Homework 3

Problem 1.

(a) From hw1 p2(d), we have $L(\vec{w}) = C\sum_{i=1}^{n} \max\{1-y_i\vec{w}^T\vec{x}_i, 0\} + \|\vec{w}\|_2^2$

$$\vec{w}^{(t+1)} = \vec{w}^{(t)} - \eta \nabla L(\vec{w}^{(t)})$$

$$\nabla L(\vec{w}) = 2\vec{w} - C\sum_{i\in M} y_i\vec{x}_i, \text{ where } M = \{i \mid 1-y_i\vec{w}^T\vec{x}_i > 0\}$$

Starting with $\vec{w}^{(0)}=0$, $\alpha_i^{(0)}=0$, $\vec{w}^{(0)} = \sum_{i=1}^{n}\alpha_i^{(0)}\vec{x}_i$, the first update is

$$\vec{w}^{(1)} = 0 - \eta\nabla L(\vec{w}^{(0)}) = \eta C \sum_{i\in M^{(0)}} y_i\vec{x}_i = \sum_{i=1}^{n}\alpha_i^{(1)}\vec{x}_i$$

Suppose that $\vec{w}^{(t)} = \sum_{i=1}^{n}\alpha_i^{(t)}\vec{x}_i$, then we have

$$\vec{w}^{(t+1)} = \vec{w}^{(t)} - \eta\nabla L(\vec{w}^{(t)}) = \sum_{i=1}^{n}\alpha_i^{(t)}\vec{x}_i - \eta\Big(2\vec{w}^{(t)} - C\sum_{i\in M^{(t)}} y_i\vec{x}_i\Big)$$

$$= (1-2\eta)\sum_{i=1}^{n}\alpha_i^{(t)}\vec{x}_i + \eta C\sum_{i\in M^{(t)}} y_i\vec{x}_i = \sum_{i=1}^{n}\alpha_i^{(t+1)}\vec{x}_i$$

because $\sum_{i\in M^{(t)}} y_i\vec{x}_i$ is also a linear combination of $\vec{x}_i$

Then, we get $\vec{w}^{(t+1)} = \sum_{i=1}^{n}\alpha_i^{(t+1)}\vec{x}_i$

Therefore, we can express the weight vector as $\vec{w}^{(t)} = \sum_{i=1}^{n}\alpha_i^{(t)}\vec{x}_i$

(b) $\vec{w} = \sum_{i=1}^{n}\alpha_i\vec{x}_i$

$$\|\vec{w}\|^2 = \vec{w}^T\vec{w} = \Big(\sum_{i=1}^{n}\alpha_i\vec{x}_i\Big)^T\Big(\sum_{i=1}^{n}\alpha_i\vec{x}_i\Big) = \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\vec{x}_i^T\vec{x}_j = \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j K_{ij} = \vec{\alpha}^T K_{ij}\vec{\alpha}$$

$$\max(1-y_i\vec{w}^T\vec{x}_i, 0) = \max\Big(1-y_i\sum_{j=1}^{n}\alpha_j\vec{x}_j^T\vec{x}_i, 0\Big) = \max\Big(1-y_i\sum_{j=1}^{n}\alpha_j K_{ji}, 0\Big)$$

Thus, $L(\vec{\alpha}) = C\sum_{i=1}^{n}\max(1-y_i\vec{w}^T\vec{x}_i, 0) + \|\vec{w}\|_2^2$

$$= C\sum_{i=1}^{n}\max\Big(1-y_i\sum_{j=1}^{n}\alpha_j K_{ji}, 0\Big) + \vec{\alpha}^T K_{ij}\vec{\alpha}$$

