Wensong Hu
EECS
University of Michigan

March 5th, 2024

HW3 — Kernel, SVM, and Neural Network

1 Directly Construct Valid Kernels

(a) $k(x, z) = k_1(x, z) + k_2(x, z)$

Answer: This is valid kernel.
Proof: Assume k_1 and k_2 are valid kernels, thus they are positive semi-definite (PSD). For any set of points $\{x^{(1)}, x^{(2)}, ..., x^{(N)}\}$, we have:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} k(x^{(i)}, x^{(j)}) = \sum_{i=1}^{N}\sum_{j=1}^{N}(k_1(x^{(i)}, x^{(j)}) + k_2(x^{(i)}, x^{(j)})) \tag{1-1}$$

Since both K_1 and K_2 are PSD, the sum is also PSD. Hence, k is a valid kernel.

(b) $k(x, z) = k_1(x, z) - k_2(x, z)$

Answer: This is not valid kernel.
Proof: Without loss of generality, let us assume k_1 and k_2 are identity kernels, thus $k_1(x, z) = k_2(x, z) = x \cdot z$. Now consider $k(x, z)$:

$$k(x, z) = k_1(x, z) - k_2(x, z) = x \cdot z - x \cdot z = 0 \tag{1-2}$$

This function is a valid kernel as it is the zero kernel. However, if $k_2(x, z) > k_1(x, z)$ for some x, z, then $k(x, z)$ would be negative and k is not PSD. Thus, k is not guaranteed to be a valid kernel.

(c) $k(x, z) = -ak_1(x, z)$

Answer: This is not a valid kernel.
Proof: Given a valid kernel k_1 and a positive real number a, the function $k(x, z) = -ak_1(x, z)$ will not be PSD because:

$$\sum_{i=1}^{N}\sum_{j=1}^{N}(-ak_1(x^{(i)}, x^{(j)})) = -a\sum_{i=1}^{N}\sum_{j=1}^{N} k_1(x^{(i)}, x^{(j)}) \tag{1-3}$$

Since K_1 is PSD, the sum on the right is non-negative, and thus the overall sum is non-positive, which violates the PSD condition.

(d) $k(x, z) = f(x)f(z)$

Answer: This is a valid kernel.
Proof: For a real-valued function f, the kernel $k(x, z) = f(x)f(z)$ is given by:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} f(x^{(i)})f(x^{(j)}) = \left(\sum_{i=1}^{N} f(x^{(i)})\right)^2 \tag{1-4}$$

The right-hand side is a square of a real number, hence it is non-negative. Therefore, the kernel matrix K is PSD, and k is a valid kernel.

(e) $k(x, z) = k_3(\phi(x), \phi(z))$

Assuming k_3 is a valid kernel over $\mathbb{R}^M \times \mathbb{R}^M$ and ϕ is a feature mapping from $\mathbb{R}^D$ to $\mathbb{R}^M$, the composed function $k(x, z)$ is given by:

$$k(x, z) = k_3(\phi(x), \phi(z)) \tag{1-5}$$

By the assumption that k_3 is positive semi-definite (PSD), for any set of points $\{x^{(1)}, ..., x^{(N)}\}$ in $\mathbb{R}^D$, the kernel matrix K with entries $K_{ij} = k_3(\phi(x^{(i)}), \phi(x^{(j)}))$ must satisfy:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} K_{ij} \geq 0 \tag{1-6}$$

Therefore, K is PSD and thus a valid kernel.

(f) $k(x, z) = p(k_1(x, z))$

Answer: This is a valid kernel.
Proof: Let p be a polynomial with non-negative coefficients and k_1 be a valid kernel. The function $k(x, z)$ is defined as:

$$k(x, z) = p(k_1(x, z)) \tag{1-7}$$

Since any polynomial with non-negative coefficients can be expressed as a sum of squares, the kernel matrix K derived from k will be PSD if k_1 is PSD. Hence, k is a valid kernel.

