# EECS545 Machine Learning
# Solution for Homework #3

## 1 [21 + 3points] Direct Construction of Valid Kernels

Notes:

- We will use $G_i$ to denote the (Gram) matrix corresponding to the kernel function $k_i(..,.)$
- For the ones that are not kernels, it is sufficient to come up with one counter example (we only list one but there could be others). In addition, it is possible to prove of validity of all kernels by proving $\forall \mathbf{u}, \mathbf{u}^\top G\mathbf{u} \geq 0$ by expanding $\mathbf{u}^\top G\mathbf{u} = \sum_{i=1}^N \sum_{j=1}^N u_i G_{ik} u_k = \sum_{i=1}^N \sum_{j=1}^N u_i k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) u_k \geq 0$. However, it is possible to take shortcuts by knowing some properties of Positive Semidefinite (PSD) Matrices. In particular, PSD + PSD = PSD, and, PSD * non-negative real number = PSD.

(a) **Answer [2 points]:** Kernel. The sum of 2 positive semidefinite matrices is a positive semidefinite matrix: $\forall \mathbf{u}, \mathbf{u}^\top G_1\mathbf{u} \geq 0, \mathbf{u}^\top G_2\mathbf{u} \geq 0$ since $k_1, k_2$ are kernels. This implies $\forall \mathbf{u}, \mathbf{u}^\top G\mathbf{u} = \mathbf{u}^\top G_1\mathbf{u} + \mathbf{u}^\top G_2\mathbf{u} \geq 0$

(b) **Answer [2 points]:** Not a kernel. Counterexample: let $k_2(\cdot,\cdot) = 2k_1(\cdot,\cdot)$ (Multiplying a PSD by a non-negative integer produces a PSD. $\forall \mathbf{u}, \mathbf{u}^\top G_1\mathbf{u} \geq 0$, which implies $\forall \mathbf{u}$, $a\mathbf{u}^\top G_1\mathbf{u} \geq 0$). Then we have $\forall \mathbf{u}, \mathbf{u}^\top G\mathbf{u} = \mathbf{u}^\top (G_1 - 2G_1)\mathbf{u} = -\mathbf{u}^\top G_1\mathbf{u} \leq 0$

(c) **Answer [2 points]:** Not a kernel. Counterexample: $a = 1$. Then we have $\forall \mathbf{u}, -\mathbf{u}^\top G_1\mathbf{u} \leq 0$

(d) **Answer [2 points]:** Kernel. Just let $\psi(\mathbf{x}) = f(\mathbf{x})$, and since $f(\mathbf{x})$ is a scalar, we have $k(\mathbf{x}, \mathbf{z}) = \psi(\mathbf{x})^\top \psi(\mathbf{z})$ and we are done.

(e) **Answer [3 points]:** Kernel. since $k_3$ is a kernel, the matrix $G_3$ obtained for any finite set $\left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\right\}$ is positive semidefinite, and so it is also positive semidefinite for the sets $\left\{\phi\left(\mathbf{x}^{(1)}\right), \ldots, \phi\left(\mathbf{x}^{(N)}\right)\right\}^2$

Alternatively, let $k_3(\mathbf{x}, \mathbf{z}) = \psi(\mathbf{x})^\top \psi(\mathbf{z})$ so that $k(\mathbf{x}, \mathbf{z}) = \psi(\phi(\mathbf{x}))^\top \psi(\phi(\mathbf{z})) = \left((\psi \odot \phi)(\mathbf{x})\right)^\top \left((\psi \odot \phi)(\mathbf{z})\right)$

(f) **Answer [3 points]:** Kernel. By combining (1a) sum, (see below or 1b answer) scalar product, (1d) powers, constant term, we see that any polynomial of a kernel $k_1$ will again be a kernel.

Multiplying a PSD by a non-negative integer produces a PSD. $\forall \mathbf{u}, \mathbf{u}^\top G_1\mathbf{u} \geq 0$, which implies $\forall \mathbf{u}$, $a\mathbf{u}^\top G_1\mathbf{u} \geq 0$

(g) **Answer [4 points]:** Kernel. $k_1$ is a kernel, thus $\exists \phi^{(1)}, k_1(\mathbf{x}, \mathbf{z}) = \phi^{(1)}(\mathbf{x})^\top \phi^{(1)}(\mathbf{z}) = \sum_i \phi_i^{(1)}(\mathbf{x})\phi_i^{(1)}(\mathbf{z})$.

Similarly, $k_2$ is a kernel, thus $\exists \phi^{(2)}, k_2(\mathbf{x}, \mathbf{z}) = \phi^{(2)}(\mathbf{x})^\top \phi^{(2)}(\mathbf{z}) = \sum_i \phi_i^{(2)}(\mathbf{x}) \phi_i^{(2)}(\mathbf{z})$

$$\begin{aligned}
k(\mathbf{x}, \mathbf{z}) &= k_1(\mathbf{x}, \mathbf{z}) k_2(\mathbf{x}, \mathbf{z}) \\
&= \sum_i \phi_i^{(1)}(\mathbf{x}) \phi_i^{(1)}(\mathbf{z}) \sum_j \phi_j^{(2)}(\mathbf{x}) \phi_j^{(2)}(\mathbf{z}) \\
&= \sum_i \sum_j \phi_i^{(1)}(\mathbf{x}) \phi_i^{(1)}(\mathbf{z}) \phi_j^{(2)}(\mathbf{x}) \phi_j^{(2)}(\mathbf{z}) \\
&= \sum_i \sum_j \left( \phi_i^{(1)}(\mathbf{x}) \phi_j^{(2)}(\mathbf{x}) \right) \left( \phi_i^{(1)}(\mathbf{z}) \phi_j^{(2)}(\mathbf{z}) \right) \\
&= \sum_{(i,j)} \psi_{i,j}(\mathbf{x}) \psi_{i,j}(\mathbf{z})
\end{aligned}$$

