science and Engineering **30-03-2022 Wednesday**
Indian Institute of Technology Bombay

Instructions

1. The paper has only two question with multiple sub parts.
2. Write your answers on a paper, scan and submit them at the end of the exam.
3. Write your name, roll number and the subject number (CS 419M) 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. You should ONLY upload the answer sheets.

1. **1.a** One of the most commonly used kernels in SVM is the Gaussian RBF kernel: $k(x_i, x_j) = exp\left(-\frac{||x_i - x_j||^2}{2\sigma}\right)$. Suppose we have three points, z_1, z_2 and x. z_1 is geometrically very close to x and z_2 is geometrically far away from x. What can you say about the value of $k(z_1, x)$ and $k(z_2, x)$? Choose one of the following and justify your choice.

i. $k(z_1, x)$ will be close to 1 and $k(z_2, x)$ will be close to 0.

ii. $k(z_1, x)$ will be close to 0 and $k(z_2, x)$ will be close to 1.

iii. $k(z_1, x)$ will be close to c_1, $c_1 \gg 1$ and $k(z_2, x)$ will be close to c_2, $c_2 \ll 0$, where $c_1, c_2 \in \mathbb{R}$

iv. $k(z_1, x)$ will be close to c_1, $c_1 \ll 0$ and $k(z_2, x)$ will be close to c_2, $c_2 \gg 1$, where $c_1, c_2 \in \mathbb{R}$

1.a	/2	

Correct answer is **i.**, RBF kernel generates a "bump" around the center x. For points z_1 close to the center of the bump, $k(z_1, x)$ will be close to 1, for points z_2 away from the center of the bump $k(z_2, x)$ will be close to 0.

1.b You are given a plot below that illustrates a dataset with two classes (marked by red and blue). In 3 separate plots, draw the decision boundaries when you train an SVM classifier with linear, polynomial (order 2) and RBF kernels respectively. You can assume that the classes have equal number of instances.

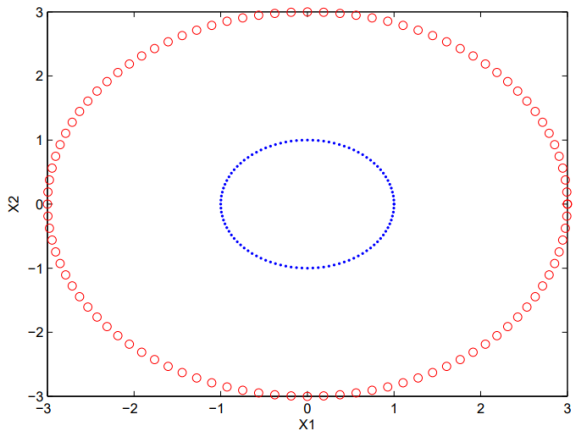
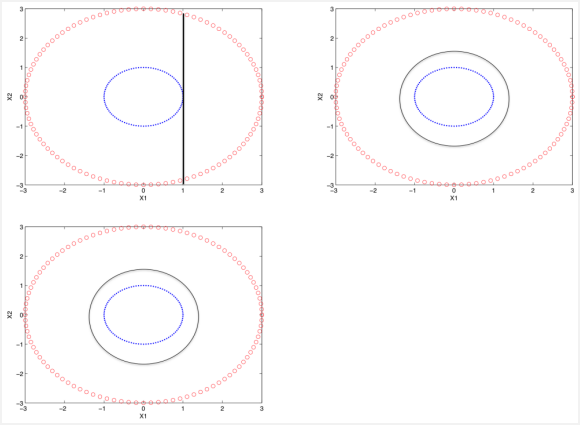


Figure 1: Figure for Question 1.b

1.b	/3	


X1
X2

1.c Support vector machines learn a decision boundary leading to the largest margin from both classes. Suppose we are training a hard-margin SVM on a tiny dataset with just 4 points as shown in the figure below. This dataset consists of two examples with class label -1 (denoted with plus), and two examples with class label +1 (denoted with triangles).

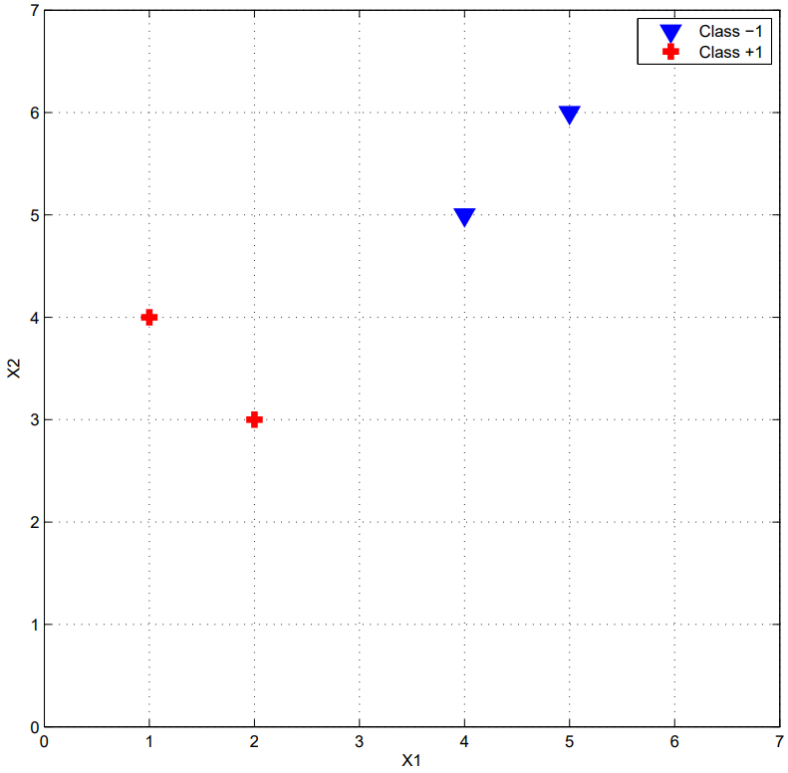


Figure 2: Figure for Question 1.c

Find the weight vector $\mathbf{w}$ and bias b. What's the equation corresponding to the decision boundary?

