Supervised Learning (COMP0078) Coursework 1

Student No. 22204524

November 16, 2022

PART I

a.1 Linear Regression

Q1

(a) Produce a plot superimposing the four different curves corresponding to each fit over four data points

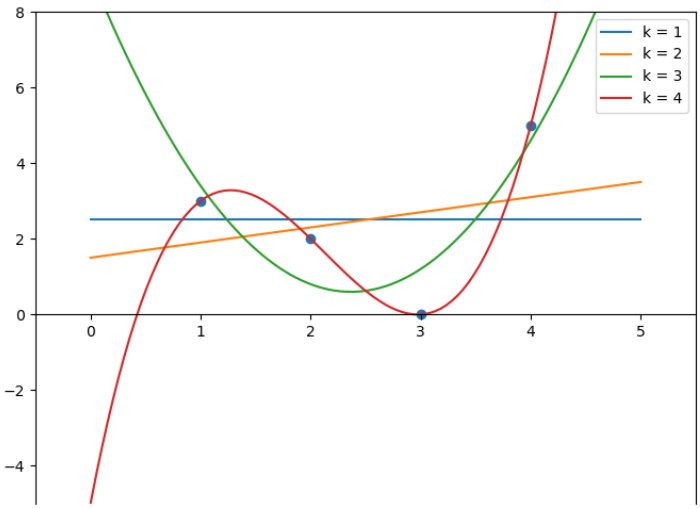

Figure 1: Plot of four different curves for dataset

(b) Equations for three curves when k = 1, 2, 3

k = 1: $y = 2.5$
k = 2: $y = 1.5 + 0.4x$
k = 3: $y = 9 – 7.1x + 1.5x^2$

(c) For each curves give the MSE:

k = 1: MSE = 3.25
k = 2: MSE = 3.05

k = 3: MSE = 0.8
k = 4: MSE = $1.84 \times 10^{-26} \approx 0$

Q2

(a) (i) Plot the function $sin^2(2\pi x)$ in the range $-1 \leq x \leq 1$ with the points of the data set superimposed

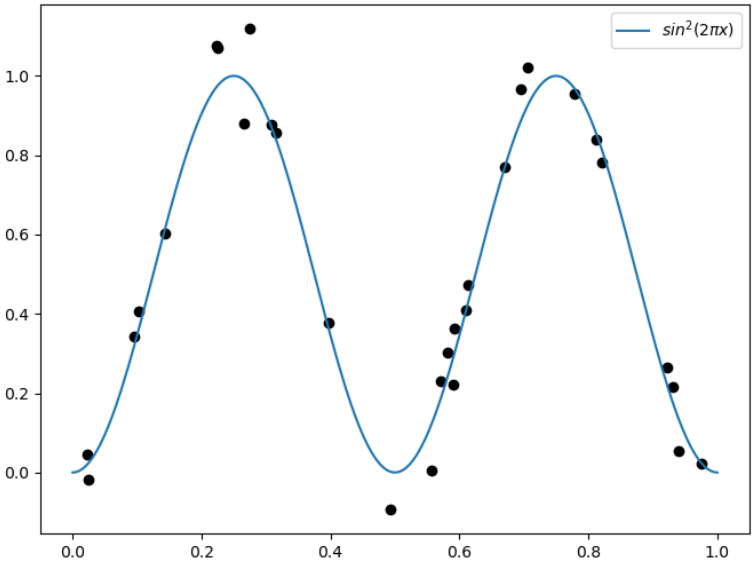

Figure 2: plot of $sin^2(2\pi x)$ and the dataset

(ii) Fit the data set with a polynomial bases of dimension k = 2, 5, 10, 14, 18 plot each of those 5 curves