(c) we define $m_i = y_i\sum_{j=1}^{n}\alpha_j K_{ji}$

So, the hinge loss for sample i is $L_i(\alpha) = \max(1-m_i, 0)$

For each i, if $m_i \geq 1$, $\frac{\partial L_i}{\partial\alpha_k} = 0$; if $m_i < 1$, $\frac{\partial L_i}{\partial\alpha_k} = -y_i\frac{\partial}{\partial\alpha_k}\sum_{j=1}^{n}\alpha_j K_{ji} = -y_i K_{ki}$

Also, $\frac{\partial}{\partial\alpha_k}(\vec{\alpha}^T K_{ij}\vec{\alpha}^T) = 2\sum_{j=1}^{n}K_{kj}\alpha_j$

Thus, $\frac{\partial L(\vec{\alpha})}{\partial\alpha_k} = 2\sum_{j=1}^{n}K_{kj}\alpha_j - C\sum_{i:\, m_i<1} y_i K_{ki}$

Problem 2

(a) $G \in R^{n \times m}$, $G_{ij} = \vec{x}_i^T \vec{z}_j$

$$G = \begin{pmatrix} \vec{x}_1^T\vec{z}_1 & \vec{x}_1^T\vec{z}_2 & \cdots & \vec{x}_1^T\vec{z}_m \\ \vec{x}_2^T\vec{z}_1 & \vec{x}_2^T\vec{z}_2 & \cdots & \vec{x}_2^T\vec{z}_2 \\ \vdots & \vdots & & \\ \vec{x}_n^T\vec{z}_1 & \vec{x}_n^T\vec{z}_2 & \cdots & \vec{x}_n^T\vec{z}_m \end{pmatrix} = \begin{pmatrix} \vec{x}_1^T \\ \vec{x}_2^T \\ \vdots \\ \vec{x}_n^T \end{pmatrix} \begin{pmatrix} \vec{z}_1 & \vec{z}_2 & \cdots & \vec{z}_m \end{pmatrix} = X^T Z$$

Thus, G can be expressed in terms of matrix multiplication.

(b) $S, R \in R^{n \times m}$, $S_{ij} = \vec{x}_i^T \vec{x}_i$, $R_{ij} = \vec{z}_j^T \vec{z}_j$, $D^2 \in R^{n \times m}$

$$D_{ij}^2 = (\vec{x}_i - \vec{z}_j)^T (\vec{x}_i - \vec{z}_j) = \vec{x}_i^T \vec{x}_i + \vec{z}_j^T z_j - 2\vec{x}_i^T \vec{z}_j = S_{ij} + R_{ij} - 2G_{ij}$$

Thus, $D^2 = S + R - 2G$

(c) We can use $D^2 = \max(S+R-2G, 0) \Rightarrow D = \sqrt{\max(S+R-2G, 0)}$

In numpy:

```
import numpy as np

D_squared = S + R - 2G

D_squared = np.maximum(D_squared, 0)

D = np.sqrt(D_squared)
```

Problem 3.

(a) $\phi(\vec{x}_i) = [1, \sqrt{2}(\vec{x}_i)_1, \sqrt{2}(\vec{x}_i)_2, \sqrt{2}(\vec{x}_i)_3, (\vec{x}_i)_1^2, (\vec{x}_i)_2^2, (\vec{x}_i)_3^2, \sqrt{2}(\vec{x}_i)_1(\vec{x}_i)_2, \sqrt{2}(\vec{x}_i)_1(\vec{x}_i)_3, \sqrt{2}(\vec{x}_i)_2(\vec{x}_i)_3]$

$$\langle \phi(\vec{x}_i), \phi(\vec{x}_j) \rangle = 1 + 2(\vec{x}_i)_1(\vec{x}_j)_1 + 2(\vec{x}_i)_2(\vec{x}_j)_2 + 2(\vec{x}_i)_3(\vec{x}_j)_3 + (\vec{x}_i)_1^2(\vec{x}_j)_1^2 + (\vec{x}_i)_2^2(\vec{x}_j)_2^2$$