Where the last equality holds because we can define $\psi_{i,j}(\mathbf{x}) = \phi_i^{(1)}(\mathbf{x}) \phi_j^{(2)}(\mathbf{x})$. Therefore, we can define $\psi(\mathbf{x})$ to be a vector containing all $\psi_{i,j}(\mathbf{x})$ for all possible pairs $(i, j)$. And we can rewrite:

$$k(\mathbf{x}, \mathbf{z}) = \psi(\mathbf{x})^\top \psi(\mathbf{z})$$

since $k$ can be written in this form, it is a kernel.

(h) **Answer [3 points]:** $\phi(\mathbf{x}) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2} x_1 x_2 \\ \sqrt{2} x_1 \\ \sqrt{2} x_2 \\ 1 \end{bmatrix}$

(Note: we set D=2. $x^2$ could be considered as scalar $x^2 = x^\top x$ unless explicitly mention/define elementwise multiplication.)

(i) **Answer [3 points extra credit]:**
Note 1: There are a lot of failure cases (wrong proofs) that can happen in this problem (especially when it comes to the infinite sum).
Note 2: Applying (a), (f), (g) infinite times is NOT a valid proof.

$$\begin{aligned}
k(\mathbf{x}, \mathbf{z}) &= \exp\left( -\frac{\|\mathbf{x} - \mathbf{z}\|^2}{2\sigma^2} \right) \\
&= \exp\left( -\frac{\|\mathbf{x}\|^2}{2\sigma^2} \right) \cdot \exp\left( -\frac{\|\mathbf{z}\|^2}{2\sigma^2} \right) \cdot \exp\left( \frac{\mathbf{x}^\top \mathbf{z}}{\sigma^2} \right) \\
&= \exp\left( -\frac{\|\mathbf{x}\|^2}{2\sigma^2} \right) \cdot \exp\left( -\frac{\|\mathbf{z}\|^2}{2\sigma^2} \right) \cdot \sum_{n=0}^{\infty} \frac{1}{n!} \left( \frac{\mathbf{x}^\top \mathbf{z}}{\sigma^2} \right)^n
\end{aligned}$$

(Note: Applying (a) through an infinite sum is invalid; fails to make an explicit construction of an inf-dim feature.)

Note that

$$\begin{aligned}
\frac{\left(\mathbf{x}^\top \mathbf{z}\right)^n}{n!\sigma^{2n}} &= \frac{1}{n!\sigma^{2n}} \sum_{k_1+\cdots+k_d=n} \binom{n}{k_1, \cdots, k_d} \Pi_{i=1}^d \left(x_i z_i\right)^{k_i} \\
&= \left( \frac{\sqrt{\binom{n}{k_1^{(1)}, \cdots, k_d^{(1)}}}}{\sigma^n \sqrt{n!}} \prod_{i=1}^d x_i^{k_i^{(1)}}, \cdots, \frac{\sqrt{\binom{n}{k_1^{(m_n)}, \cdots, k_d^{(m_n)}}}}{\sigma^n \sqrt{n!}} \prod_{i=1}^d x_i^{k_i^{(m_n)}} \right) \\
&\cdot \left( \frac{\sqrt{\binom{n}{k_1^{(1)}, \cdots, k_d^{(1)}}}}{\sigma^n \sqrt{n!}} \prod_{i=1}^d z_i^{k_i^{(1)}}, \cdots, \frac{\sqrt{\binom{n}{k_1^{(m_n)}, \cdots, k_d^{(m_n)}}}}{\sigma^n \sqrt{n!}} \prod_{i=1}^d z_i^{k_i^{(m_n)}} \right) \\
&= \tilde{\phi}_n(\mathbf{x})^\top \tilde{\phi}_n(\mathbf{z}).
\end{aligned}$$

where, for a given $n$, $\tilde{\phi}_n(\mathbf{x})$ denotes the finite-dimensional vector that consists of all the monomials of degree $n$ (See Lecture 8, p.16) for $\mathbf{x} = (x_1, \cdots, x_d)$, (with some proper coefficients), whose length is $m_n$.

(Note: The answer should handle factorization of the infinite sum properly. If not, it is easy to be valid only when $d = 1$.)