(g) $k(x, z) = k_1(x, z)k_2(x, z)$

Answer: This is a valid kernel.
Proof: Assume k_1 and k_2 are valid kernels, thus PSD. For any set of points $\{x^{(1)}, ..., x^{(N)}\}$ in $\mathbb{R}^D$ and coefficients $\{c_1, ..., c_N\}$, the function $k(x, z)$ is given by:

$$k(x, z) = k_1(x, z)k_2(x, z) \tag{1-8}$$

By the Schur product theorem, the Hadamard (element-wise) product of two PSD matrices is also PSD. Therefore, the kernel matrix K constructed from k is PSD, and hence k is a valid kernel.

(h) Feature Map for the Kernel Function

Given the kernel function $k(x, z) = (x^T z + 1)^2$, we need to find a feature map ϕ associated with $k(x, z)$ such that $k(x, z) = \phi(x)^T \phi(z)$. Assuming $D = 2$, let $x = (x_1, x_2)$ and $z = (z_1, z_2)$. We expand $k(x, z)$ as follows:

$$\begin{aligned} k(x, z) &= (x^T z + 1)^2 && (1\text{-}9) \\ &= (x_1 z_1 + x_2 z_2 + 1)^2 && (1\text{-}10) \\ &= x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2 + 2x_1 z_1 + 2x_2 z_2 + 1 && (1\text{-}11) \end{aligned}$$

We can now define the feature map $\phi(x)$ as:

$$\phi(x) = (x_1^2, \sqrt{2}x_1 x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1) \tag{1-12}$$

Thus, we have:

$$k(x, z) = \phi(x)^T \phi(z) \tag{1-13}$$

(i) Gaussian Kernel as Infinite-Dimensional Inner Product

The Gaussian Kernel is defined as:

$$k(x, z) = \exp\left(-\frac{\|x - z\|^2}{2\sigma^2}\right) \tag{1-14}$$

We start by expanding the squared norm using Hint 1:

$$\|x - z\|^2 = x^T x + z^T z - 2x^T z \tag{1-15}$$

The kernel can be rewritten using the power series expansion for the exponential function, given by Hint 2:

$$\exp(x) = \sum_{n=0}^{\infty} \frac{x^n}{n!} \tag{1-16}$$

Applying this to the Gaussian Kernel, we have:

$$k(x, z) = \exp\left(-\frac{x^T x}{2\sigma^2}\right) \exp\left(-\frac{z^T z}{2\sigma^2}\right) \sum_{n=0}^{\infty} \frac{1}{n!} \left(\frac{x^T z}{\sigma^2}\right)^n \tag{1-17}$$

Let's define a feature map ϕ for the vector x with respect to the kernel function. Based on Hint 3, each element of $\phi(x)$ can be expressed as an explicit formula without limits or infinite summations. We can consider the Taylor series expansion for each term in the series:

$$\phi(x) = \left[\ldots, \frac{x_i^n}{\sqrt{n!}\sigma^n}, \frac{\sqrt{2^n} x_i^n x_j^n}{n!\sigma^{2n}}, \ldots\right] \tag{1-18}$$

where each element corresponds to a term in the expansion of the Gaussian Kernel. Here i and j are indices over the dimensions of x, and n ranges from 0 to infinity.

This represents an infinite-dimensional feature map where each dimension corresponds to a different monomial of x and z scaled appropriately by powers of σ and factorial terms. The inner product $\phi(x)^T \phi(z)$ gives us the Gaussian Kernel $k(x, z)$.