$$+ (\vec{x}_i)_3^2(\vec{x}_j)_3^2 + 2(\vec{x}_i)_1(\vec{x}_i)_2(\vec{x}_j)_1(\vec{x}_j)_2 + 2(\vec{x}_i)_1(\vec{x}_i)_3(\vec{x}_j)_1(\vec{x}_j)_3 + 2(\vec{x}_i)_2(\vec{x}_i)_3(\vec{x}_j)_2(\vec{x}_j)_3$$

$$= 1 + 2\vec{x}_i^T\vec{x}_j + (\vec{x}_i^T\vec{x}_j)^2$$

Since, $\phi(\vec{x}_i) \in R^D$, we can get $D = 10$

Explicitly computing the feature mappings becomes expensive as D increase. But kernel methods allow us to compute $\langle \phi(\vec{x}_i), \phi(\vec{x}_j) \rangle$ without ever computing the feature mappings which will save time and computation resources.

(b) $A_1 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

$\det(A_1 - \lambda I) = 0 \Rightarrow \begin{vmatrix} 1-\lambda & 1 \\ 1 & 1-\lambda \end{vmatrix} = (1-\lambda)^2 - 1 = \lambda^2 - 2\lambda = 0$

$\Rightarrow \lambda = 0, 2 \quad \geq 0$

$\Rightarrow A_1$ is positive semidefinite

$A_2 = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 2 \end{bmatrix}$

$\det(A_2 - \lambda I) = 0 \Rightarrow \begin{vmatrix} 2-\lambda & 1 & 1 \\ 1 & 2-\lambda & 0 \\ 1 & 0 & 2-\lambda \end{vmatrix} = (2-\lambda)^3 + 0 + 0 - (2-\lambda) - 0 - (2-\lambda)$

$= 8 - 12\lambda + 6\lambda^2 - \lambda^3 - 4 + 2\lambda$

$= (2-\lambda)(\lambda^2 - 4\lambda + 2)$

$\Rightarrow \lambda = 2, 2+\sqrt{2}, 2-\sqrt{2} > 0$

$\Rightarrow A_2$ is strictly positive definite

$A_3 = \begin{bmatrix} 2 & 1 & -1 \\ 1 & 1 & 1 \\ -1 & 1 & 2 \end{bmatrix}$

$\det(A_3) = 4 - 1 - 1 - 1 - 2 - 2 = -3 < 0$

$\Rightarrow A_3$ is neither strictly positive definite nor positive semidefinite

(c) RBF Kernel : $K(\vec{x}, \vec{x}') = \exp\left(-\frac{\|\vec{x}-\vec{x}'\|^2}{2l^2}\right)$

we know $\|\vec{x}-\vec{x}'\|^2 = \|\vec{x}\|^2 + \|\vec{x}'\|^2 - 2\vec{x}^T\vec{x}'$

So, $k(\vec{x}, \vec{x}') = \exp\left(-\frac{\|\vec{x}\|^2}{2l^2}\right) \cdot \exp\left(-\frac{\|\vec{x}'\|^2}{2l^2}\right) \cdot \exp\left(\frac{\vec{x}^T\vec{x}'}{l^2}\right)$

$= \exp\left(-\frac{\|\vec{x}\|^2}{2l^2}\right) \cdot \exp\left(-\frac{\|\vec{x}'\|^2}{2l^2}\right) \cdot \sum_{K=0}^{\infty} \frac{1}{K!}\left(\frac{\vec{x}^T\vec{x}'}{l^2}\right)^K$

we can define $\phi(\vec{x}) = \left[\exp\left(-\frac{\|\vec{x}\|^2}{2l^2}\right) \cdot \frac{1}{\sqrt{K!}}\left(\frac{\vec{x}^\alpha}{l^{|\alpha|}}\right)\right]_{\alpha \in N^d}$

Thus, this is equivalent to $K(\vec{x}, \vec{x}') = \langle \phi(\vec{x}), \phi(\vec{x}') \rangle$

