CS-E4830 Kernel Methods in Machine Learning
Assignment 1

Kunal Ghosh, 546247

September 20, 2016

1 Solution to Question 1

1.1 (a) True

Its given that K_1 and K_2 are Positive semi-definite matrices. So, for any vector v we have

$$\begin{aligned}
v^T K_1 v \geq 0 \text{ and } v^T K_2 v \geq 0 \\
\text{from question } K = aK_1 + bK_2 \text{ where } a, b \in \mathcal{R}^+ \\
\text{then, } v^T K v &= v^T(aK_1 + bK_2)v \\
&= v^T(aK_1)v + v^T(bK_2)v \\
&= a(v^T K_1 v) + b(v^T K_2)v \\
&\text{Since, } a, b > 0 \text{ and } v^T K_1 v, v^T K_2 v \geq 0 \\
&\implies a(v^T K_1 v) + b(v^T K_2)v \geq 0 \\
&\implies v^T(aK_1 + bK_2)v \geq 0 \\
&\implies v^T K v \geq 0
\end{aligned} \tag{1}$$

So, K is Positive semi-definite. Hence a Kernel Matrix.

1.2 (b) False

If matrix K is defined as $K = K_1 - K_2$, where K_1, K_2 are positive semi-definite matrices, then

$$\begin{aligned}
v^T K v &= v^T(K_1 - K_2)v \\
&= v^T(K_1)v - v^T(K_2)v
\end{aligned} \tag{2}$$

We know that, $v^T K_1 v, v^T K_2 v \geq 0$. However, the difference of two non-negative numbers is not always non-negative.

$$\begin{aligned}
&\implies v^T(K_1)v - v^T(K_2)v \ngeq 0 \\
&\implies v^T K v \ngeq 0
\end{aligned} \tag{3}$$

Hence $K = K_1 - K_2$ is not always positive semi-definite. Hence K need not be a Kernel Matrix.

1.3 (c) False

If matrix K is defined as $K = K_1 K_2$ where K_1, K_2 are positive semi-definite matrices, then **[Counter Example]** : Product of two symmetric matrices is not always symmetric.

for, $K_1 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$ and $K_2 = \begin{bmatrix} -2 & 1 \\ 1 & -3 \end{bmatrix}$

where $\det(K_1) > 0$ and $\det(K_2) > 0$ so all the eigen values of K_1 and K_2 are positive. Hence K_1 and K_2 are positive semi-definite matrices. But,

$K = K_1 K_2 = \begin{bmatrix} -3 & 4 \\ 3 & -4 \end{bmatrix}$

is not a symmetric matrix. Therefore $K = K_1 K_2$ need not be a Kernel Matrix.

1.4 (d) False

[Counter Example] : Examples of K_1 and K_2 given above (previous section) are valid Kernel Matrices but don't have all positive entries.

1.5 (e) True

e). $K_1(x_i, x_j) = \phi_1(x_i)^T \phi_1(x_j) = \sum_{l=1}^{d} \phi_{1l}(x_i) \cdot \phi_{1l}(x_j)$.

$K_2(x_i, x_j) = \phi_2(x_i)^T \phi_2(x_j) = \sum_{m=1}^{d} \phi_{2m}(x_i) \cdot \phi_{2m}(x_j)$.

so $K(x_i, x_j) = K_1(x_i, x_j) \cdot K_2(x_i, x_j) = \left[\sum_{l=1}^{d} \phi_{1l}(x_i) \cdot \phi_{1l}(x_j)\right]\left[\sum_{m=1}^{d} \phi_{2m}(x_i) \cdot \phi_{2m}(x_j)\right]$

Expanding out a few terms.

$= \left[\phi_{11}(x_i) \cdot \phi_{11}(x_j) + \phi_{12}(x_i) \cdot \phi_{12}(x_j) + \dots + \phi_{1d}(x_i) \cdot \phi_{1d}(x_j)\right]$ has d terms

$*$

$\left[\phi_{21}(x_i) \cdot \phi_{21}(x_j) + \phi_{22}(x_i) \cdot \phi_{22}(x_j) + \dots + \phi_{2d}(x_i) \cdot \phi_{2d}(x_j)\right]$ has d terms.