And we have

$$\begin{aligned}
k(\mathbf{x}, \mathbf{z}) &= \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right) \cdot \exp\left(-\frac{\|\mathbf{z}\|^2}{2\sigma^2}\right) \cdot \sum_{n=0}^{\infty} \frac{1}{n!} \left(\frac{\mathbf{x}^\top \mathbf{z}}{\sigma^2}\right)^n \\
&= \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right) \cdot \exp\left(-\frac{\|\mathbf{z}\|^2}{2\sigma^2}\right) \cdot \sum_{n=0}^{\infty} (\tilde{\phi}_n(\mathbf{x}))^\top (\tilde{\phi}_n(\mathbf{z})) \\
&= \left(\exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right) \bigcup_{n=0}^{\infty} \tilde{\phi}_n(\mathbf{x})\right)^\top \left(\exp\left(-\frac{\|\mathbf{z}\|^2}{2\sigma^2}\right) \bigcup_{n=0}^{\infty} \tilde{\phi}_n(\mathbf{z})\right) \\
&= \phi(\mathbf{x})^\top \phi(\mathbf{z})
\end{aligned}$$

where $\bigcup_{n=0}^{\infty} \tilde{\phi}_n(\mathbf{x})$ denotes the (infinite) concatenation of all the $\tilde{\phi}_n(\mathbf{x})$ vectors through $n = 0, 1, 2, \cdots$, i.e.,

$$\begin{aligned}
\bigcup_{n=0}^{\infty} \tilde{\phi}_n(\mathbf{x}) = \Bigg( & \underbrace{\cdots, \frac{\sqrt{\binom{1}{k_1, \cdots, k_d}}}{\sigma} \prod_{i=1}^d x_i^{k_i}, \cdots}_{\tilde{\phi}_1(\mathbf{x})}, \quad \underbrace{\cdots, \frac{\sqrt{\binom{2}{k_1, \cdots, k_d}}}{\sigma^2 \sqrt{2!}} \prod_{i=1}^d x_i^{k_i}, \cdots}_{\tilde{\phi}_2(\mathbf{x})}, \\
& \cdots, \quad \underbrace{\cdots, \frac{\sqrt{\binom{n}{k_1, \cdots, k_d}}}{\sigma^n \sqrt{n!}} \prod_{i=1}^d x_i^{k_i}, \cdots}_{\tilde{\phi}_n(\mathbf{x})}, \quad \cdots \Bigg).
\end{aligned}$$

## 2 [17 points] Implementing Soft Margin SVM by Optimizing Primal Objective

(a) **Answer [5 points]:** The max(0, ) operator can be thought of a piece-wise function. Calculating the derivative of a piece-wise function can be done by taking the derivative of each piece. In particular, let $f = \max(0, k)$ then $\frac{\partial f}{\partial z} = \frac{\partial k}{\partial z}$ if $k > 0$, and $\frac{\partial f}{\partial z} = \frac{\partial 0}{\partial z} = 0$ if $0 > k$. We can compactly write $\frac{\partial}{\partial z} \max(0, k) = \mathbb{I}[k > 0] \frac{\partial k}{\partial z}$

Alternatively (which we will use below), we can rewrite $\max(0,\ k) = \mathbb{I}[k > 0]k$. The error function we are trying to minimize is:

$$\begin{aligned} E(\mathbf{w}, b) &= \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \max\left(0, 1 - y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)\right) \\ &= \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{N} \mathbb{I}\left[1 - y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) > 0\right]\left(1 - y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)\right) \\ &= \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{N} \mathbb{I}\left[y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) < 1\right]\left(1 - y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)\right) \end{aligned}$$

Then, it easily follows that:

$$\begin{aligned} \nabla_{\mathbf{w}} E(\mathbf{w}, b) &= \mathbf{w} - C\sum_{i=1}^{N} \mathbb{I}\left[y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) < 1\right] y^{(i)} \mathbf{x}^{(i)} \\ \frac{\partial}{\partial b} E(\mathbf{w}, b) &= -C\sum_{i=1}^{N} \mathbb{I}\left[y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) < 1\right] y^{(i)} \end{aligned}$$

(b) **Answer [7 points]:** Autograder.

(c) **Answer [2 points]:** Both cases are the correct answers.
Parameters and accuracies:

- Iter 1: accuracy = 54.17%
  weight=[ 0.224, -0.0855, 0.545, 0.206 ]
  bias=0.01
- Iter 3: accuracy = 54.17%
  weight=[ 0.44848122, -0.17019759, 1.08918105, 0.41163421]
  bias=0.02
- Iter 10: accuracy = 95.83%
  weight=[-0.1648026, -0.80606447, 1.37816462, 0.57445096]
  bias=-0.14
- Iter 30: accuracy = 95.83%
  weight=[-0.20885861, -0.69979483, 1.30489009, 0.56605151]
  bias=-0.175
- Iter 100: accuracy = 95.83%
  weight=[-0.28240917, -0.77529188, 1.75856715, 0.82441652]
  bias=-0.315

# 3 [20 points] Asymmetric Cost SVM

Consider applying an SVM to a supervised learning problem where the cost of a false positive (mistakenly predicting +1 when the label is -1) is different from the cost of a false negative (predicting -1 when the label is +1). The asymmetric cost SVM models these asymmetric costs by posing the following optimization problem:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C_0 \sum_{i:y^{(i)}=-1} \xi^{(i)} + C_1 \sum_{i:y^{(i)}=1} \xi^{(i)}$$

s.t.

$$y^{(i)}(\mathbf{w}^\top\mathbf{x}^{(i)} + b) \geq 1 - \xi^{(i)}, \forall i = 1, \ldots, N$$

$$\xi^{(i)} \geq 0, \forall i = 1, \ldots, N$$

Here, $C_0$ is the cost of a false positive; $C_1$ is the cost of a false negative. (Both $C_0$ and $C_1$ are fixed, known, constants.)