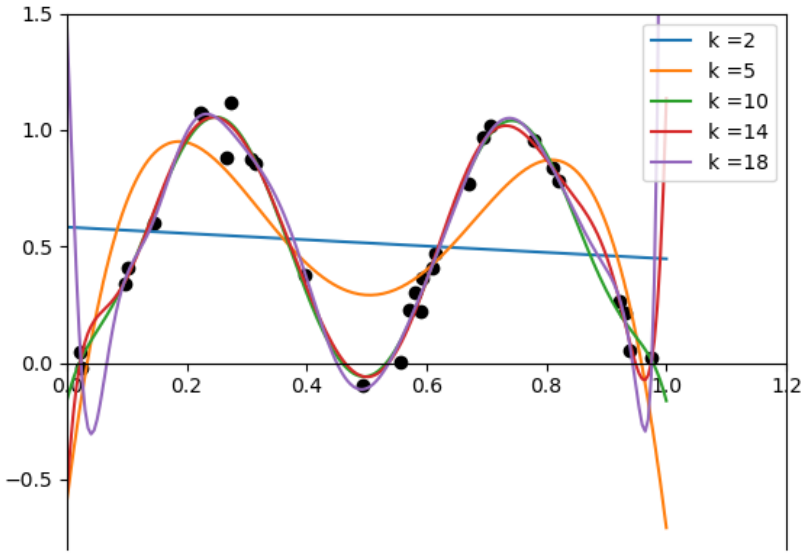

Figure 3: plot of the curve for different k

(b) Let the training error $te_k(S)$ denote the MSE of the fitting of the data set S with polynomial basis of dimension k. Plot the natural log (ln) of the training error versus the polynomial dimension k = 1…18 (this should be a decreasing function).

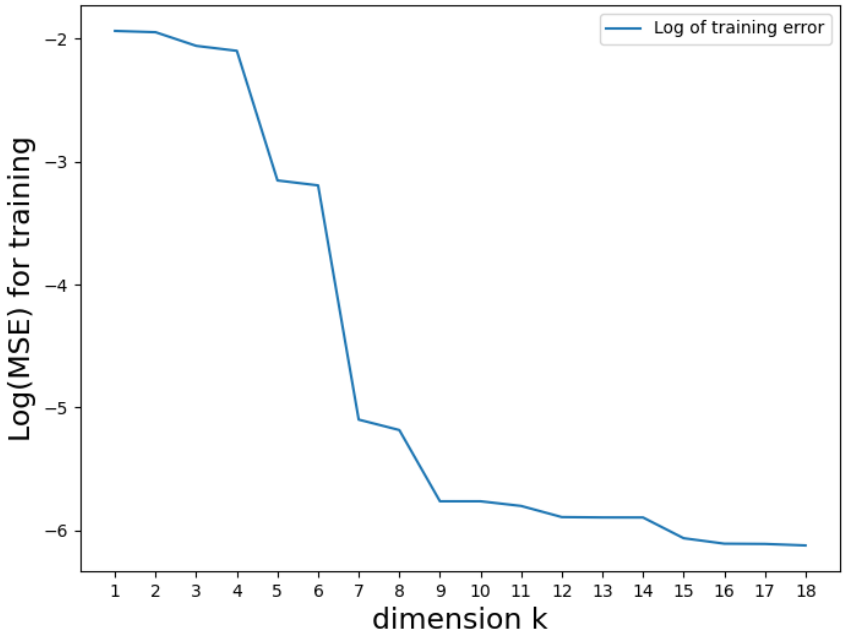

Figure 4: the Log (MSE) of the training error

(c) Define the test error to be the MSE of the test set T on the polynomial of

dimension k fitted from training set S. Plot the ln of the test error versus the polynomial dimension k = 1…18. Unlike the training error this is not a decreasing function. This is the phenomena of over fitting. Although the training error decreases with growing k the test error eventually increases since rather than fitting the function, in a loose sense, we begin to fit to the noise.