= when we find the product of the above two terms it is going to be a term with dxd terms.

$= \Big[\phi_{11}(x_i)\phi_{11}(x_j)\phi_{21}(x_i)\phi_{21}(x_j) + \phi_{11}(x_i)\phi_{11}(x_j)\phi_{22}(x_i)\phi_{22}(x_j) + \dots + \phi_{11}(x_i)\phi_{11}(x_j)\phi_{2d}(x_i)\phi_{2d}(x_j)$ (d terms)

d terms

$+ \phi_{12}(x_i)\phi_{12}(x_j)\phi_{21}(x_i)\phi_{21}(x_j) + \dots$

$\vdots$

$+ \phi_{1d}(x_i)\phi_{1d}(x_j) \cdot \phi_{21}(x_i)\phi_{21}(x_j) + \dots + \phi_{1d}(x_i)\phi_{1d}(x_j)\phi_{2d}(x_i)\phi_{2d}(x_j)\Big]$

~~Individual terms among these~~ ~~These~~ dxd terms can be ~~written~~ re-written as.

$\Big[(\phi_{11}(x_i)\phi_{21}(x_i))(\phi_{11}(x_j)\phi_{21}(x_j)) + (\phi_{11}(x_i)\phi_{22}(x_i))(\phi_{11}(x_j)\phi_{22}(x_j)) + \dots + (\phi_{11}(x_i)\phi_{2d}(x_i))(\phi_{11}(x_j)\phi_{2d}(x_j))$

$+ (\phi_{12}(x_i)\phi_{21}(x_i))(\phi_{12}(x_j)\phi_{21}(x_j)) + (\phi_{12}(x_i)\phi_{22}(x_i)) \cdot (\phi_{12}(x_j) \cdot \phi_{22}(x_j)) + \dots + (\phi_{12}(x_i)\phi_{2d}(x_i))(\phi_{12}(x_j)\phi_{2d}(x_j))$

$+ \dots$

$\vdots$

$+ (\phi_{1d}(x_i)\phi_{21}(x_i))(\phi_{1d}(x_j)\phi_{21}(x_j)) + (\phi_{1d}(x_i) \cdot \phi_{22}(x_i))(\phi_{1d}(x_j) \cdot \phi_{22}(x_j)) + \dots + (\phi_{1d}(x_i)\phi_{2d}(x_i))(\phi_{1d}(x_j)\phi_{2d}(x_j))$

This sum can be re-written as a dot-product of a vector $\phi(x_i)$ with itself. i.e. $\phi(x_i)^T\phi(x_j)$ ~~such that~~ where $\phi(x_j)$ is defined as follows.

$\Big[\phi_{11}(x) \cdot \phi_{21}(x),\ \phi_{11}(x)\phi_{22}(x),\ \dots,\ \phi_{11}(x)\phi_{2d}(x),$

$\phi_{12}(x)\phi_{21}(x),\ \phi_{12}(x) \cdot \phi_{22}(x),\ \dots,\ \phi_{12}(x) \cdot \phi_{2d}(x),$

$\vdots$

$\phi_{1d}(x) \cdot \phi_{21}(x),\ \dots\dots,\ \dots,\ \phi_{1d}(x) \cdot \phi_{2d}(x)\Big]^T$

This is exactly what $\phi(x)$ would be, if it were defined as $\phi_1(x) \otimes \phi_2(x)$. Hence if $K(x_i, x_j) = K_1(x_i, x_j) \times K_2(x_i, x_j)$ then $K(x_i, x_j)$ can be written as $\phi(x_i)^T \phi(x_j)$ where $\phi(x) = \phi_1(x) \otimes \phi_2(x)$.

2 Solution to Question 2

Question 2: To show, $h(x) = \text{sign}\left(\|\Phi(x) - c_-\|^2 - \|\Phi(x) - c_+\|^2\right)$ can be re-written as

$$\text{sign}\left(\sum_{i=1}^{m} \alpha_i K(x, x_i) + b\right).$$

we know that $\|\Phi(x) - c_-\|^2 = \langle \Phi(x) - c_-, \Phi(x) - c_- \rangle$.

so $h(x) = \text{sign}\left(\langle \Phi(x) - c_-, \Phi(x) - c_- \rangle - \langle \Phi(x) - c_+, \Phi(x) - c_+ \rangle\right)$.

