COMP0087 Supervised Learning Coursework 1

Ho Yin Chiu 12011552
Hou Nam Chiang 15055142

15th November, 2020

Question 1

(a) Plot of fitted curves:

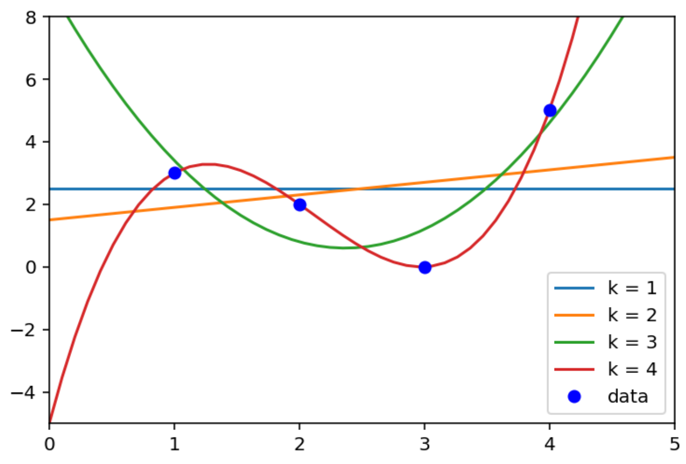

Figure 1: Plot of fitted curves to given data points

(b) Equations corresponding to the curves for $k = 1, 2, 3$:
$k = 1: \quad y = 2.5$
$k = 2: \quad y = 1.5 + 0.4x$
$k = 3: \quad y = 9 - 7.1x + 1.5x^2$

(c) For each of fitted curve $k = 1, 2, 3, 4$, the mean square error (MSE) is:

$$MSE = \frac{1}{m}(\mathbf{Xw} - \mathbf{y})^T(\mathbf{Xw} - \mathbf{y})$$

$k = 1: \quad \text{MSE} = 3.25$
$k = 2: \quad \text{MSE} = 3.05$
$k = 3: \quad \text{MSE} = 0.8$
$k = 4: \quad \text{MSE} = 0$

Question 2

(a) i. Plot of $g_{0.07}(x)$ for 30 points

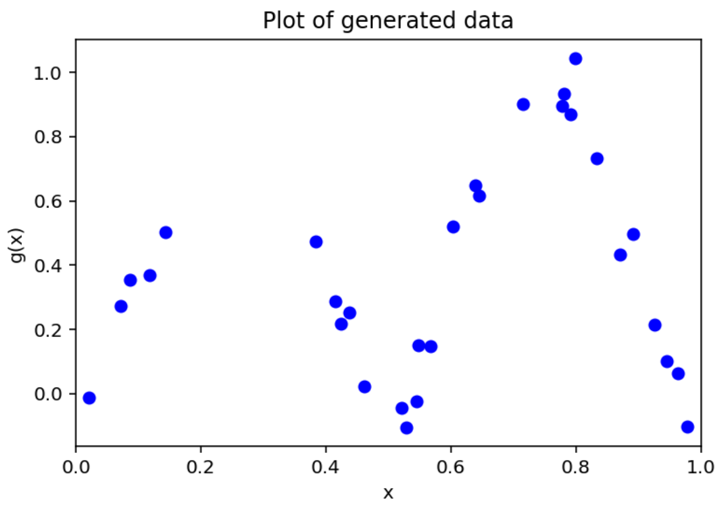

Figure 2: $g_{0.07}(x)$ in the range $0 \leq 0 \leq 1$ for 30 points.

ii. Fit of the data set with a polynomial bases of dimension $k = 2, 5, 10, 14, 18$. The fitted curves are plotted superimposing the data points:

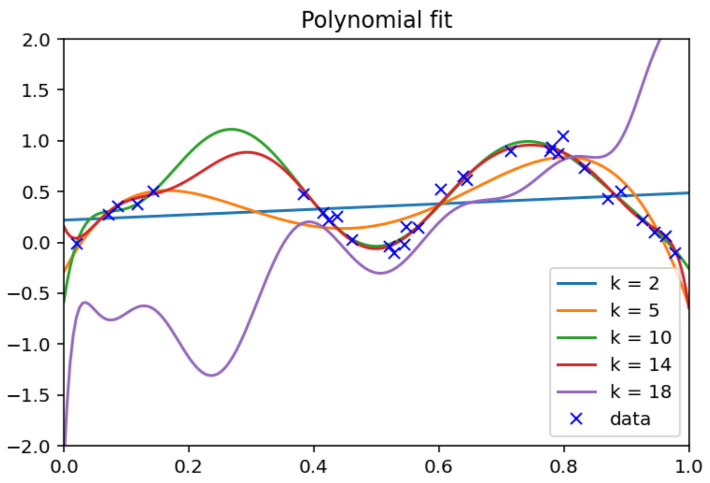

Figure 3: Fitted curves for various k with data points.