2 Implementing Soft Margin SVM by Optimizing Primal Objective

2.1 Find derivatives

Take derivatives about loss function with respect to w first:

$$\nabla_w E(w,b) = \frac{\partial}{\partial w}\frac{1}{2}||w||^2 + C\sum_{i=1}^{N} max(0, 1 - y^{(i)}(w^T\phi(x^{(i)}) + b)) \tag{1-19}$$

$$= w + C\sum_{i=1}^{N} \frac{\partial}{\partial w} max(0, 1 - y^{(i)}(w^T\phi(x^{(i)}) + b)) \tag{1-20}$$

In order to take derivatives about the $max()$, we using indicator function:

$$\frac{\partial}{\partial w} max(0, 1 - y^{(i)}(w^T\phi(x^{(i)}) + b)) = \begin{cases} y^{(i)}\phi(x^{(i)}) & 1 - y^{(i)}(w^T\phi(x^{(i)}) + b) > 0 \\ 0 & otherwise \end{cases} \tag{1-21}$$

$$= -\mathbb{1}[y^{(i)}(w^T\phi(x^{(i)}) + b) < 1]y^{(i)}\phi(x^{(i)}) \tag{1-22}$$

Bring above equation back:

$$\nabla_w E(w,b) = w - C\sum_{i=1}^{N} \mathbb{1}[y^{(i)}(w^T\phi(x^{(i)}) + b) < 1]y^{(i)}\phi(x^{(i)}) \tag{1-23}$$

Derivative w.r.t w is proved.

Take derivatives about loss function w.r.t. b:

$$\nabla_b E(w,b) = \frac{\partial}{\partial w}\frac{1}{2}||w||^2 + C\sum_{i=1}^{N} max(0, 1 - y^{(i)}(w^T\phi(x^{(i)}) + b)) \tag{1-24}$$

$$= 0 + C\sum_{i=1}^{N} \frac{\partial}{\partial b} max(0, 1 - y^{(i)}(w^T\phi(x^{(i)}) + b)) \tag{1-25}$$

where:

$$\frac{\partial}{\partial b} max(0, 1 - y^{(i)}(w^T\phi(x^{(i)}) + b)) = \begin{cases} y^{(i)} & 1 - y^{(i)}(w^T\phi(x^{(i)}) + b) > 0 \\ 0 & otherwise \end{cases} \tag{1-26}$$

$$= -\mathbb{1}[y^{(i)}(w^T\phi(x^{(i)}) + b) < 1]y^{(i)} \tag{1-27}$$

Bring above equation back:

$$\nabla_w E(w,b) = -C\sum_{i=1}^{N} \mathbb{1}[y^{(i)}(w^T\phi(x^{(i)}) + b) < 1]y^{(i)} \tag{1-28}$$

Derivative w.r.t b is proved.

2.2 Code Implementation

Submitted to Autograder

2.3 Discussion

```
>>> [NumEpochs: 1] Accuracy: 54.17%
    b: [0.01], W: [[ 0.224  -0.0855  0.545   0.206 ]]
>>> [NumEpochs: 3] Accuracy: 54.17%
    b: [0.02], W: [[ 0.44848122 -0.17019759  1.08918105  0.41163421]]
>>> [NumEpochs: 10] Accuracy: 95.83%
    b: [-0.14], W: [[-0.1648026  -0.80606447  1.37816462  0.57445096]]
>>> [NumEpochs: 30] Accuracy: 95.83%
    b: [-0.175], W: [[-0.20885861 -0.69979483  1.30489009  0.56605151]]
>>> [NumEpochs: 100] Accuracy: 95.83%
    b: [-0.315], W: [[-0.28240917 -0.77529188  1.75856715  0.82441652]]
```

3 Asymmetric Cost SVM

3.1 Lagrangian

Given the optimization problem:

$$\begin{aligned} \underset{w,b,\xi}{\text{minimize}} \quad & \frac{1}{2}w^T w + C_0 \sum_{y^{(i)}=-1} \xi^{(i)} + C_1 \sum_{y^{(i)}=1} \xi^{(i)} \\ \text{subject to} \quad & y^{(i)}(w^T \phi(x^{(i)}) + b) \geq 1 - \xi^{(i)},\ i = 1, \ldots, N, \\ & \xi^{(i)} \geq 0,\ i = 1, \ldots, N. \end{aligned} \tag{1-29}$$