(d) According to the definition, $K(\cdot, \cdot)$ is a valid kernel, if the function $K(\vec{x}, \vec{x}')$ is both

- symmetric $k(\vec{x}, \vec{x}') = K(\vec{x}', \vec{x})$ for all $\vec{x}, \vec{x}'$
- positive semidefinite: $K(\cdot, \cdot)$ is PSD if for all finite subsets $\{\vec{x}_1, \cdots, \vec{x}_m\}$, $\vec{x}_i \in X$, K is a PSD matrix

Thus, for $\tilde{K}(\vec{x}, \vec{x}') = c\,k(\vec{x}, \vec{x}')$, given that $K(\vec{x}, \vec{x}')$ is a valid kernel and $c \geq 0$

we can have $\tilde{K}(\vec{x}, \vec{x}') = c\,k(\vec{x}, \vec{x}') = c\,k(\vec{x}', \vec{x}) = \tilde{K}(\vec{x}', \vec{x})$, and

$K \in R^{m\times n}$, $K_{ij} = k(\vec{x}_i, \vec{x}_j) \geq 0$, because K is PSD. $\forall \vec{v} \in R^m$, $\vec{v}^T K \vec{v} \geq 0$

Since, $\tilde{K} = c \cdot K$, for any $\vec{v} \in R^m$, $\vec{v}^T\tilde{K}\vec{v} = \vec{v}^T(cK)\vec{v} = c \cdot \vec{v}^T K \vec{v} \geq 0$

$\tilde{K}$ is also PSD.

Thus, $\tilde{K}(\vec{x}, \vec{x}') = c\,k_1(\vec{x}, \vec{x}')$ is a valid kernel

Problem 4

(a) Kernelize the K-means algorithm

In standard K-means, we are given $\vec{x}_1, \cdots, \vec{x}_n \in R^d$, and the goal is to assign each point to one of K-clusters s.t. the total squared distance to the cluster center is minimized $L = \sum_{i=1}^{n} \|\vec{x}_i - \vec{\mu}_{c_i}\|^2$, where $c_i \in \{1, \cdots, K\}$ is the cluster assignment of point $\vec{x}_i$, $\vec{\mu}_i$ is the center of cluster j.

We map data into a feature space via a mapping $\phi(\cdot)$, and use a kernel function $K(\vec{x}, \vec{x}') = \langle \phi(\vec{x}), \phi(\vec{x}') \rangle$

The new objective becomes $L = \sum_{i=1}^{n} \|\phi(\vec{x}_i) - \vec{\mu}_{c_i}\|^2$ with cluster center $\vec{\mu}_j = \frac{1}{|C_j|} \sum_{\vec{x}_l \in C_j} \phi(\vec{x}_l)$

$$\|\phi(\vec{x}_i) - \mu_j\|^2 = \langle \phi(\vec{x}_i), \phi(\vec{x}_i) \rangle - 2 \langle \phi(\vec{x}_i), \vec{\mu}_j \rangle + \langle \vec{\mu}_j, \vec{\mu}_j \rangle$$

$$= K(\vec{x}_i, \vec{x}_i) - \frac{2}{|C_j|} \sum_{\vec{x}_l \in C_j} K(\vec{x}_i, \vec{x}_l) + \frac{1}{|C_j|^2} \sum_{\vec{x}_l, \vec{x}_m \in C_j} K(\vec{x}_l, \vec{x}_m)$$

(b) In kernel K-means, the cluster centers $\vec{\mu}_j$ are defined in the high-dimensional feature space induced by the kernel function. Since, we do not have explicit access to the mapping $\phi(\vec{x})$, we cannot visualize or represent these cluster centers even the original input space is 2D

Problem 5

(a) Assuming noise-free training data $D=\{(\vec{x}_i, f_i)\}_{i=1,\cdots,n}$ with $f_i = f(\vec{x}_i)$