(b) Plot of the natural log of the training error versus the polynomial dimension $k = 1, \ldots, 18$:

$$ln(MSE) = ln\left[\frac{1}{m}(\mathbf{Xw} - \mathbf{y})^T(\mathbf{Xw} - \mathbf{y})\right]$$

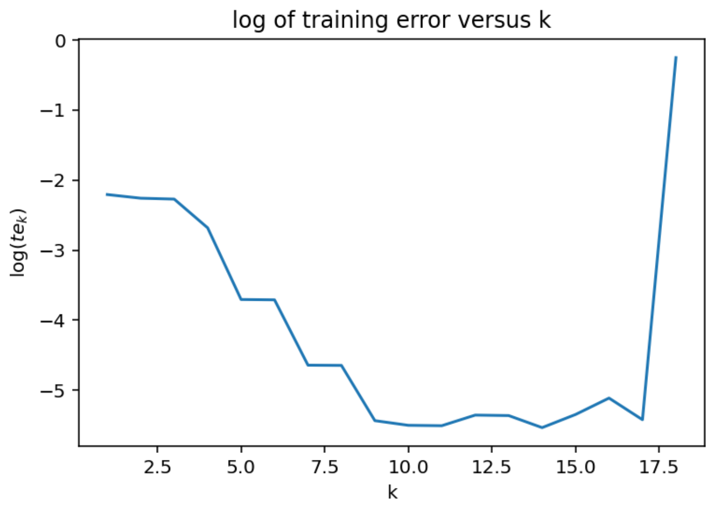

Figure 4: Natural log of training error versus polynomial dimension k.

(c) Plot of the natural log of the testing error versus the polynomial dimension $k = 1, \ldots, 18$:

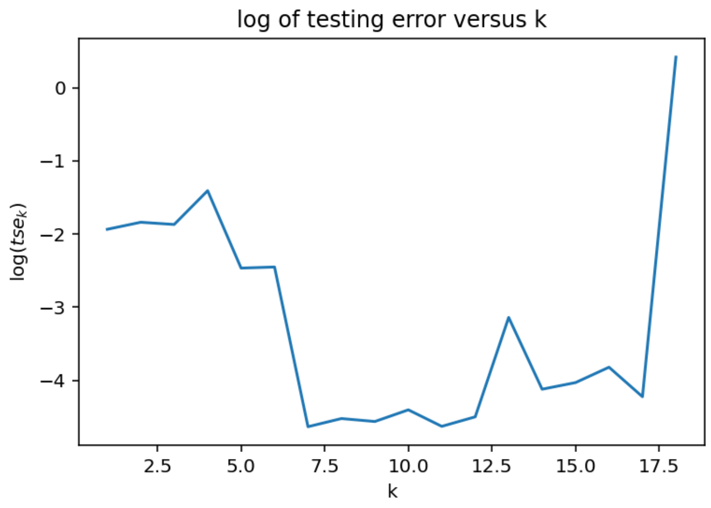

Figure 5: Natural log of testing error versus polynomial dimension k.

(d) Average results of 100 runs for training and testing error

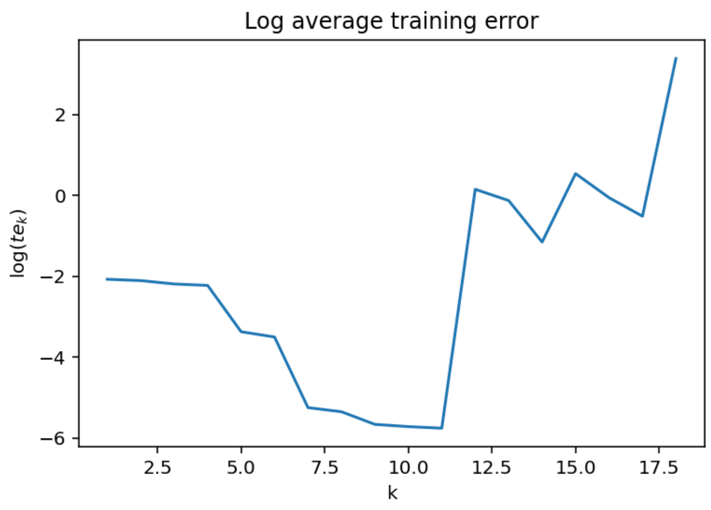

Figure 6: Natural log of average training error versus polynomial dimension k.

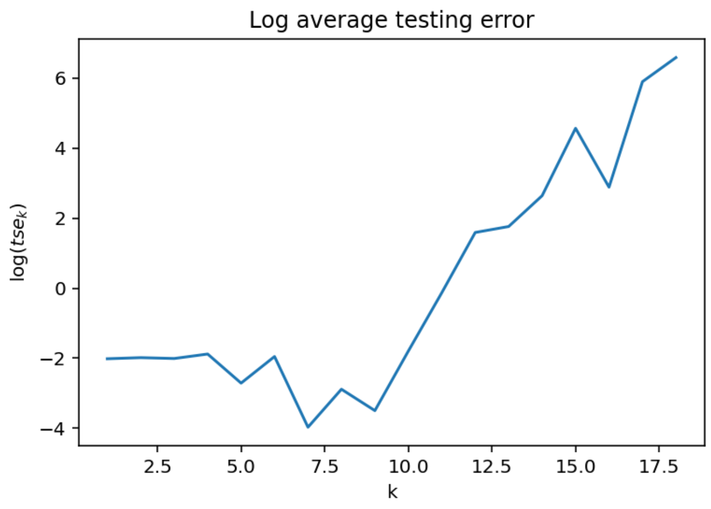

Figure 7: Natural log of average testing error versus polynomial dimension k.