(a) **Answer [4 points]** We will find the dual optimization problem. First, write down the Lagrangian. Use $\alpha^{(i)}$ and $\mu^{(i)}$ to denote the Lagrange multipliers corresponding to the two constraints ($\alpha^{(i)}$ for the first constraint, and $\mu^{(i)}$ for the second constraint (slack variables)) in the primal optimization problem above.

**Answer:** Note that the inequality constraints can be re-written as $1 - \xi^{(i)} - y^{(i)}(\mathbf{w}^\top\mathbf{x}^{(i)} + b) \leq 0$ and $-\xi^{(i)} \leq 0$.

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C_0 \sum_{i:y^{(i)}=-1}^{N} \xi^{(i)} + C_1 \sum_{i:y^{(i)}=1}^{N} \xi^{(i)} - \sum_{i=1}^{N} \alpha^{(i)} \left[ y^{(i)}(\mathbf{w}^\top\mathbf{x}^{(i)} + b) - 1 + \xi_i \right] - \sum_{i=1}^{N} \mu^{(i)}\xi^{(i)}$$

where $\alpha^{(i)} \geq 0,\ \mu^{(i)} \geq 0$ for all $i = 1, \ldots, N$.

(b) **Answer [6 points]** Calculate the following derivatives with respect to the primal variables:

**Answer:**

$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = \mathbf{w} - \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} \mathbf{x}^{(i)}$

$\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu})}{\partial b} = -\sum_{i=1}^{N} \alpha^{(i)} y^{(i)}$

$\nabla_{\xi^{(i)}}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = C^{(i)} - \alpha^{(i)} - \mu^{(i)}$ for $i = 1, \ldots, N$ where, $C^{(i)} = C_0\mathbb{I}[y^{(i)} = -1] + C_1\mathbb{I}[y^{(i)} = 1]$.

(c) **Answer [10 points]** Find the dual optimization problem. You should write down the dual optimization problem in the following form:

**Answer:**

We already know that the minimizer $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$ of the Lagrangian $\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu})$ must make the derivative of the Lagrangian zero.

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = 0 \quad \Longrightarrow \quad \mathbf{w}^* = \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} \mathbf{x}^{(i)}$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu})}{\partial b} \quad \Longrightarrow \quad \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} = 0$$

$$\nabla_{\xi^{(i)}}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = 0 \quad \Longrightarrow \quad C^{(i)} - \alpha^{(i)} - \mu^{(i)} = 0$$

Since $\alpha^{(i)} \geq 0, \mu^{(i)} \geq 0$ and $C^{(i)} - \alpha^{(i)} = \mu^{(i)}$, $0 \leq \alpha^{(i)} \leq C^{(i)}$.

Substitute $\mathbf{w}^*$ from above into the Lagrangian to get the dual function:

$$\begin{aligned}\tilde{\mathcal{L}}(\boldsymbol{\alpha}) = \min_{\mathbf{w},b,\boldsymbol{\xi}} \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) &= \frac{1}{2}\left(\sum_{j=1}^{N} \alpha^{(j)} y^{(j)} \mathbf{x}^{(j)}\right)^\top \left(\sum_{i=1}^{N} \alpha^{(i)} y^{(i)} \mathbf{x}^{(i)}\right) + \sum_{i=1}^{N} \xi^{(i)}(C^{(i)} - \alpha^{(i)} - \mu^{(i)}) \\ &\quad + \sum_{i=1}^{N} \alpha^{(i)} - \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} \left(\sum_{j=1}^{N} \alpha^{(j)} y^{(j)} \mathbf{x}^{(j)}\right)^\top \mathbf{x}^{(i)} - b \sum_{i=1}^{N} \alpha^{(i)} y^{(i)}\end{aligned}$$

As we also got $\sum_i \alpha^{(i)} y^{(i)} = 0$ and $C^{(i)} - \alpha^{(i)} - \mu^{(i)} = 0$, we can simplify the equation as

$$\tilde{\mathcal{L}}(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha^{(i)} - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y^{(j)} y^{(i)} \alpha^{(j)} \alpha^{(i)} (\mathbf{x}^{(j)})^\top \mathbf{x}^{(i)}$$

Therefore the dual problem becomes

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{N} \alpha^{(i)} - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y^{(j)} y^{(i)} \alpha^{(j)} \alpha^{(i)} (\mathbf{x}^{(j)})^\top \mathbf{x}^{(i)}$$

s.t. $0 \le \alpha^{(i)} \le C^{(i)}$ for $i = 1, \ldots, N$; $\sum_{i=1}^{N} \alpha^{(i)} y^{(i)} = 0$; $C^{(i)} = C_0 \mathbb{I}[y^{(i)} = -1] + C_1 \mathbb{I}[y^{(i)} = 1]$

# 4 [20 points] SVMs with CVXOPT

(a) Following the hints, we can rewrite the SVM dual optimization problem as:

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

For reference, we will represent each equation "blob" with a different symbol:

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & \bigstar + \clubsuit \\ \text{subject to} \quad & \blacksquare \\ & \blacktriangle \\ & \heartsuit \end{aligned} \tag{2}$$

Solutions:

(1) We will use $\bigstar$ to solve for $\boldsymbol{P}$. Notice for $\bigstar$ that

$$\boldsymbol{v}^\top \boldsymbol{P} \boldsymbol{v} = \sum_{n}^{N} \sum_{m}^{N} v^{(n)} v^{(m)} P_m^{(n)}$$

So we can simply let (for $m, n = 1 \ldots N$:

$$P_m^{(n)} = y^{(n)} y^{(m)} k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)})$$

(2) We will use ♣ to solve for $\boldsymbol{q}$. For $n = 1 \dots N$, let

$$q_n = -1$$

(3) We will use ■ and ▲ to solve for $\boldsymbol{G}$ and $\boldsymbol{h}$. To do this, we will "stack" two matrices and vectors to for $\boldsymbol{G}$ and $\boldsymbol{h}$. Intuitively, we will stack two identity matrices for $\boldsymbol{G}$, and we will stack to constant vectors for $\boldsymbol{h}$.

$$\boldsymbol{G} = \text{stack}(\text{Identity}(N), -\text{Identity}(N))$$

and for $n = 1 \dots 2N$,

$$h_n = \begin{cases} C & n \leq N \\ 0 & n > N \end{cases}$$

(4) We will use ♡ to solve for $\boldsymbol{A}$ and $\boldsymbol{b}$. Notice that there is only one constraint in ♡. So, $\boldsymbol{b}$ will be a 1-dimensional vector and $\boldsymbol{A}$ will be an $N$ by 1 dimensional matrix.

$$b_1 = 0$$

and for $n = 1 \dots N$,

$$A_n^{(1)} = y^{(n)}$$

(b) Autograder.

(c)

# 5 [24 points] Neural Network Layer Implementation

(a) **[8 points] Fully-Connected Layer**
Consider the inference equation given in the question:

$$\mathbf{Y} = \mathbf{X}\mathbf{W} + \mathbf{B}$$

Furthermore, using the fact that we have the following dimensions of the given variables, $\mathbf{X} \in \mathbb{R}^{N \times D_{in}}$, $\mathbf{W} \in \mathbb{R}^{D_{in} \times D_{out}}$, and $\mathbf{B} \in \mathbb{R}^{N \times D_{out}}$, we can make the sample-wise inference equation as:

$$\mathbf{y}^{(n)} = \mathbf{x}^{(n)}\mathbf{W} + \mathbf{b}$$

In this case, $\mathbf{y}^{(n)} \in \mathbb{R}^{1 \times D_{out}}$, $\mathbf{b} \in \mathbb{R}^{1 \times D_{out}}$ and $\mathbf{x}^{(n)} \in \mathbb{R}^{1 \times D_{in}}$ for $1 \leq n \leq N$. Please note that the inference equation can now be seen in the following matrix multiplication format for an arbitrary value of $n$ where $1 \leq n \leq N$:

$$\begin{bmatrix} Y_1^{(n)} & Y_2^{(n)} & \cdots & Y_{D_{out}}^{(n)} \end{bmatrix} = \begin{bmatrix} X_1^{(n)} & X_2^{(n)} & \cdots & X_{D_{in}}^{(n)} \end{bmatrix} \begin{bmatrix} W_{1,1} & \cdots & W_{1,D_{out}} \\ W_{2,1} & \cdots & W_{2,D_{out}} \\ \vdots & \ddots & \vdots \\ W_{D_{in},1} & \cdots & W_{D_{in},D_{out}} \end{bmatrix} + \begin{bmatrix} b_1 & b_2 & \cdots & b_{D_{out}} \end{bmatrix}$$

Please note here that the notation $X_m^{(n)}$ represents the $m$-th dimension of the $n$-th training example $X_{n,m}$. This can be further simplified to the following summation expression for all $m$ where $1 \leq m \leq D_{out}$:

$$Y_m^{(n)} = \sum_{i=1}^{D_{in}} X_i^{(n)} W_{i,m} + b_m$$

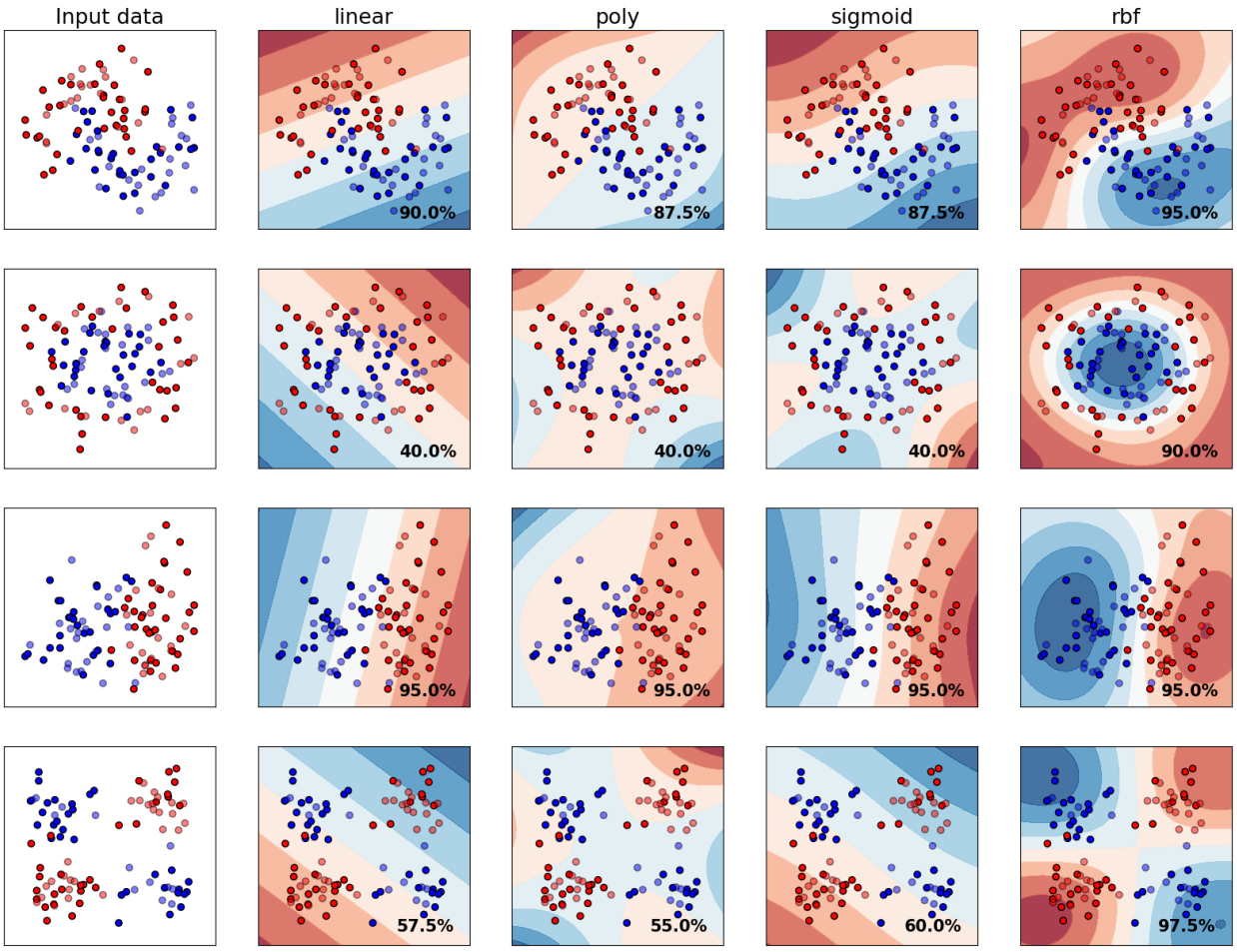



Figure 1: CVXOPT SVM decision boundaries and accuracies.

**[2 points] Gradient $\frac{\partial \mathcal{L}}{\partial \mathbf{W}}$:**

Using the equation given in the hint, we have:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial W_{i,j}} &= \sum_{n=1}^{N} \sum_{m=1}^{D_{out}} \frac{\partial \mathcal{L}}{\partial Y_m^{(n)}} \frac{\partial Y_m^{(n)}}{\partial W_{i,j}} \\
&= \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial Y_j^{(n)}} \frac{\partial Y_j^{(n)}}{\partial W_{i,j}} \\
&= \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial Y_j^{(n)}} X_i^{(n)}
\end{aligned}$$