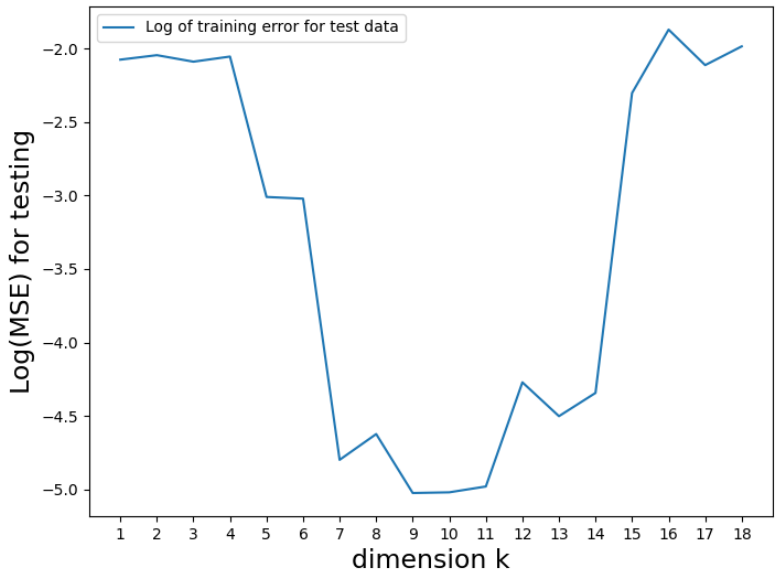

Figure 5: the Log (MSE) of the testing error

(d) For any given set of random numbers we will get slightly different training curves and test curves. It is instructive to see these curves smoothed out. For this part repeat items (b) and (c) but instead of plotting the results of a single \run" plot the average results of a 100 runs (note: plot the ln(avg) rather than the avg(ln)).

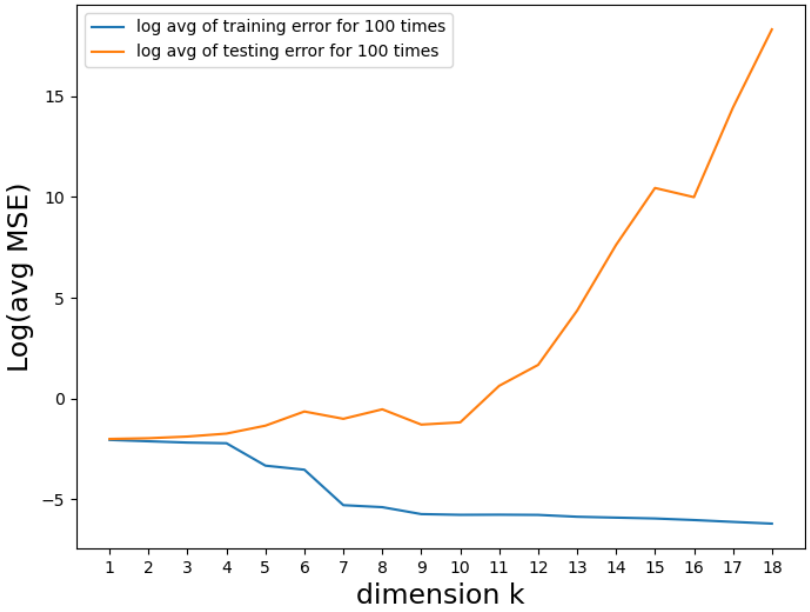

Figure 6: the Log(avg MSE) for training and testing data

Q3

Repeat the experiments in 2 (b-d) with the new basis.