Question 3

Repeat question 2 (b-d) but with basis

$$\sin(1\pi x)\sin(2\pi x)\ldots\sin(\pi x)$$

The plots are shown below:

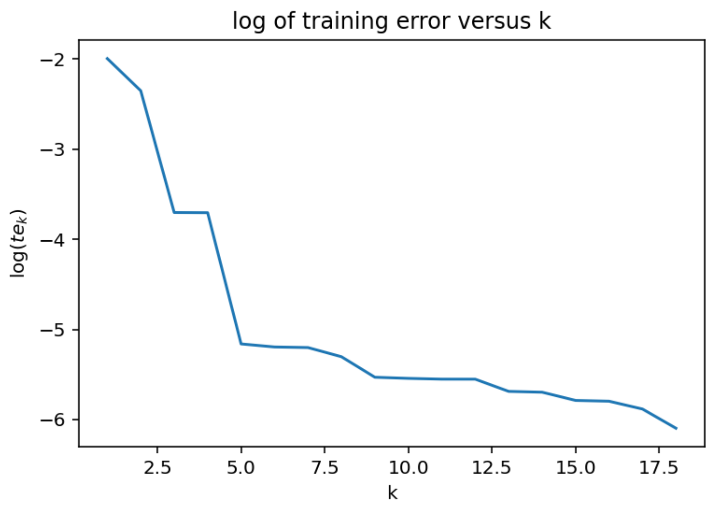

Figure 8: Natural log of training error versus polynomial dimension k.

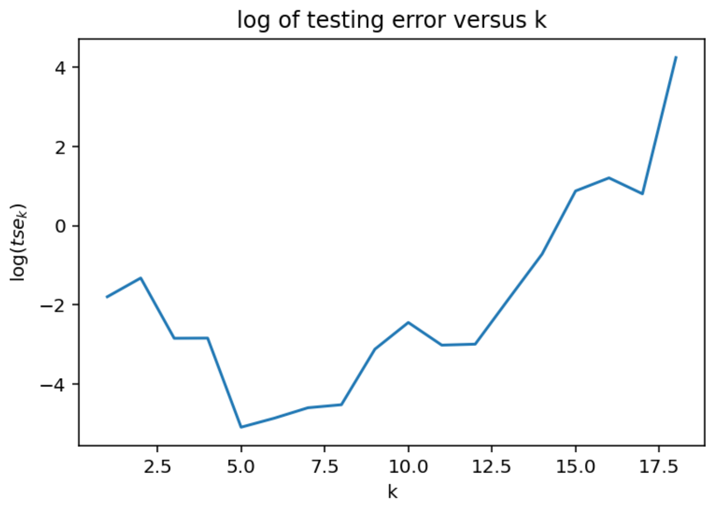

Figure 9: Natural log of testing error versus polynomial dimension k.

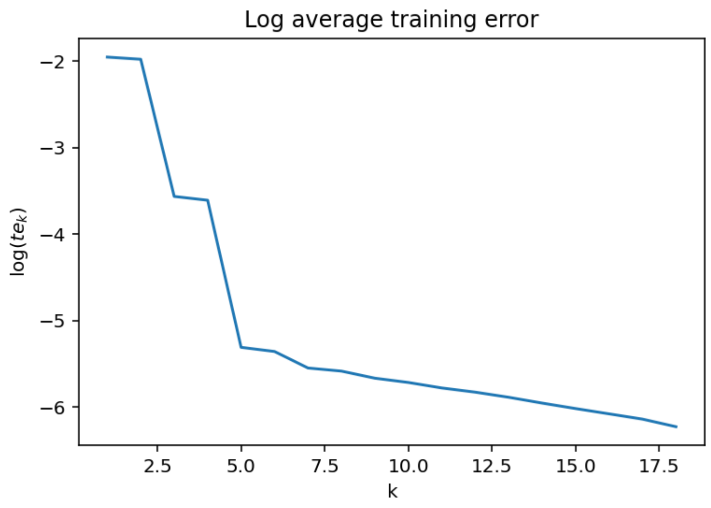

Figure 10: Natural log of average training error versus polynomial dimension k.

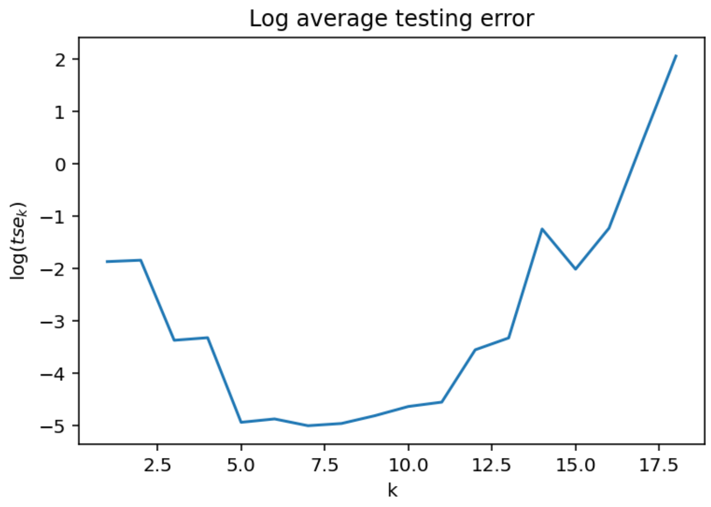

Figure 11: Natural log of average testing error versus polynomial dimension k.

Question 4

The data is split into 2/3 for training set and 1/3 for test set. Then we calculate the MSE averaged over 20 runs for naive regression, linear regression on single attribute and linear regression on all attributes. We do this on both training set and test set, then we show the results.

(a) Naive regression:
MSE on train = 84.118
MSE on test = 85.290

(b) Simple interpretation of the constant function:
By using the simple transformation of feature of x_i to just 1, we throw away information for the linear regression to learn. Therefore the best prediction is to predict with the average y value every time, which is equivalent to predicting with a constant function.

(c) Linear regression on single attributes:
MSE on train = 66.229
MSE on test = 66.813

(d) Linear regression on all attributes:
MSE on train = 21.663
MSE on test = 25.503

Question 5

(a) Best γ and σ in for a particular run with cross validation is shown in the code.