We define the Lagrangian as:

$$L(w, b, \xi, \alpha, \mu) = \frac{1}{2}w^T w + C_0 \sum_{y^{(i)}=-1} \xi^{(i)} + C_1 \sum_{y^{(i)}=1} \xi^{(i)} - \sum_{i=1}^{N} \alpha^{(i)}[y^{(i)}(w^T \phi(x^{(i)}) + b) - 1 + \xi^{(i)}] - \sum_{i=1}^{N} \mu^{(i)} \xi^{(i)}. \tag{1-30}$$

3.2 Derivative w.r.t primal variables

The derivatives of the Lagrangian with respect to the primal variables are:

$$\nabla_w L = w - \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} \phi(x^{(i)}), \tag{1-31}$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{N} \alpha^{(i)} y^{(i)}, \tag{1-32}$$

$$\frac{\partial L}{\partial \xi^{(i)}} = C_0 \mathbb{1}[y^{(i)} = -1] + C_1 \mathbb{1}[y^{(i)} = 1] - \alpha^{(i)} - \mu^{(i)} \tag{1-33}$$

3.3 Dual problem

Setting these derivatives to zero, we get the conditions for optimality:

$$w = \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} \phi(x^{(i)}), \tag{1-34}$$

$$0 = \sum_{i=1}^{N} \alpha^{(i)} y^{(i)}, \tag{1-35}$$

$$\alpha^{(i)} = C_0 \mathbb{1}[y^{(i)} = -1] + C_1 \mathbb{1}[y^{(i)} = 1] - \mu^{(i)} \tag{1-36}$$

The dual problem is then formulated as:

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & W(\alpha) = \sum_{i=1}^{N} \alpha^{(i)} - \frac{1}{2} \sum_{i,j=1}^{N} \alpha^{(i)} \alpha^{(j)} y^{(i)} y^{(j)} \phi(x^{(i)})^T \phi(x^{(j)}) \\ \text{subject to} \quad & 0 \leq \alpha^{(i)} \leq C_0 \text{ for } y^{(i)} = -1, \\ & 0 \leq \alpha^{(i)} \leq C_1 \text{ for } y^{(i)} = 1, \\ & \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} = 0 \end{aligned} \tag{1-37}$$

4 SVMs with Convex Optimization

4.1 Convert dual problem to quadratic form

Given the dual optimization problem:

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & \sum_{n=1}^{N} \alpha^{(n)} - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha^{(n)} \alpha^{(m)} y^{(n)} y^{(m)} k(x^{(n)}, x^{(m)}) \\ \text{subject to} \quad & 0 \leq \alpha^{(n)} \leq C, \\ & \sum_{n=1}^{N} \alpha^{(n)} y^{(n)} = 0, \end{aligned} \tag{1-38}$$

where $y^{(n)} \in \{-1, 1\}$, $x^{(n)} \in \mathbb{R}^D$, and $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ is a kernel function, and $\alpha \in \mathbb{R}^N$
The CVXOPT library solves the following problem:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & \frac{1}{2} v^T P v + q^T v \\ \text{subject to} \quad & Gv \leq h, \\ & Av = b. \end{aligned} \tag{1-39}$$

The equivalence of the two problems can be shown by setting:

- $v \in \mathbb{R}^{N \times 1}, v = \alpha$

- $P \in \mathbb{R}^{N\times N}, P_{nm} = y^{(n)}y^{(m)}k(x^{(n)}, x^{(m)})$
- $G \in \mathbb{R}^{2N\times N}, G = \begin{bmatrix} -I \\ I \end{bmatrix}$
- $A \in \mathbb{R}^{1\times N}, A = y^T$
- $q \in \mathbb{R}^{N\times 1}, q = [-1, \ldots, -1]^T$
- $h \in \mathbb{R}^{2N\times 1}, h = [0, \ldots, 0, C, \ldots, C]^T$
- $b \in \mathbb{R}^1, b = 0$

4.2 Code Implementation

Submitted to Autograder.