Using the property of linearity of inner product according to which
$\langle x+y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$ and $\langle x, y+z \rangle = \langle x, y \rangle + \langle x, z \rangle$.

$$h(x) = \text{sign}\left(\langle \Phi(x), \Phi(x) - c_- \rangle - \langle c_-, \Phi(x) - c_- \rangle - \left\{\langle \Phi(x), \Phi(x) - c_+ \rangle - \langle c_+, \Phi(x) - c_+ \rangle\right\}\right)$$

$$= \text{sign}\left(\langle \Phi(x), \Phi(x) \rangle - \langle \Phi(x), c_- \rangle - \left\{\langle c_-, \Phi(x) \rangle - \langle c_-, c_- \rangle\right\} - \left\{\langle \Phi(x), \Phi(x) \rangle - \langle \Phi(x), c_+ \rangle\right\} + \left\{\langle c_+, \Phi(x) \rangle - \langle c_+, c_+ \rangle\right\}\right)$$

$$= \text{sign}\left(\cancel{\langle \Phi(x), \Phi(x) \rangle} - \langle \Phi(x), c_- \rangle - \langle c_-, \Phi(x) \rangle + \langle c_-, c_- \rangle - \cancel{\langle \Phi(x), \Phi(x) \rangle} + \langle \Phi(x), c_+ \rangle + \langle c_+, \Phi(x) \rangle - \langle c_+, c_+ \rangle\right.$$

Continues in next page…

also $\langle a, b\rangle \equiv \langle b, a\rangle$.

$$\therefore\ h(x) = \text{sign}\left(2\langle \Phi(x), c_+\rangle - 2\langle \Phi(x), c_-\rangle + \langle c_-, c_-\rangle - \langle c_+, c_+\rangle\right).$$

$$= \text{sign}\left(2\langle \Phi(x), c_+ - c_-\rangle + \langle c_-, c_-\rangle - \langle c_+, c_+\rangle\right) \quad — (1)$$

$$\cancel{= \text{sign}\left(2\langle \Phi(x), c_+ - c_-\rangle + \right.}$$

c_- and c_+ are vectors. $\therefore$ $\langle c_-, c_-\rangle = \left[\frac{1}{m_-}\sum_{i\in I^-}\Phi(x_i)\right]^T\left[\frac{1}{m_-}\sum_{i\in I^-}\Phi(x_i)\right]$

$$= \frac{1}{m_-^2}\sum_{i,j\in I^-}\Phi(x_i)^T\cdot\Phi(x_j). = \frac{1}{m_-^2}\sum_{i,j\in I^-}\Phi(x_{I_i})^T\Phi(x_{I_j}) \quad \cancel{\text{where } x_I = x_i \ \forall i\in I^-}$$

$$= \frac{1}{m_-^2}\sum_{i,j\in I^-}\langle \Phi(x_{I_i}), \Phi(x_{I_j})\rangle$$

$$= \frac{1}{m_-^2}\sum_{i,j\in I^-} K(x_i, x_j). \quad \text{using kernel trick.} \quad — (2)$$

similarly $\langle c_+, c_+\rangle = \frac{1}{m_+^2}\sum_{i,j\in I^+} K(x_i, x_j). \quad — (3).$

and

$$\langle \Phi(x), c_+ - c_-\rangle = \Phi(x)^T\left[\frac{1}{m_+}\sum_{i\in I^+}\Phi(x_i) - \frac{1}{m_-}\sum_{i\in I^-}\Phi(x_i)\right]$$

$$= \frac{1}{m_+}\sum_{i\in I^+}\Phi(x)^T\Phi(x_i) - \frac{1}{m_-}\sum_{i\in I^-}\Phi(x)^T\Phi(x_i). \quad — (1)$$

$$= \sum_{i\in m}\alpha_i\Phi(x)^T\Phi(x_i).$$

$$\alpha_i = \begin{cases}\frac{1}{m_+} & \text{if } y_i = +1\\ -\frac{1}{m_-} & \text{if } y_i = -1\end{cases}$$