(b) Plot of cross-validation error as a function of γ and σ in log scale:

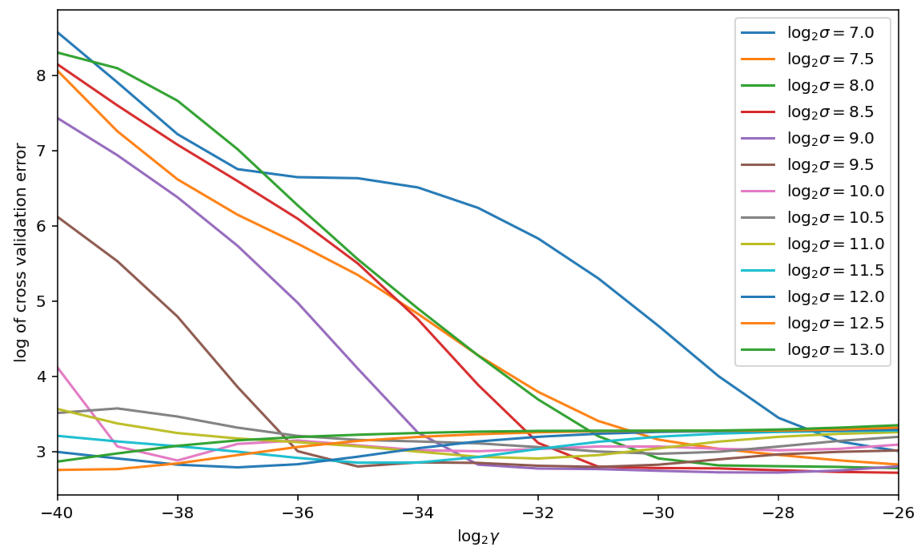

Figure 12: Cross validation error as a function of γ and σ in log scale.

(c) MSE on the training and test set for the best γ and σ:
Training set MSE = 9.111
Test set MSE = 10.078

(d) The training for question 4 and 5 has been repeated and the overall MSE results for training and test set are summarised in the below table:

Method	**MSE train**	**MSE test**
Naive Regression	84.12 ± 5.52	85.29 ± 10.86
Linear Regression (attributes 1)	71.70 ± 3.69	73.01 ± 7.74
Linear Regression (attributes 2)	74.08 ± 4.48	72.52 ± 8.92
Linear Regression (attributes 3)	64.93 ± 4.09	64.52 ± 8.25
Linear Regression (attributes 4)	80.54 ± 5.67	85.17 ± 11.49
Linear Regression (attributes 5)	67.33 ± 4.36	72.79 ± 8.88
Linear Regression (attributes 6)	43.92 ± 2.72	43.41 ± 5.27
Linear Regression (attributes 7)	73.93 ± 5.87	69.87 ± 11.60
Linear Regression (attributes 8)	79.20 ± 4.23	79.49 ± 8.51
Linear Regression (attributes 9)	72.75 ± 5.17	71.34 ± 10.24
Linear Regression (attributes 10)	66.25 ± 2.89	65.57 ± 5.75
Linear Regression (attributes 11)	62.16 ± 3.81	64.17 ± 7.65
Linear Regression (attributes 12)	37.97 ± 2.09	39.90 ± 4.42
Linear Regression (all attributes)	21.66 ± 2.50	25.50 ± 5.40
Kernel Ridge Regression	7.80 ± 1.28	12.62 ± 2.50

Question 6

(a) The loss function is

$$L_c(y, \hat{y}) = [y \neq \hat{y}]c_y$$

To find the Bayes estimator, we want to find $\hat{y}$ such that it minimises $E[L_c]$. Let the Bayes estimator be $\hat{y}^*$, then

$$\begin{aligned}
\hat{y}^* &= \underset{\hat{y}}{\operatorname{argmin}}\, E[L_c(y, \hat{y})] \\
&= \underset{\hat{y}}{\operatorname{argmin}} \sum_{x \in X, y \in Y} [y \neq \hat{y}]c_y p(x, y) \\
&= \underset{\hat{y}}{\operatorname{argmin}} \sum_{x \in X} \sum_{y \in Y} [y \neq \hat{y}]c_y p(y|x)p(x)
\end{aligned}$$

At a specific point $x = x'$, we have

$$\begin{aligned}
\hat{y}^*(x') &= \underset{\hat{y}(x')}{\operatorname{argmin}} \sum_{y \in Y} [y \neq \hat{y}(x')]c_y p(y|x')p(x') \\
&= \underset{\hat{y}(x')}{\operatorname{argmin}} \sum_{y \in Y} [y \neq \hat{y}(x')]c_y p(y|x')
\end{aligned}$$

Since $\hat{y}(x') \in \{1, 2, \ldots, k\}$ and it can only be one of those, to minimise the sum we need to avoid the largest contribution in the sum, so we have

$$\hat{y}^*(x') = \underset{y}{\operatorname{argmax}}\, c_y p(y|x')$$

and we have the Bayes estimator

$$\hat{y}^*(x) = \underset{y}{\operatorname{argmax}}\, c_y p(y|x)$$

(b) The F-loss is

$$L_F(y, \hat{y}) := F(\hat{y}) - F(y) + (y - \hat{y})F'(y)$$

i. When $F(x) = x^2$, the F-loss is

$$\begin{aligned}
L_F(y, \hat{y}) &= \hat{y}^2 - y^2 + 2y(y - \hat{y}) \\
&= \hat{y}^2 - y^2 + 2y^2 - 2\hat{y}y \\
&= \hat{y}^2 - 2\hat{y}y + y^2 \\
&= (y - \hat{y})^2
\end{aligned}$$

which is the usual squared loss.