From the lecture notes, we have $cov(f_*) = V[f_*] = K_{**} - K_*^T K^{-1} K_*$,

where $K = K(X,X) \in R^{n\times n}$ $K_* = K(X, \vec{x}^*) \in R^n$, $K_{**} = K(\vec{x}^*, \vec{x}^*) \in R$

$cov(f_*) = K(\vec{x}^*, \vec{x}^*) - K(X, \vec{x}^*)^T K(X,X)^{-1} K(X, \vec{x}^*)$

If the test point $\vec{x}^*$ is the same as the training point $\vec{x}_i$, then

$cov(f_i) = K(\vec{x}_i, \vec{x}_i) - K(X, \vec{x}_i)^T K(X,X)^{-1} K(X, \vec{x}_i)$

Let $K_{i,:}$ denote the i-th row of K

$cov(f_i) = K_{ii} - K_{i,:} K^{-1} K_{:,i} = 0$

(b) Assume observations with Gaussian noise: $\vec{y} = \vec{f} + \vec{\varepsilon}$, $\vec{f} \sim N(0, K_\theta)$

$\vec{\varepsilon} \sim N(0, \sigma_n^2 I)$, then $\vec{y} \sim N(0, K_\theta + \sigma_n^2 I)$

Then $p(\vec{y}|X,\theta) \sim N(\vec{y}|0, K_\theta + \sigma_n^2 I)$

$$p(\vec{y}|X,\theta) = \frac{1}{(2\pi)^{n/2} |K_\theta + \sigma_n^2 I|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\vec{y}^T (K_\theta + \sigma_n^2 I)^{-1} \vec{y}\right)$$

The log marginal likelihood (LML) is

$$\log p(\vec{y}|X,\theta) = -\frac{1}{2}\vec{y}^T (K_\theta + \sigma_n^2 I)^{-1}\vec{y} - \frac{1}{2}\log|K_\theta + \sigma_n^2 I| - \frac{n}{2}\log(2\pi)$$

(c) Using Cholesky decomposition, $K_y = K_\theta + \sigma_n^2 I = LL^T$, where L is a lower triangular matrix.

Define $\alpha = L^T \backslash (L \backslash \vec{y}) = (L^T)^{-1} L^{-1} \vec{y} = (LL^T)^{-1}\vec{y} = K_y^{-1}\vec{y}$

Since, $\vec{y}^T K_y^{-1} \vec{y} = \vec{y}^T \alpha$, $|K_y| = |LL^T| = (\prod_{i=1}^n L_{ii})^2$, $\log|K_y| = 2\sum_{i=1}^n \log L_{ii}$

Then $\log p(\vec{y}|X,\theta) = -\frac{1}{2}\vec{y}^T\alpha - \sum_{i=1}^n \log L_{ii} - \frac{n}{2}\log(2\pi)$

(d) Define the negative log marginal likelihood

$$L(\theta) = -\log p(\vec{y}|X,\theta) = \frac{1}{2}\vec{y}^T K_y^{-1}\vec{y} + \frac{1}{2}\log|K_y| + \frac{n}{2}\log(2\pi)$$

$$\frac{\partial L}{\partial \theta_j} = \frac{1}{2}\vec{y} K_y^{-1} \frac{\partial K_y}{\partial \theta_i} K_y^{-1}\vec{y} - \frac{1}{2} tr\left(K_y^{-1}\frac{\partial K_y}{\partial \theta_i}\right)$$

$$= \frac{1}{2}\left[\alpha^T \frac{\partial K_y}{\partial \theta_i}\alpha - \frac{1}{2} tr\left(K_y^{-1}\frac{\partial K_y}{\partial \theta_i}\right)\right]$$

$$= \frac{1}{2} tr\left[(\alpha\alpha^T - K_y^{-1})\frac{\partial K_y}{\partial \theta_i}\right]$$