$$\therefore\ h(x) = \text{sign}\left(2\sum_{i=1}^{m}\alpha_i\,\Phi(x)^T\Phi(x_i) + \underbrace{\frac{1}{m_+^2}\sum_{i,j\in I^+}K(x_i, x_j) + \frac{1}{m_-^2}\sum_{i,j\in I^-}K(x_i, x_j)}_{b}\right)$$

$$h(x) = \text{sign}\left(\sum_{i=1}^{m}\alpha_i\,\Phi(x)^T\Phi(x_i) + b\right).$$

taking the 2 out, common doesn't affect the sign.

$$b = \frac{1}{2m_+^2}\sum_{i,j\in I^+}K(x_i, x_j) + \frac{1}{2m_-^2}\sum_{i,j\in I^-}K(x_i, x_j)$$

$$\alpha_i = \begin{cases}\frac{1}{m_+} & \text{if } y_i = +1.\\ \frac{-1}{m_-} & \text{if } y_i = -1.\end{cases}$$

2 Solution to Question 3

```
function kernel_mat = gaussian_kernel(X, Z, sigma)
  % X is (n,d)
  % Z is (m,d)
  % sigma is scalar
  % gaussian kernel = exp( -0.5 * (1/sigma^2)* || phi(x1) - phi(x2) ||^2)
  % Assuming, here X and Z are already projected in phi (feature) space.
  % So in this case, gaussian kernel = exp( -0.5 * (1/sigma^2)* || x1 - x2 ||^2)
  % Dimension of kernel_mat is (n,m)
  [n,d] = size(X);
  [m,d] = size(Z);
  kernel_mat = zeros(n,m);
  for row = 1:n
    kernel_mat(row,:) = sum(bsxfun(@minus,Z,X(row,:)).^2,2)';
  end
  kernel_mat = exp(kernel_mat * (-0.5 /(sigma^2)));
end
```

3 Solution to Question 4

```
function y_pred = parzen_classify(Kx_train, Kx_train_test, y_train)
  m_pos = sum(y_train == 1);
  m_neg = length(y_train) - m_pos;
  % size(Kx_train) (800,800)
  % size(Kx_train_test) (800,200)
  b = zeros(2,1);
  negidxs = find(y_train == -1);
  for i = negidxs'
    b(1) = b(1) + sum(Kx_train(i,negidxs));
  end
  b(1) = b(1)/(2*m_neg^2);

  posidxs = find(y_train == 1);
  for i = posidxs'
    b(2) = b(2) + sum(Kx_train(i,posidxs));
  end
  b(2) = b(2)/(2*m_pos^2);

  const = b(1) - b(2);

  % here alpha is (800,1)
  alpha = (1/m_pos)*ones(size(y_train));
  alpha(negidxs) = (-1/m_neg);

  y_pred = sign((alpha' * Kx_train_test) + const);
end
```

4 Solution to Question 5

In the caption in the figures below σ is a parameter of the gaussian kernel which is used for the task of classification in this problem. Among the figures, fig-1 shows the learning curves and fig-2 shows the decision boundary for various values of σ. More description in the captions of the figures.