4.3 Discussion

CVXOPTSVM performance

	linear	poly	sigmoid	rbf
dataset 0	90.00	87.50	87.50	95.00
dataset 1	40.00	40.00	40.00	90.00
dataset 2	95.00	95.00	95.00	95.00
dataset 3	57.50	55.00	60.00	97.50

Figure 1: SVM Accuracy on Several Datasets

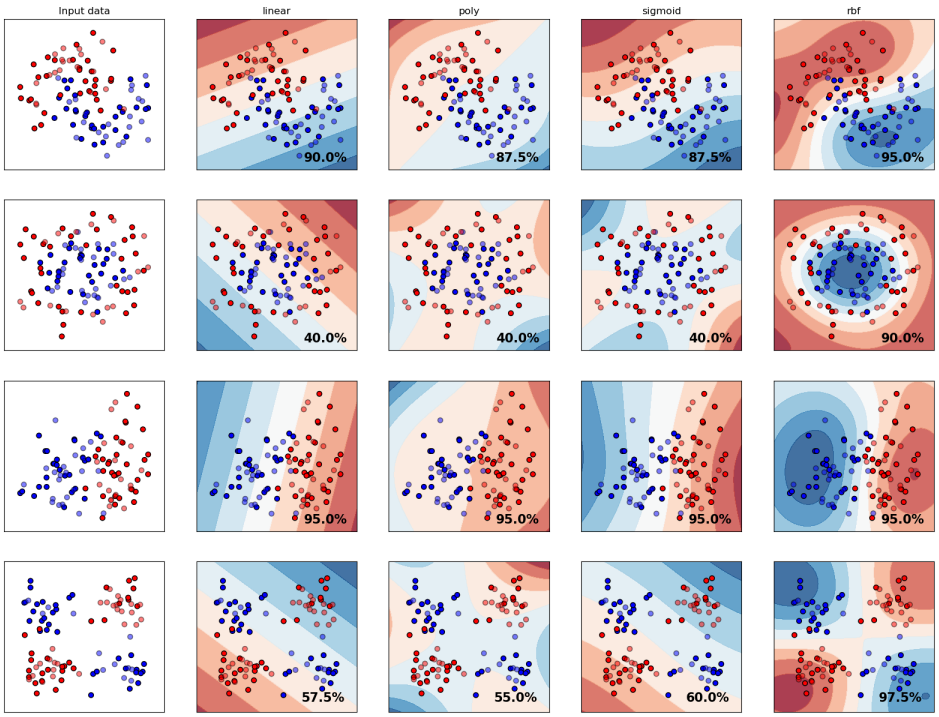

Figure 2: Decision Boundary and Accuracy

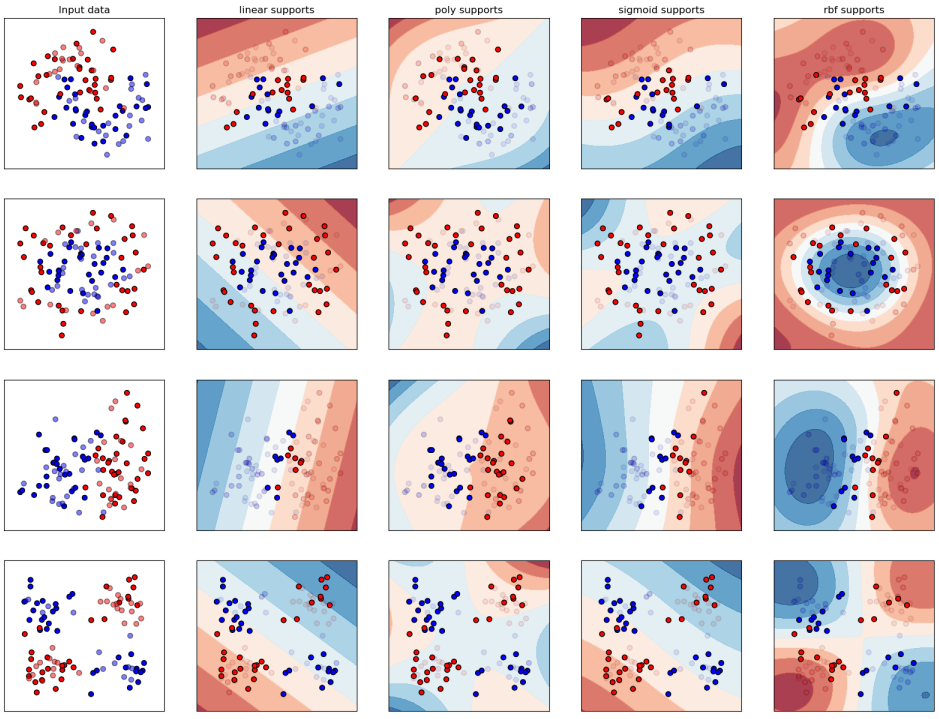

Figure 3: Decision Boundary and Support Vectors

5 Neural Network Layer Implementation

5.1 Fully connected layer partial derivatives

$$\frac{\partial L}{\partial W} = \frac{\partial Y}{\partial W}\frac{\partial L}{\partial Y} \tag{1-40}$$
$$= X^T \nabla_Y L \tag{1-41}$$

$$\frac{\partial L}{\partial b} = \frac{\partial Y}{\partial b}\frac{\partial L}{\partial Y} \tag{1-42}$$