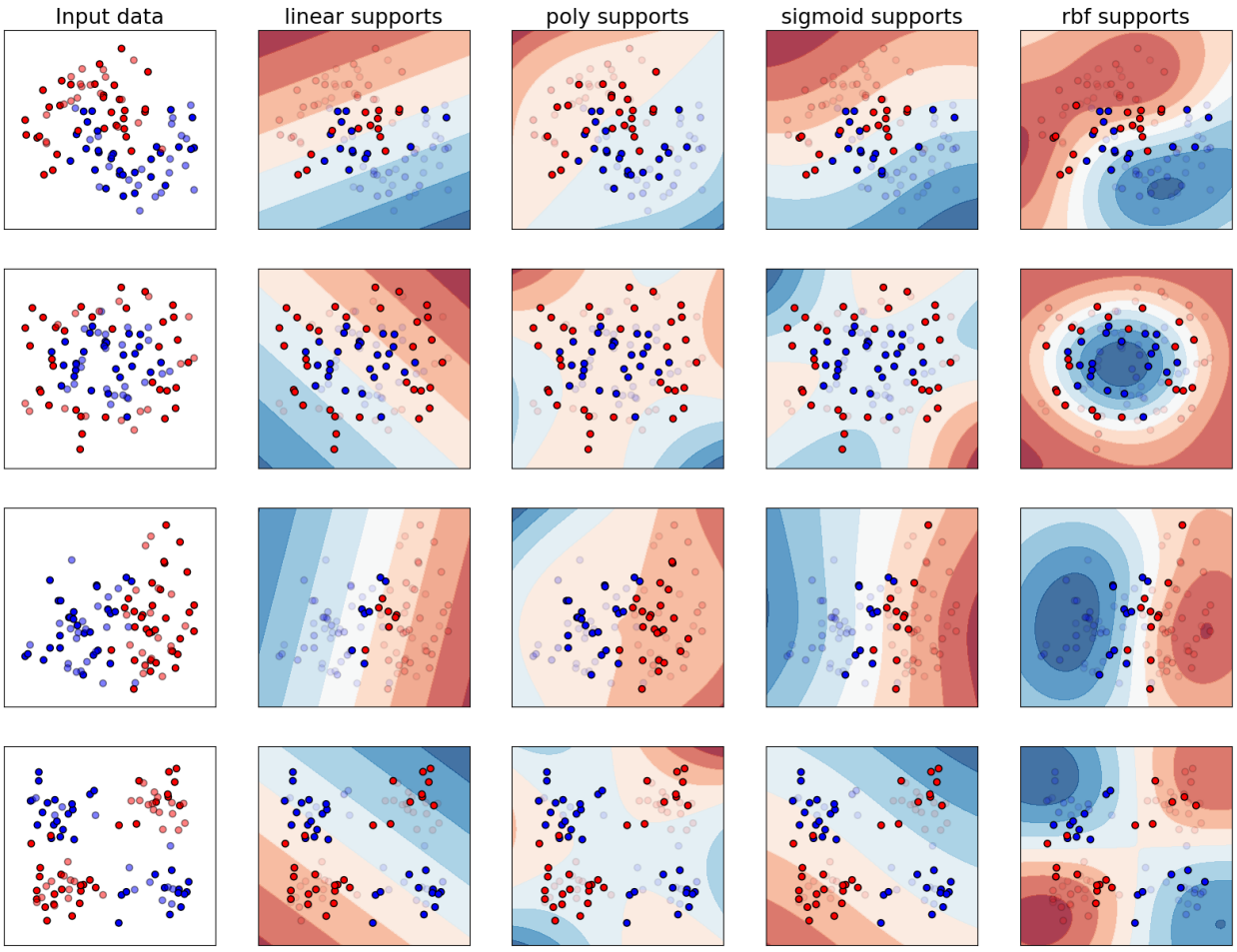



Figure 2: CVXOPT SVM support vectors

With this, consider the following matrix multiplication to understand the vectorized solution:

$$\frac{\partial \mathcal{L}}{\partial W} = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial W_{1,1}} & \frac{\partial \mathcal{L}}{\partial W_{1,2}} & \cdots & \frac{\partial \mathcal{L}}{\partial W_{1,D_{out}}} \\ \frac{\partial \mathcal{L}}{\partial W_{2,1}} & \frac{\partial \mathcal{L}}{\partial W_{2,2}} & \cdots & \frac{\partial \mathcal{L}}{\partial W_{2,D_{out}}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \mathcal{L}}{\partial W_{D_{in},1}} & \frac{\partial \mathcal{L}}{\partial W_{D_{in},2}} & \cdots & \frac{\partial \mathcal{L}}{\partial W_{D_{in},D_{out}}} \end{bmatrix}$$

$$= \begin{bmatrix} X_1^{(1)} & X_1^{(2)} & \cdots & X_1^{(N)} \\ X_2^{(1)} & X_2^{(2)} & \cdots & X_2^{(N)} \\ \cdots & \cdots & \cdots & \cdots \\ X_{D_{in}}^{(1)} & X_{D_{in}}^{(2)} & \cdots & X_{D_{in}}^{(N)} \end{bmatrix} \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial Y_1^{(1)}} & \frac{\partial \mathcal{L}}{\partial Y_2^{(1)}} & \cdots & \frac{\partial \mathcal{L}}{\partial Y_{D_{out}}^{(1)}} \\ \frac{\partial \mathcal{L}}{\partial Y_1^{(2)}} & \frac{\partial \mathcal{L}}{\partial Y_2^{(2)}} & \cdots & \frac{\partial \mathcal{L}}{\partial Y_{D_{out}}^{(2)}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \mathcal{L}}{\partial Y_1^{(N)}} & \frac{\partial \mathcal{L}}{\partial Y_2^{(N)}} & \cdots & \frac{\partial \mathcal{L}}{\partial Y_{D_{out}}^{(N)}} \end{bmatrix}$$

$$= \boxed{\mathbf{X}^\top \frac{\partial \mathcal{L}}{\partial \mathbf{Y}}}$$

**[3 points] Gradient $\frac{\partial \mathcal{L}}{\partial \mathbf{b}}$:**

Using the equation given in the hint, we have:

$$\frac{\partial \mathcal{L}}{\partial b_j} = \sum_{n=1}^{N} \sum_{m=1}^{D_{out}} \frac{\partial \mathcal{L}}{\partial Y_m^{(n)}} \frac{\partial Y_m^{(n)}}{\partial b_j}$$

$$= \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial Y_j^{(n)}} \cdot 1$$

$$= \boxed{\begin{bmatrix} \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial Y_1^{(n)}} & \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial Y_2^{(n)}} & \cdots & \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial Y_{D_{out}}^{(n)}} \end{bmatrix}}$$