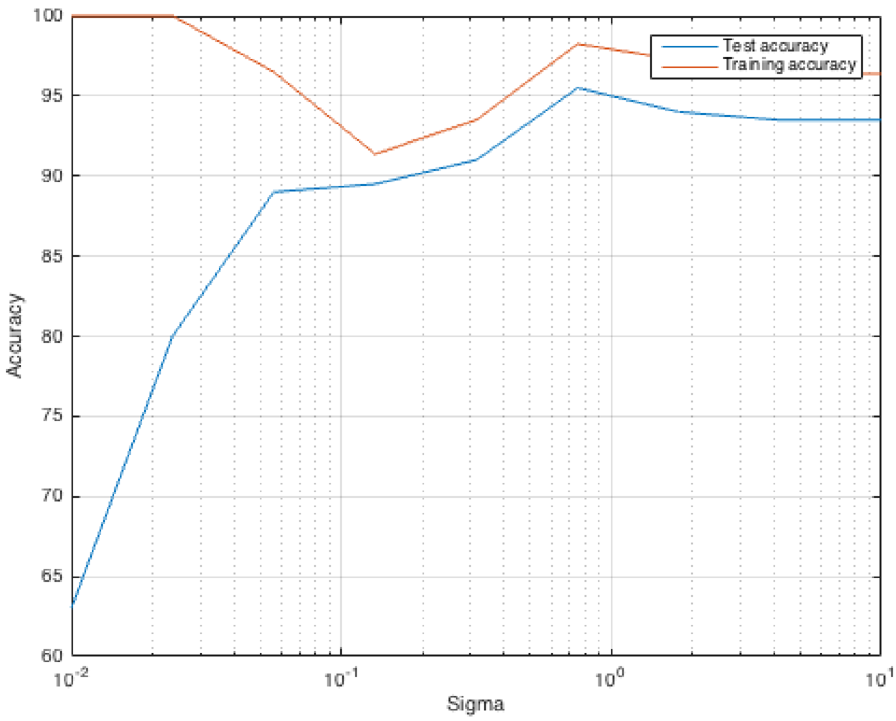

Figure 1: Learning curves: Initially, with low values of sigma there is overfitting (high training accuracy, low test accuracy) as the sigma values increase the model generalizes better until sigma reaches ≈ 0.7 then the model starts to underfit.

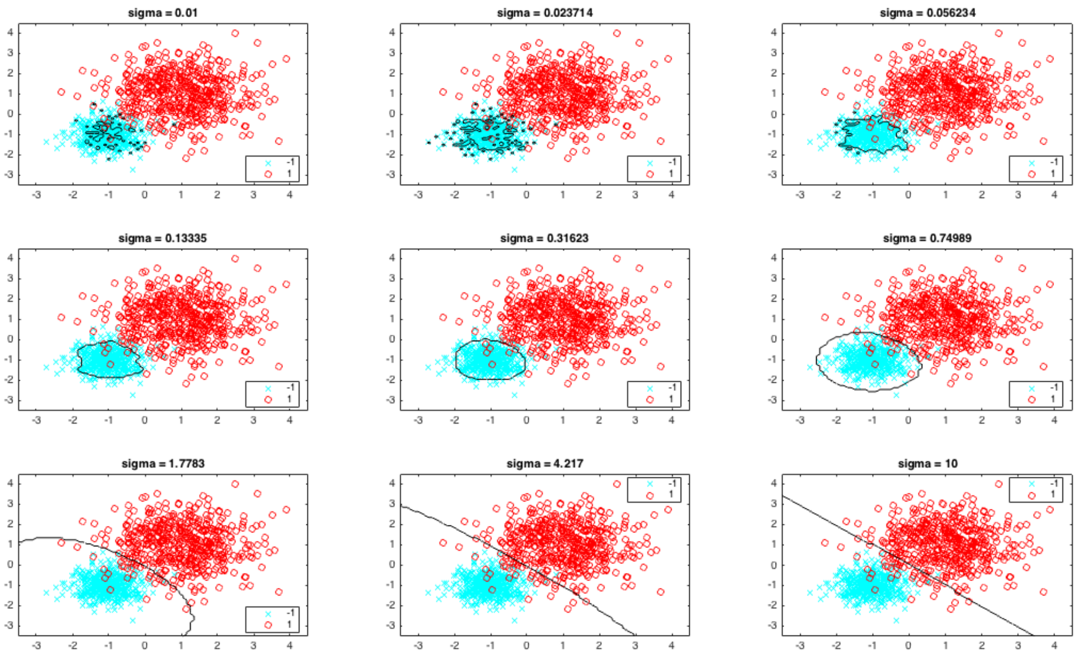

Figure 2: Decision Boundaries: For low values of sigma overfitting is clearly visible (small "circles" tightly encircling certain datapoints). As the sigma increases the model generalizes better, for $\sigma \approx 0.7$ the test accuracy is the best (least number of red-circles inside the decision boundary). Finally, as the sigma keeps on increasing the decision boundary tends to a linear decision boundary which isn't the most ideal for this dataset as can be seen in the learning curves Fig-1 above where both the training and test accuracy drops as the sigma is increased beyond ≈ 0.7.