$$= [1, \ldots, 1]\nabla_Y L \tag{1-43}$$

where, $[1, \ldots, 1] \in \mathbb{R}^{1\times N}$

$$\frac{\partial L}{\partial X} = \frac{\partial Y}{\partial X}\frac{\partial L}{\partial Y} \tag{1-44}$$
$$= (\nabla_Y L)W^T \tag{1-45}$$

5.2 ReLU layer

$$\frac{\partial L}{\partial X} = \frac{\partial Y}{\partial X}\frac{\partial L}{\partial Y} \tag{1-46}$$
$$= (\nabla_X Y) * (\nabla_Y L) \tag{1-47}$$

where, $(\nabla_X Y)_{i,j} = \mathbb{1}[X_{i,j} >= 0]$

5.3 Code Implementation

Code Submitted to Autograder.

5.4 Training Loss and Train/test Accuracy

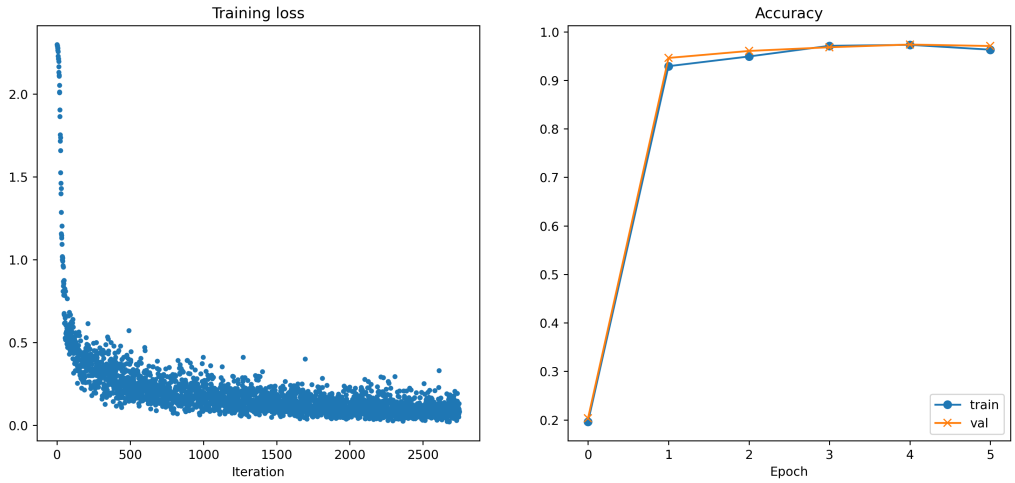

Figure 4: Training Loss and Train/test Accuracy

5.5 Test Accuracy on MNIST

Test accuracy: 96.31%

Submitted by Wensong Hu on March 5th, 2024.