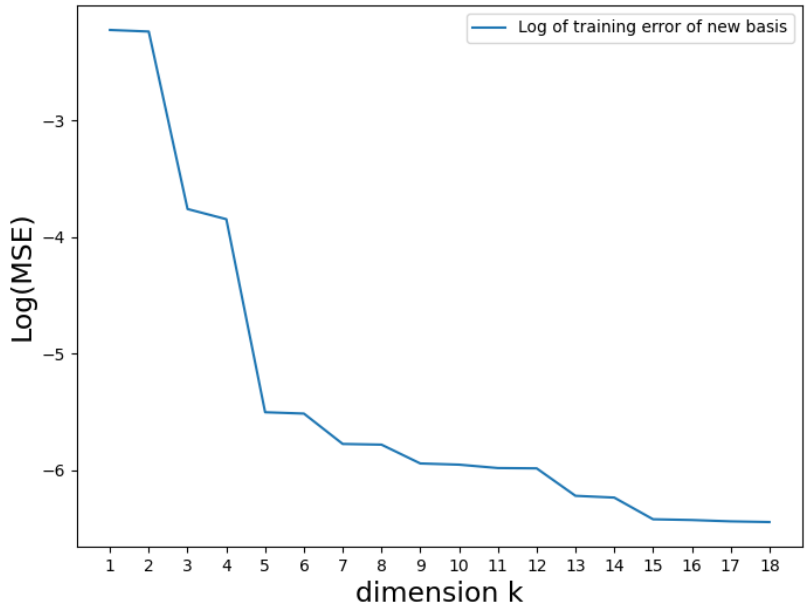

Figure 7: Log(MSE) for the training data

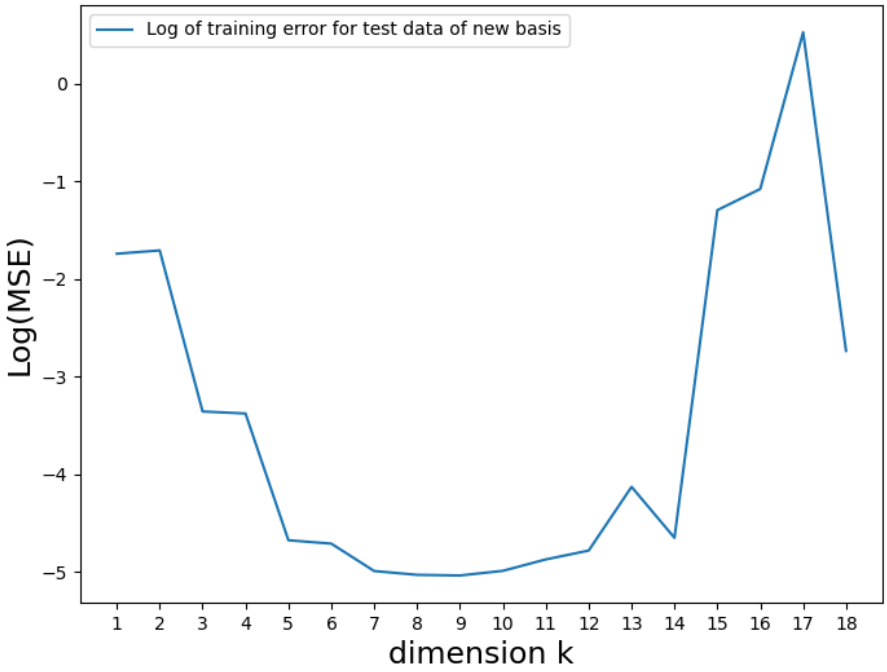

Figure 8: Log (MSE) for the testing data

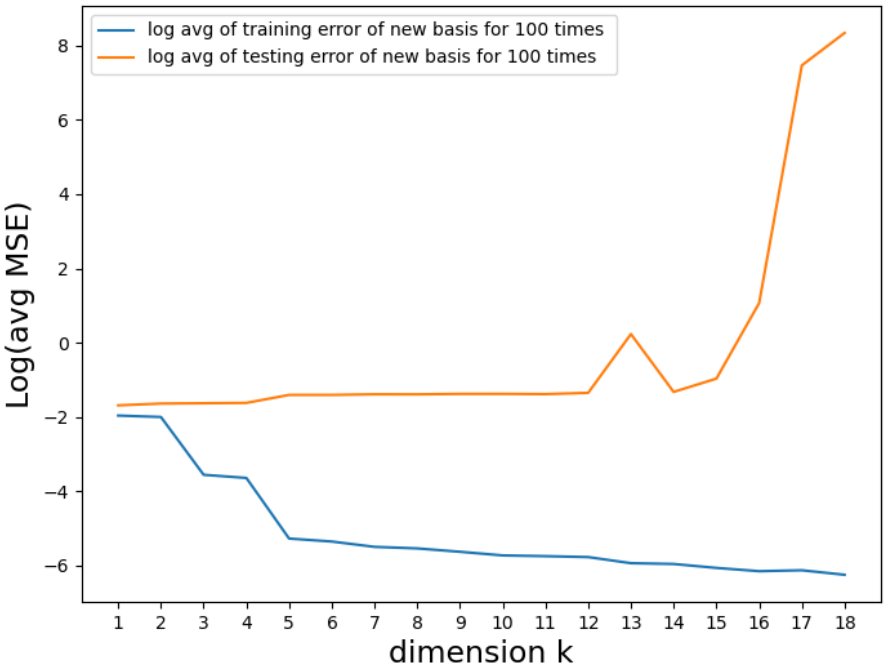

Figure 9: Log (MSE) for the training and testing data

1.2 Filtered Boston housing and kernels

Q4

The training set should be 2/3, and the test set should be 1/3, of your data in (a)-(c). In the following average your results over 20 runs (each run based on a different (2/3,1/3) random split).

(a)

MSE for training data over 20 runs is 85.38330939781102
MSE for testing data over 20 runs is 82.67584590568484

(b) The Naïve Regression is just calculate the average value of the y in training set.