**[3 points] Gradient $\frac{\partial \mathcal{L}}{\partial \mathbf{X}}$:**

Using the equation given in the hint, we have:

$$\frac{\partial \mathcal{L}}{\partial X_i^{(n)}} = \sum_{m=1}^{D_{out}} \frac{\partial \mathcal{L}}{\partial Y_m^{(n)}} \frac{\partial Y_m^{(n)}}{\partial X_i^{(n)}}$$
$$= \sum_{m=1}^{D_{out}} \frac{\partial \mathcal{L}}{\partial Y_m^{(n)}} W_{i,m}$$

With this, consider the following matrix multiplication to understand the vectorized solution:

$$\frac{\partial \mathcal{L}}{\partial X} = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial X_1^{(1)}} & \frac{\partial \mathcal{L}}{\partial X_2^{(1)}} & \cdots & \frac{\partial \mathcal{L}}{\partial X_{D_{in}}^{(1)}} \\ \frac{\partial \mathcal{L}}{\partial X_1^{(2)}} & \frac{\partial \mathcal{L}}{\partial X_2^{(2)}} & \cdots & \frac{\partial \mathcal{L}}{\partial X_{D_{in}}^{(2)}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \mathcal{L}}{\partial X_1^{(N)}} & \frac{\partial \mathcal{L}}{\partial X_2^{(N)}} & \cdots & \frac{\partial \mathcal{L}}{\partial X_{D_{in}}^{(N)}} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial Y_1^{(1)}} & \frac{\partial \mathcal{L}}{\partial Y_2^{(1)}} & \cdots & \frac{\partial \mathcal{L}}{\partial Y_{D_{out}}^{(1)}} \\ \frac{\partial \mathcal{L}}{\partial Y_1^{(2)}} & \frac{\partial \mathcal{L}}{\partial Y_2^{(2)}} & \cdots & \frac{\partial \mathcal{L}}{\partial Y_{D_{out}}^{(2)}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \mathcal{L}}{\partial Y_1^{(N)}} & \frac{\partial \mathcal{L}}{\partial Y_2^{(N)}} & \cdots & \frac{\partial \mathcal{L}}{\partial Y_{D_{out}}^{(N)}} \end{bmatrix} \begin{bmatrix} W_{1,1} & W_{2,1} & \cdots & W_{D_{in},1} \\ W_{1,2} & W_{2,2} & \cdots & W_{D_{in},2} \\ \cdots & \cdots & \cdots & \cdots \\ W_{1,D_{out}} & W_{2,D_{out}} & \cdots & W_{D_{in},D_{out}} \end{bmatrix}$$

$$= \boxed{\frac{\partial \mathcal{L}}{\partial \mathbf{Y}} \mathbf{W}^\top}$$

(b) **[3 points] Gradient of ReLU**

Let $X$ be a tensor and $Y = \text{ReLU}(X)$. Express $\frac{\partial \mathcal{L}}{\partial \mathbf{X}}$ in terms of $\frac{\partial \mathcal{L}}{\partial \mathbf{Y}}$.

For a scalar $x$,

$$y = \text{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases}$$
$$\Rightarrow \frac{\partial y}{\partial x} = \begin{cases} 0 \text{ if } x < 0 \\ 1 \text{ if } x > 0 \end{cases} = \mathbb{I}[x > 0]$$
$$\Rightarrow \frac{\partial \mathcal{L}}{\partial x} = \frac{\partial \mathcal{L}}{\partial y}\frac{\partial y}{\partial x} = \frac{\partial \mathcal{L}}{\partial y}\mathbb{I}[x > 0]$$

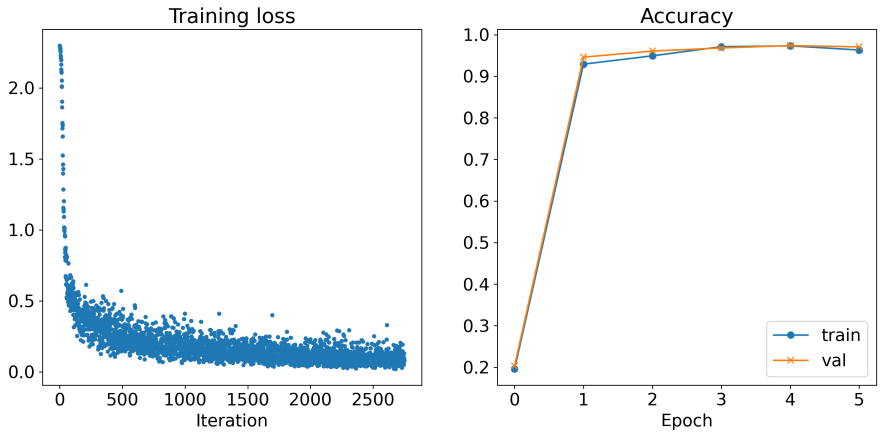
Since the ReLU operation is element-wise, we can generalize this to tensors $X$ and $Y$,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{X}} = \frac{\partial \mathcal{L}}{\partial \mathbf{Y}} \odot \mathbb{I}[\mathbf{X} > 0]$$

Where $\odot$ represents the Hadamard or element-wise product and $\mathbb{I}[.]$ represents the indicator function.

(c) **Answer [14 points]:** Autograder.

(d) **Answer [1 points]:**




(e) **Answer [1 points]:** We got 96.31% as the test accuracy.