ii. When $y = \hat{y}$ and substitute this into the definition of F-loss, we have,

$$L_F(y, y) = F(y) - F(y) + (y - y)F'(y) = 0$$

So, $y = \hat{y} \Rightarrow L_F(y, \hat{y}) = 0$.
Now, when $L_F(y, \hat{y}) = 0$, from the definition of F-loss we have,

$$0 = F(\hat{y}) - F(y) + (y - \hat{y})F'(y)$$

differentiating both sides with respect to y

$$\begin{aligned} -F'(y) + F'(y) + (y - \hat{y})F''(y) &= 0 \\ (y - \hat{y})F''(y) &= 0 \end{aligned}$$

Since F is strictly convex, we know that $F''(y)$ is strictly positive, then we arrive at

$$y = \hat{y}$$

So we also have $L_F(y, \hat{y}) = 0 \Rightarrow y = \hat{y}$, proving $y = \hat{y} \Leftrightarrow L_F(y, \hat{y}) = 0$.

iii. At the minimum point of $L_F(y, \hat{y})$,

$$\frac{\partial L_F(y, \hat{y})}{\partial y} = \frac{\partial L_F(y, \hat{y})}{\partial \hat{y}} = 0$$

Then using the definition of F-loss, at the minimum we have

$$\frac{\partial L_F(y, \hat{y})}{\partial y} = -F'(y) + F'(y) + (y - \hat{y})F''(y) = (y - \hat{y})F''(y) = 0$$

$$\frac{\partial L_F(y, \hat{y})}{\partial \hat{y}} = F'(\hat{y}) - F'(y) = 0$$

Therefore at minimum we have the relations

$$\begin{gathered} y = \hat{y} \\ F'(y) = F'(\hat{y}) \end{gathered} \tag{1}$$

where we have used $F''(y) \geq 0$ since F is strictly convex.
Now we want to test whether $L_F(y, \hat{y})$ is convex or not at the minimum, so we proceed to compute the Hessian at minimum. We have,

$$\frac{\partial^2 L_F}{\partial \hat{y} \partial y} = \frac{\partial^2 L_F}{\partial y \partial \hat{y}} = -F''(y)$$

$$\frac{\partial L_F^2}{\partial y^2} = F''(y) + (y - \hat{y})F'''(y)$$

$$\frac{\partial L_F^2}{\partial \hat{y}^2} = F''(\hat{y})$$

Then by substituting the relation (1) to the Hessian at minimum, we have

$$\frac{\partial^2 L_F}{\partial \hat{y} \partial y} = \frac{\partial^2 L_F}{\partial y \partial \hat{y}} = -F''(y)$$

$$\frac{\partial L_F^2}{\partial y^2} = F''(y)$$

$$\frac{\partial L_F^2}{\partial \hat{y}^2} = F''(y)$$

Hence the Hessian matrix at the minimum of L_F is

$$H = \begin{pmatrix} F''(y) & -F''(y) \\ -F''(y) & F''(y) \end{pmatrix} = F''(y) \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Computing the product $\mathbf{v}^T H \mathbf{v}$ for an arbitrary vector $\mathbf{v} = (v_1, v_2)^T$, we have

$$\begin{aligned} & \mathbf{v}^T H \mathbf{v} \\ =& F''(y) \begin{pmatrix} v_1 & v_2 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \\ =& F''(y)(v_1^2 - 2v_1v_2 + v_2^2) \\ =& F''(y)(v_1 - v_2)^2 \\ \geq& 0 \end{aligned}$$

since $F''(y) > 0$ and $(v_1 - v_2)^2 \geq 0$.
Hence, the H is positive semidefinite, this implies that L_F is convex at its minimum point. As we have already found that at minimum $y = \hat{y}$ and we know from question (b) ii. that at $y = \hat{y}$, we have $L_F = 0$, it follows that L_F is at the global minimum when $L_F = 0$, therefore we arrive at

$$L_F(y, \hat{y}) \geq 0$$

iv. Expected F-loss is

$$\begin{aligned}
&E[L_F(y,\hat{y})] \\
=&E[F(\hat{y}) - F(y) + (y-\hat{y})F'(y)] \\
=&\sum_{x\in X}\sum_{y\in Y}(F(\hat{y}) - F(y) + (y-\hat{y})F'(y))p(y|x)p(x)
\end{aligned}$$

We want to find the minimum of expected F-loss at a specific point, then let the point be $x = x'$, and we have

$$E[L_F(y,\hat{y}(x'))] = \sum_{y\in Y}(F(\hat{y}(x')) - F(y) + (y-\hat{y}(x'))F'(y))p(y|x')p(x')$$

To find the Bayes estimator, we find $\hat{y}(x')$ such that it minimises the expected F-loss at x'. Let $z = \hat{y}(x')$, then we have

$$\begin{aligned}
&\operatorname*{argmin}_z E[L_F(y,z)] \\
=&\operatorname*{argmin}_z \sum_{y\in Y}(F(z) - F(y) + (y-z)F'(y))p(y|x')
\end{aligned}$$

At the minimum point, the derivative of the above expression is 0, then we differentiate the above expression with respect to z and set it to zero, we have

$$\begin{aligned}
\frac{\partial}{\partial z}\sum_{y\in Y}(F(z) - F(y) + (y-z)F'(y))p(y|x') &= 0 \\
\sum_{y\in Y}(F'(z) - F'(y))p(y|x') &= 0 \\
\sum_{y\in Y}F'(z)p(y|x') &= \sum_{y\in Y}F'(y)p(y|x') \\
F'(z) &= \sum_{y\in Y}F'(y)p(y|x') \\
F'(z) &= E[F'(y)|x']
\end{aligned}$$