(c)

Attribute	MSE for train	MSE for test
1	72.7286	70.1138
2	73.6455	73.4163
3	63.4822	67.5735
4	80.6226	84.9862
5	70.3456	66.6600
6	42.3184	46.8781
7	72.4030	72.9504
8	77.6646	82.4980
9	71.6125	73.4868
10	66.3308	65.5842
11	63.2681	61.8816
12	38.1715	39.4049

(d)

MSE for training data over 20 runs is22.562217523362087
MSE for testing data over 20 runs is23.543928728212734

1.3 Kernelized ridge regression

Q5

(a) Perform kernel ridge regression on the training set using five-fold cross validation to choose among all pairing of the values of γ and σ. Choose the γ and σ values that perform the best to compute the predictor (by then retraining with those parameters on the training set) that you will use to report the test and training error.

The best gamma is 2^{-35}
The best sigma is 2^{9}

(b) Plot the cross-validation error.

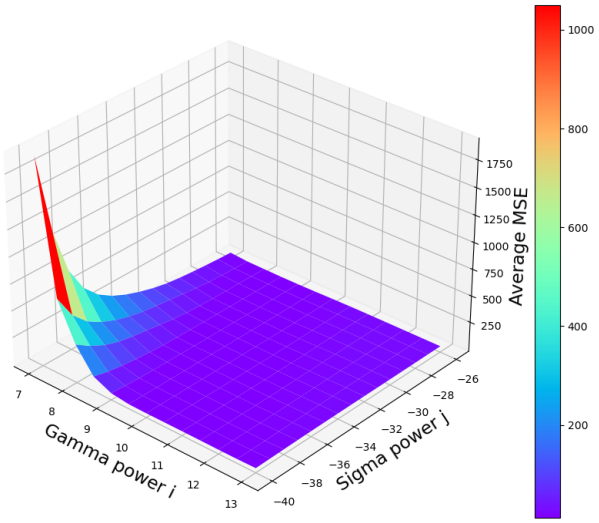

Figure 10: plot of cross-validation error

(c) Calculate the MSE on the training and test sets for the best gamma and sigma

The best training MSE = 7.9571730050770535
The best test MSE = 12.899159683693378

(d) Repeat the following steps to get the table.

Attribute	MSE for train	MSE for test
Naïve Regression	85.0600 ± 4.4864	83.4957 ± 9.0835
Linear Regression (attribute 1)	72.1793 ±4.6123	71.4100 ± 9.1012
Linear Regression (attribute 2)	71.9026 ± 5.4638	76.9602 ± 11.0616
Linear Regression (attribute 3)	65.2964 ± 5.3986	63.8577 ± 10.8190
Linear Regression (attribute 4)	82.5649 ± 4.1486	80.8566 ±8.2403
Linear Regression (attribute 5)	68.9414 ± 4.3789	69.4597 ± 8.6426
Linear Regression (attribute 6)	44.1005 ± 3.8011	43.0819 ± 7.7586
Linear Regression (attribute 7)	74.4355 ± 4.9211	68.7653 ± 9.8331
Linear Regression (attribute 8)	80.6015 ± 4.1222	76.6131 ± 8.2285
Linear Regression (attribute 9)	70.7689 ± 3.4724	75.2986 ± 7.3063
Linear Regression (attribute 10)	65.2231 ± 3.7258	67.5921 ± 7.5193
Linear Regression (attribute 11)	63.1718 ± 3.2797	61.9150 ± 6.4788
Linear Regression (attribute 12)	38.4935 ± 2.3663	38.7248 ± 4.7274
Linear Regression (all attributes)	22.2372 ± 2.1309	24.4497 ± 4.7029
Kernel Ridge Regression	7.8319 ± 1.1701	12.4688 ± 3.2224