1.c	/4	

SVM tries to maximize the margin between two classes. Therefore, the optimal decision boundary is diagonal and it crosses the point $(3, 4)$. It is perpendicular to the line between support vectors $(4, 5)$ and $(2, 3)$, hence it is slope is $\mathrm{m} = -1$. Thus the line equation is $(x_2 - 4) = -1\,(x_1 - 3) = x_1 + x_2 = 7$. From this equation, we can deduce that the weight vector has to be of the form (w_1, w_2), where $w_1 = w_2$. It also has to satisfy the following equations:

$$2w_1 + 3w_2 + b = 1 \text{ and}$$
$$4w_1 + 5w_2 + b = -1$$

Hence $w_1 = w_2 = -1/2$ and $b = 7/2$

2. This question consists of different parts from different topics taught in the class.

2.a Consider the following neural network.

$$\begin{aligned} \text{Linear}_1(\boldsymbol{x}) &= W_1^\top \boldsymbol{x} \\ \text{Linear}_2(\boldsymbol{x}) &= W_2^\top \boldsymbol{x} \\ \text{Threshold}(\cdot) &= \max[b, \cdot] \end{aligned} \tag{1}$$

We predict the response $y \in \mathbb{R}$ from the features $\boldsymbol{x} \in \mathbb{R}^d$ as follows:

$$y = \text{Linear}_1(\text{Threshold}(\text{Linear}_2(\text{Linear}_2(\boldsymbol{x})))) \tag{2}$$

Find the total number of trainable parameters.

2.a /3

Notice that Linear_2 is applied twice in succession, that means its input and output vectors have the same dimension. Thus $W_2 \in \mathbb{R}^{d \times d}$. Also, the following threshold function has a single parameter b. Since we have the information that $y \in \mathbb{R}$, we know from the final Linear_1 application that $W_1 \in \mathbb{R}^{d \times 1}$. Thus the number of total trainable parameters are: $d^2 + d + 1$

2.b Design a Feed Forward Network that takes a real valued 1-dimensional input x and produces real-valued output, given by the following function:

$$f(x) = \begin{cases} -2022 & x < -2022 \\ x & -2022 \leq x < 2022 \\ 2022 & x \geq 2022 \end{cases}$$

Assume the hidden layers have ReLU activations (Note: $\text{ReLU}(\cdot) = \max[0, \cdot]$). Specify the input layer, number of hidden layers, weights/biases for each layer and output layer of your network.

2.b /3

One possible way is as follows: one input node, one hidden layer with two hidden nodes, and one output node.
The input-to-hidden node weights are $w_1 = +1$ and $w_2 = +1$, and the hidden-to-output node weights are $w_3 = +1$ and $w_4 = -1$. Hidden nodes have biases $b_1 = +2022$ and $b_2 = -2022$. Output node bias is $b_{\text{output}} = -2022$.

2.c Under the same assumptions as above, i.e. all hidden layers have ReLU activations, design a Feed Forward network, which takes as input two boolean-valued inputs x_1 and x_2, and produces the output $x_1 \oplus x_2$.

Specify the input layer, number of hidden layers, weights/biases for each layer and output layer of your network.

x_1	x_2	$x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	0

2.c /3

XOR(x,y) = relu (2x+2y) - relu(2x+2y-1) - relu(2x+2y-3)
Note: Other valid proposals possible

2.d The figure below plots the number of minibatch gradient descent iterations required to reach a given loss, as a function of the batch size. Use the plot to answer the following questions:

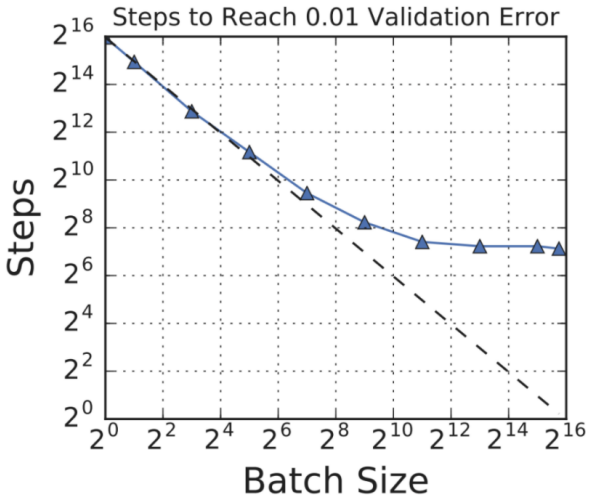


Figure 3: Figure for Question 2.d

i. It is observed that for small batch sizes, the number of iterations required to reach the target loss decreases as the batch size increases. Explain this phenomenon.

ii. It is also observed that for large batch sizes, the number of required iterations does not change much as the batch size is increased. Why is that?

2.d /2

i. Larger batch sizes reduces the variance in the gradient estimation of SGD. Hence, larger batch converges faster.
Mention of variance, low accuracy of gradient estimate given by smaller batch-sizes, or noise being decreased is sufficient to get full marks for this question.
ii. As the batch size grows larger, SGD effectively becomes full batch gradient descent, so not much noise in gradient estimate.

Total: 20