Hence, we have the Bayes estimator for the F-loss that satisfies the relation

$$F'(\hat{y}^*(x)) = E[F'(y)|x] \tag{2}$$

where $\hat{y}^*(x)$ is the Bayes estimator at point x.
Consider now the special case $F(x) = |x|^p$, for even $p > 1$, we have

$$F(x) = |x|^p = x^p$$

So,

$$F'(x) = px^{p-1}$$

for p even. Then we have

$$\begin{aligned}
F'(\hat{y}^*(x)) &= p\hat{y}^*(x)^{p-1} \\
F'(y) &= py^{p-1}
\end{aligned}$$

Substitute them back into the Bayes estimator relation (2), we have

$$\begin{aligned} p\hat{y}^*(x)^{p-1} &= E[py^{p-1}|x] \\ \hat{y}^*(x)^{p-1} &= E[y^{p-1}|x] \\ \hat{y}^*(x) &= \sqrt[p-1]{E[y^{p-1}|x]} \end{aligned} \tag{3}$$

For $p = 2$, $F(x) = x^2$, and we have recovered the squared loss as found in question (b) i., and by (3) we also have

$$\hat{y}^*(x) = E[y|x]$$

which is the Bayes estimator for the squared loss that we have also recovered.

Question 7

(a) Considering the function:

$$K_c(\mathbf{x}, \mathbf{z}) := \left[\sum_{i=1}^{n} x_i z_i\right] + c \tag{4}$$

where $\mathbf{x}, \mathbf{z} \in \mathbb{R}^n$.

For K_c to be a positive semidefinite kernel, it must satisfy two requirements:

(a) The kernel is symmetric.

(b) The matrix $K_c(\mathbf{x}, \mathbf{z})$ is positive semidefinite for $\mathbf{x}, \mathbf{z} \in \mathbb{R}^n$.

Expanding (4), and rearranging, we have:

$$\begin{aligned}
K_c(\mathbf{x}, \mathbf{z}) &= x_1 z_1 + x_2 z_2 + ... + x_n z_n + c \\
&= x_1 z_1 + x_2 z_2 + ... + x_n z_n + \sqrt{c}^2 \\
&= \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ \sqrt{c} \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \\ \sqrt{c} \end{bmatrix} \\
&= \langle \mathbf{x}', \mathbf{z}' \rangle
\end{aligned}$$

From the above, we can see that the inner product $\langle \mathbf{x}', \mathbf{z}' \rangle$ must be symmetric such that:

$$\begin{aligned}
K_c(\mathbf{x}, \mathbf{z}) &= \langle \mathbf{x}', \mathbf{z}' \rangle \\
&= \langle \mathbf{z}', \mathbf{x}' \rangle \\
&= x_1 z_1 + x_2 z_2 + ... + x_n z_n + c
\end{aligned}$$

Therefore, the first requirement that K_c must be symmetric is proved.

For the second requirement, for K_c to be positive semidefinite, the below condition must be satisfied by K_c:

$$\sum_{i,j} a_i a_j K_c(\mathbf{x}_i, \mathbf{x}_j) \geq 0$$

for any $a_i \in \mathbb{R}, i, j \in \{1, \dots, m\}$. Then by substituting $K_c(\mathbf{x}_i, \mathbf{x}_j)$, we have

$$\begin{aligned}
\sum_{i,j} a_i a_j \left(c + \sum_k x_{ik} x_{jk} \right) &\geq 0 \\
c \sum_{i,j} a_i a_j + \sum_{i,j} a_i a_j \sum_k x_{ik} x_{jk} &\geq 0 \\
c \left(\sum_i a_i \right) \left(\sum_j a_j \right) + \sum_k \sum_{i,j} a_i x_{ik} a_j x_{jk} &\geq 0 \\
c \left(\sum_i a_i \right)^2 + \sum_k \left(\sum_j a_i x_{ik} \right) \left(\sum_j a_j x_{jk} \right) &\geq 0 \\
c \left(\sum_i a_i \right)^2 + \left\langle \sum_i a_i \mathbf{x}_i, \sum_j a_j \mathbf{x}_j \right\rangle &\geq 0 \\
c \left(\sum_i a_i \right)^2 + \left\| \sum_i a_i \mathbf{x}_i \right\|^2 &\geq 0
\end{aligned}$$

Solve for the inequality for c and we arrive at

$$c \geq -\frac{\left\| \sum_i a_i \mathbf{x}_i \right\|^2}{\left(\sum_i a_i \right)^2}$$

for any $\mathbf{x}_i \in \mathbb{R}^n$ and $a_i \in \mathbb{R}$. This inequality tells us that c must be greater than or equal to the right hand side for K_c to be positive semidefinite. Looking at the right hand side, we know that maximum achievable value of the right hand side given any $\mathbf{x}_i$ and a_i is 0. Since we want K_c to be positive semidefinite for any a_i and $\mathbf{x}_i$, therefore c must satisfy:

$$c \geq 0$$

Therefore when $c \geq 0$ we have a positive semidefinte $K_c(\mathbf{x}, \mathbf{z})$.