Part II

2.1 k-Nearest Neighbors

Q6

Produce a visualization of similar to the figure

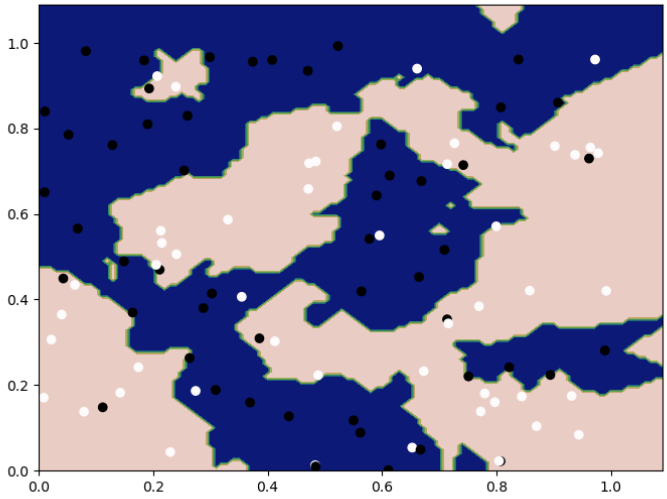

Figure 11: k-Nearest Neighbors visualization

Q7

2.1.2 Estimated generalization error of k-NN as a function of k

Produce a visualization using Protocol A

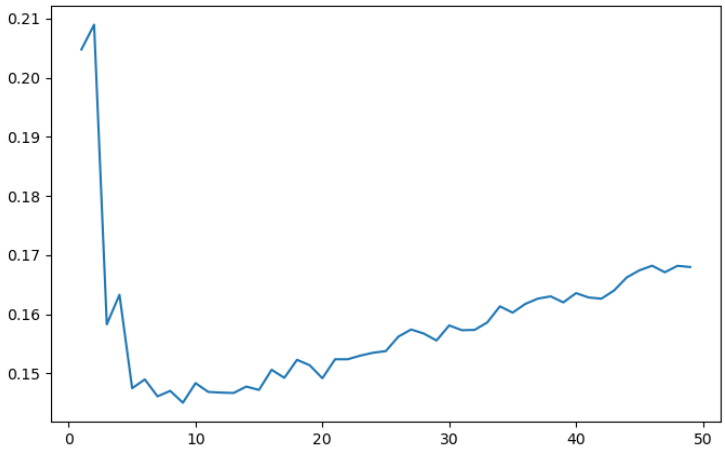

Figure 12: Generalization error against the k

According to the figure 12, we can firstly observe that the average generalization error decreases. Because at the beginning stage, the larger k would lead to fit the data. However, when the k is around 10, the average generalization error starts to increase because increasing the k would cause overfit, which would generate more error.

2.1.3 Determine the optimal k as a function of the number of training points (m)

Produce a visualization using Protocol B.

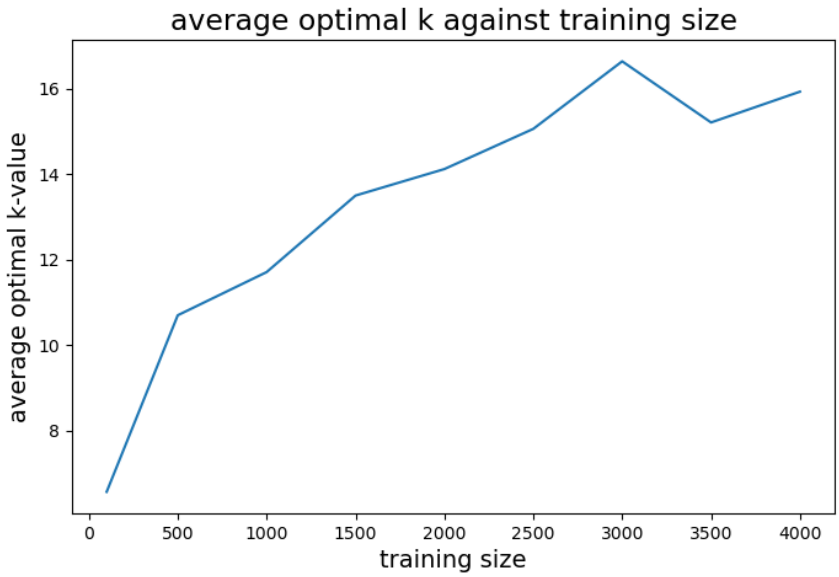

Figure 13: Optimal k value against the number of neighbors

This value of the optimal k increases at the small training size, if the size of the training set is small enough, the optimal k would be 1 or 2. However, as the training size increases, the value of the optimal k remains stable. It means that the value of k is enough and larger k values are trivial.

PART III

3.1 Questions

Q9

(a) For what value of c that K_c a positive semi-definite kernel?

Firstly, according to the Positive Semidefinite Kernel (PSD), for function K: $R^n \times R^n \rightarrow R$ is a positive semidefinite if and only if $K(x,t) = <\phi(x), \phi(t)>$, $x, t \in R^n$. To prove K is the PSD kernel, the $\sum_i^n \sum_j^n K(x_i, x_j) \geq 0$:

$$\sum_i^n \sum_j^n a_i a_j K(x_i, x_j) = \sum_i \sum_j a_i a_j (c + x_i^T x_j)$$

$$= \sum_i \sum_j a_i a_j c + \sum_i \sum_j a_i a_j (x_i^T x_j)$$

$$= \| a \|^2 c + \sum_i \sum_j a_i x_i^T x_j a_j$$

$$= \| a \|^2 c + \| \sum_i a_i x_i \|^2$$

As the proof shown above, $\| a \|^2 c$ and $\| \sum_i a_i x_i \|^2$ should be both larger or equal to 0, the K_c could be PSD kernel. It is obvious that the $\| a \|^2$ and $\| \sum_i a_i x_i \|^2$ are both larger or equal to 0. However, to make sure $\| a \|^2 c + \| \sum_i a_i x_i \|^2$ larger of equal to 0, the c must be larger or equal to 0.

(b) Suppose we use K_c as a kernel function with linear regression. Explain how c influence the solution.

For the linear regression problem, the dual optimization problem after kernelization could be shown as follow:

$$\alpha^* = \underset{\alpha\, \mathcal{E}\, R^\ell}{\operatorname{argmin}} \frac{1}{\ell} \sum_{i=1}^{\ell} (\sum_{j=1}^{\ell} a_j K_{c\,i,j} - y_i)^2$$

$$= \underset{\alpha\, \mathcal{E}\, R^\ell}{\operatorname{argmin}} \frac{1}{\ell} \sum_{i=1}^{\ell} (\sum_{j=1}^{\ell} a_j (c + K_{i,j}) - y_i)^2$$

$$= \underset{\alpha\, \mathcal{E}\, R^\ell}{\operatorname{argmin}} \frac{1}{\ell} \sum_{i=1}^{\ell} (\sum_{j=1}^{\ell} a_j c + \sum_{j=1}^{\ell} a_j K_{i,j} - y_i)^2$$

$$= \underset{\alpha\, \mathcal{E}\, R^\ell}{\operatorname{argmin}} (\frac{1}{\ell} \sum_{i=1}^{\ell} (\sum_{j=1}^{\ell} a_j K_{i,j} - y_i)^2 + c^2 (\sum_{j=1}^{\ell} a_j)^2 + 2c \left(\sum_{j=1}^{\ell} a_j \right) \frac{1}{\ell} \sum_{i=1}^{\ell} (\sum_{j=1}^{\ell} a_j K_{i,j} - y_i))$$

According to the proof above, it is obvious that the second and the third part of the optimization problem is two functions about the c. When c is equal to 0, the solution is the normal linear regression solution. However, when c is larger than 0, the second and third part could be treated as complex hypothesis class. The solution space for this problem would be smaller with the existence of c.

Q10

For the linear regression with a Gaussian kernel $K_\beta(x,t) = \exp(-\beta \parallel x - t \parallel^2)$, and the f is of the form $f(t) = \sum_{i=1}^{m} \alpha_i K_\beta(x_i, t)$. For the vector α could be derived as follow:

$$\alpha = K_\beta(x,x)^{-1} y$$

The $K_\beta(x,x)$ could be displayed as follow:

$$K_\beta(x,x) = \begin{bmatrix} 1 & K_\beta(x_1,x_2) & \cdots & K_\beta(x_1,x_m) \\ K_\beta(x_2,x_1) & 1 & \cdots & K_\beta(x_2,x_m) \\ \vdots & \vdots & \ddots & \vdots \\ K_\beta(x_m,x_1) & K_\beta(x_m,x_2) & \ldots & 1 \end{bmatrix}$$

The inverse of the $K_\beta(x,x)$ could be displayed as follow:

$$K_\beta(x,x)^{-1} = constant \begin{bmatrix} 1 & \cdots & k_{1(m-1)} & k_{1m} \\ \vdots & 1 & \ldots & \vdots \\ k_{(m-1)1} & \ldots & \ddots & \vdots \\ k_{m1} & \cdots & k_{m\,(m-1)} & 1 \end{bmatrix}$$