(b) Let our kernel function K_c be

$$K_c(\mathbf{x}, \mathbf{z}) = c + K(\mathbf{x}, \mathbf{z})$$

where

$$K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{z}$$

Suppose we want to perform linear regression using K_c, then from equation (11) of the question sheet, by setting $\gamma = 0$, we have the optimisation problem:

$$\begin{aligned}
\boldsymbol{\alpha}^* &= \underset{\boldsymbol{\alpha}\in\mathbb{R}^l}{\operatorname{argmin}} \frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j K_{c\ i,j} - y_i\right)^2 \\
&= \underset{\boldsymbol{\alpha}\in\mathbb{R}^l}{\operatorname{argmin}} \frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j (c + K_{i,j}) - y_i\right)^2 \\
&= \underset{\boldsymbol{\alpha}\in\mathbb{R}^l}{\operatorname{argmin}} \frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j c + \sum_{l=1}^{l}\alpha_j K_{i,j} - y_i\right)^2 \\
&= \underset{\boldsymbol{\alpha}\in\mathbb{R}^l}{\operatorname{argmin}} \left[\frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j K_{i,j} - y_i\right)^2 + c^2\left(\sum_{j=1}^{l}\alpha_j\right)^2 + 2c\left(\sum_{j=1}^{l}\alpha_j\right)\frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j K_{i,j} - y_i\right)\right]
\end{aligned}$$

where we have now the complexity term is

$$\begin{aligned}
&c^2\left(\sum_{j=1}^{l}\alpha_j\right)^2 + 2c\left(\sum_{j=1}^{l}\alpha_j\right)\frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j K_{i,j} - y_i\right) \\
&= c f_1(\boldsymbol{\alpha}) + c^2 f_2(\boldsymbol{\alpha})
\end{aligned}$$

where f_1, f_2 are functions for $\boldsymbol{\alpha}$. Therefore c can be thought of as a regularisation term for $\boldsymbol{\alpha}$ for the given optimisation problem. When $c = 0$, we recover a typical optimisation problem where our kernel is just $K_c(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T\mathbf{z}$. As c increases, the complexity term restricts the size of the hypothesis space for $\boldsymbol{\alpha}$, hence the solution space for our problem will be smaller. As $c \to \infty$, the solution for the optimisation problem is such that $\boldsymbol{\alpha}^* = \mathbf{0}$, and the predicted value will always be $y_{\text{test}} = 0$.

Question 8

Suppose we perform linear regression with a Gaussian kernel

$$K_\beta(\mathbf{x},\mathbf{t}) = \exp(-\beta||\mathbf{x}-\mathbf{t}||^2) \tag{5}$$

Qualitatively, the kernel can be thought of as a Gaussian with its peak situated at location $\mathbf{x}$ in the kernel space. By varying β in the kernel, we can vary the 'spikiness' of the Gaussian. For small β, the Gaussian is very spread out and its 'spikiness' is very small. For large β, the Gaussian squeezes and most of its mass concentrates around $\mathbf{x}$, this means that it 'spikiness' is large. When $\beta \to \infty$, all of its mass will concentrate at $\mathbf{x}$ and it effectively becomes a needle in the kernel space, with maximum 'spikiness'.
If we imagine the case with large β, then for a point $\mathbf{t}$ which is not in the neighbourhood of $\mathbf{x}$, the kernel of them $K_\beta(\mathbf{x},\mathbf{t})$ is vanishing, since $\mathbf{t}$ is nowhere close to the mass of the Gaussian at $\mathbf{x}$.
Mathematically, we can express this as we take the limit of $\beta \to \infty$, then we have

$$K_\beta(\mathbf{x},\mathbf{t}) = \begin{cases} 1 & \mathbf{x}=\mathbf{t} \\ 0 & \mathbf{x}\neq\mathbf{t} \end{cases} \tag{6}$$

for $\beta \to \infty$ and $||\mathbf{x}-\mathbf{t}||^2 \sim \mathcal{O}(1)$. This follows directly by using the Gaussian kernel in (5).
Now, suppose there is another point $\mathbf{t}'$ which is farther away from $\mathbf{x}$ than $\mathbf{t}$ and $||\mathbf{t}'-\mathbf{t}||^2 \sim \mathcal{O}(1)$, which means $||\mathbf{x}-\mathbf{t}||^2 < ||\mathbf{x}-\mathbf{t}'||^2$, then we have the kernels

$$K_\beta(\mathbf{x},\mathbf{t}) \gg K_\beta(\mathbf{x},\mathbf{t}') \tag{7}$$

As $\beta \to \infty$.
This is the case since from the Gaussian kernel (5), we have

$$\exp(-\beta||\mathbf{x}-\mathbf{t}||^2) \gg \exp(-\beta||\mathbf{x}-\mathbf{t}'||^2)$$

when $||\mathbf{x}-\mathbf{t}||^2 < ||\mathbf{x}-\mathbf{t}'||^2$ and $\beta \to \infty$.