By combining those two equations above, the vector α could be derived as follow:

$$\alpha = K_\beta(x,x)^{-1}y = constant \begin{bmatrix} 1 & \cdots & k_{1(m-1)} & k_{1m} \\ \vdots & 1 & \ldots & \vdots \\ k_{(m-1)1} & \ldots & \ddots & \vdots \\ k_{m1} & \cdots & k_{m\,(m-1)} & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

$$= constant \begin{bmatrix} y_1 + \cdots + y_m k_{1m} \\ \vdots \\ \vdots \\ y_1 k_{m1} + \cdots + y_m \end{bmatrix}$$

According to the formula of the Gaussian kernel, $\| x - t \|^2$ should be larger or equal to 0. If β increases, the $K_\beta(x_i, t)$ would be converge to 0. What's more, the α would also converge to 0. The f(t) could be displayed as follow:

$$f(t) = constant \begin{bmatrix} y_1 + \cdots + y_m k_{1m} \\ \vdots \\ \vdots \\ y_1 k_{m1} + \cdots + y_m \end{bmatrix} \begin{bmatrix} K_\beta(x_1, t) \\ K_\beta(x_2, t) \\ \vdots \\ K_\beta(x_m, t) \end{bmatrix}$$

This equation could be rewritten as shown below:

$$f(t) = \left(K_\beta(x_1,t) + k_{21}K_\beta(x_2,t) + \cdots + k_{m1}K_\beta(x_m,t)\right) y_1$$

$$+ \left(k_{12}K_\beta(x_1,t) + K_\beta(x_2,t) \cdots + k_{m2}K_\beta(x_m,t)\right) y2$$

$$\vdots$$

$$+ \left(k_{1m}K_\beta(x_1,t) + K_{2m}(x_2,t) \cdots + K_\beta(x_m,t)\right) y_m$$

For the 1-NN classifier, the closest point x_p should meet:

$$\| x_p - t \|^2 < \| x_i - t \|^2 \rightarrow K_\beta(x_p, t) > K_\beta(x_i, t)$$

According to the f(t), if the β is large enough, the $K_\beta(x_i, t)$ would be converge to 0 which was discussed previously, to make sure the y value for the point p be the main factor to predict, the $\beta \rightarrow \infty$ and the $f(t) \rightarrow \sum_{i=1}^{m} y_i K_\beta(x_i, t)$.
As a result, the function $\beta = \hat{\beta}\,(x_1, \cdots x_m, t)$ should be large enough to make sure that $f(t) \rightarrow \sum_{i=1}^{m} y_i K_\beta(x_i, t)$. As a result, any value larger than it could enable the trained linear classifier to simulate a 1-NN classifier.

Q11

Firstly, the matrix X should be defined, where 1 means the mole appear on the corresponding hole and 0 means the mole disappear. The matrix could be shown as follow:

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix}$$

For this problem, the aim is to make a null matrix, which means all the values in the matrix are 0:

$$f(X) = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix}$$

To clarify the algorithm, a simple 3×3 example would be examined. The X is defined as follow:

$$X = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

If the central position is chosen, the X would be a null matrix for just one time. According to the definition as given in the problem, the one hit operation could also be displayed in matrix form:

$$Hit_{22} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

The Hit_1 matrix means the position x_{22} is hit and as a result the x_{02}, x_{12}, x_{22}, x_{23}, x_{32} change from 1 to 0. The logic for calculating could be shown as follow:

$$1 + 1 = 0 + 0 = 0(disappear)$$
$$0 + 1 = 1 + 0 = 1\ (appear)$$

The whole process above could be written in the equation as shown below:

$$X + Hit_{22} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

However, if there are some numbers of hit, the process could be described as follow:

$$X + \sum_{i,j} k_{ij} H_{ij} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

For the $n \times n$ case, the whole process could be displayed:

$$X + \sum_{i,j} k_{ij} H_{ij} = \begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix}_{n \times n}$$

It could be rewritten:

$$\begin{bmatrix} h_{11,11} & \cdots & h_{nn,11} \\ \vdots & \ddots & \vdots \\ h_{11,nn} & \cdots & h_{nn,nn} \end{bmatrix} \begin{bmatrix} k_{11} \\ k_{12} \\ k_{13} \\ \vdots \\ k_{31} \\ k_{32} \\ k_{33} \end{bmatrix} = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ \vdots \\ x_{31} \\ x_{32} \\ x_{33} \end{bmatrix}$$

For the solution, all the k_{ij} should be solved to get the number of hitting for the corresponding position. The complexity for this problem is $O(n^6)$.