Now, consider the model that we learnt from the data $\{\mathbf{x}_1,\mathbf{x}_2,\dots,\mathbf{x}_m\}$ using linear regression, we have the model

$$f(\mathbf{t}) = \sum_{i=1}^{m} \alpha_i K_\beta(\mathbf{x}_i,\mathbf{t})$$

Qualitatively, we can imagine $f(\mathbf{t})$ is being composed of a linear combination of $K_\beta(\mathbf{x}_i,\mathbf{t})$, each weighted by α_i. This means that for large β, the kernel space is filled with many needles sticking out of the $\mathbf{x}$ plane, where each needle is situated at $\mathbf{x}_i$. Then we can imagine when $\mathbf{t}$ is located very close to any $\mathbf{x}_i$, we have a non-vanishing $f(\mathbf{t})$ and a vanishing $f(\mathbf{t})$ otherwise.
However, let us consider the sum at the leading order. In the sum, the dominant term is $K_\beta(\mathbf{x}_k,\mathbf{t})$ when the closest point to $\mathbf{t}$ is $\mathbf{x}_k$. In other words, the dominant kernel is $K_\beta(\mathbf{x}_k,\mathbf{t})$ for $||\mathbf{x}_k-\mathbf{t}||^2 < ||\mathbf{x}_i-\mathbf{t}||^2$, $i \in \{1,2,\dots,m\}\setminus\{k\}$. This implies directly from (7). So when all $\alpha_i \sim \mathcal{O}(1)$ and the closest point to $\mathbf{t}$ is $\mathbf{x}_k$, the linear regression model at the leading order in the limit $\beta \to \infty$ becomes

$$f(\mathbf{t}) = \alpha_k K_\beta(\mathbf{x}_k,\mathbf{t})$$

At prediction for a point $\mathbf{t}$ using the trained model as a classifier, at leading order, we have

$$\begin{aligned}\text{sign}(f(\mathbf{t})) &= \text{sign}(\alpha_k K_\beta(\mathbf{x}_k,\mathbf{t})) \\ &= \text{sign}(\alpha_k)\end{aligned} \tag{8}$$

for $\beta \to \infty$. The last line implies since $K_\beta \geq 0$.

From equation (12) of the question sheet, we have $\boldsymbol{\alpha} = (\mathbf{K} + \gamma l \mathbf{I}_m)^{-1}\mathbf{y}$. In our linear regression case we have $\gamma = 0$. So after rearrangement and setting γ to 0, our $\boldsymbol{\alpha}$ satisfies $\mathbf{y} = \mathbf{K}\boldsymbol{\alpha}$, and we have

$$\begin{aligned} y_i &= \sum_{j=1}^{m} \alpha_j K_\beta(\mathbf{x}_j, \mathbf{x}_i) \\ &= \alpha_i K_\beta(\mathbf{x}_i, \mathbf{x}_i) \\ &= \alpha_i \end{aligned}$$

as $\beta \to \infty$. Where we have used (6). Therefore as $\beta \to \infty$ we have

$$\operatorname{sign}(\alpha_i) = \operatorname{sign}(y_i) \tag{9}$$

Finally, by combining (8) and (9), we arrive at our kernel linear regression model when $\beta \to \infty$

$$\operatorname{sign}(f(\mathbf{t})) = \operatorname{sign}(y_k)$$

for which y_k is the label of the closest point to $\mathbf{t}$.

The final relation tells us that to predict from feature $\mathbf{t}$, we use the same label as the closest point. This is exactly the same as the 1-Nearest-Neighbour classifier. Hence, in the limit of $\beta \to \infty$, we have the linear regression with Gaussian kernel identical to a 1-Nearest-Neighbour classifier.

Question 9

Given that we have a board with $n \times n$ holes, we can model the initial board configuration as a matrix such that:

$$\mathbf{B}_{init} = \left[b_{ij}\right] \tag{10}$$

where $i, j \in \{1, ..., n\}$, and b_ij:

$$b_{ij} = \begin{cases} 0 & \textit{mole hiding} \\ 1 & \textit{mole appearing} \end{cases}$$

With this setting, we aim to provide a series of transformation so that the resultant matrix is a null matrix, such that:

$$f(\mathbf{B}_{init}) = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \tag{11}$$

First let's explore the operations of hitting a mole in this context. Consider a 3×3 $\mathbf{B}_{init}$ with the following configuration:

$$\mathbf{B}_{init} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

If we can use a hit matrix $\mathbf{H}$ that can turn all the 1s into 0s, then we can have the null matrix, one such matrix is:

$$\begin{aligned} \mathbf{H}_{22} &= \begin{bmatrix} h_{22,11} & h_{22,12} & h_{22,13} \\ h_{22,21} & h_{22,22} & h_{22,23} \\ h_{22,31} & h_{22,32} & h_{22,33} \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \end{aligned}$$

This matrix means we are hitting on the entry $(2, 2)$. We can add that to $\mathbf{B}_{init}$ and then do a XOR operation which have the following arithmetic:

$$0 + 1 = 1 + 0 = 1$$
$$0 + 0 = 1 + 1 = 0$$

Doing this, we result in a null matrix:

$$\mathbf{B}_{init} + \mathbf{H}_{22} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Therefore, we can use a series of hit matrix $\mathbf{H}_{ij}$ to describe our hitting location, and since not all the entries need to be hit, we can introduce a coefficient k_{ij} for each matrix $\mathbf{H}_{ij}$, the equation above become:

$$\mathbf{B}_{init} + \sum_{i,j} k_{ij}\mathbf{H}_{ij} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

With this, expanding this to a general $n \times n$ matrix (11) can be rewritten as:

$$\mathbf{B}_{init} + \sum_{i,j} k_{ij}\mathbf{H}_{ij} = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix}_{n\times n} \tag{12}$$

Rearranging and do a $mod(2)$ operation on both side gives:

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

Since matrix addition is commutative, the order of hitting on the entries are irrelevant. So solving (14) for all the k_{ij} will give us the number of times each entries needed to be hit. Regarding the complexity of the algorithm, if we use Gaussian Jordan elimination for the inverse, the complexity is $\mathcal{O}(m^3)$ with respect to a $m \times m$ matrix. Therefore, the complexity of the algorithm to solve the puzzle is $\mathcal{O}(n^6)$.