COMP0078 Supervised Learning CW1

23073836, 22206530 (Group 39)

November 9, 2023

1 Part I

1.1 Linear Regression

Q1(a)

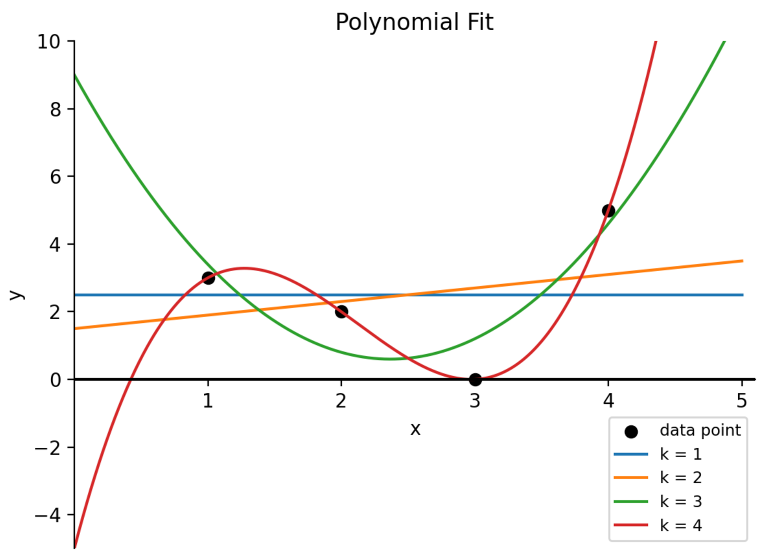

Figure 1: Polynomial bases of different dimensions fitting the data.

Q1(b)

$(k = 1)$, Equation: $y = 2.5$
$(k = 2)$, Equation: $y = 1.5 + 0.4x$
$(k = 3)$, Equation: $y = 9 - 7.1x + 1.5x^2$

Q1(c)

$(k = 1)$, MSE $= 3.250$
$(k = 2)$, MSE $= 3.050$
$(k = 3)$, MSE $= 0.800$
$(k = 4)$, MSE $= 0.000$

Q2(a).i

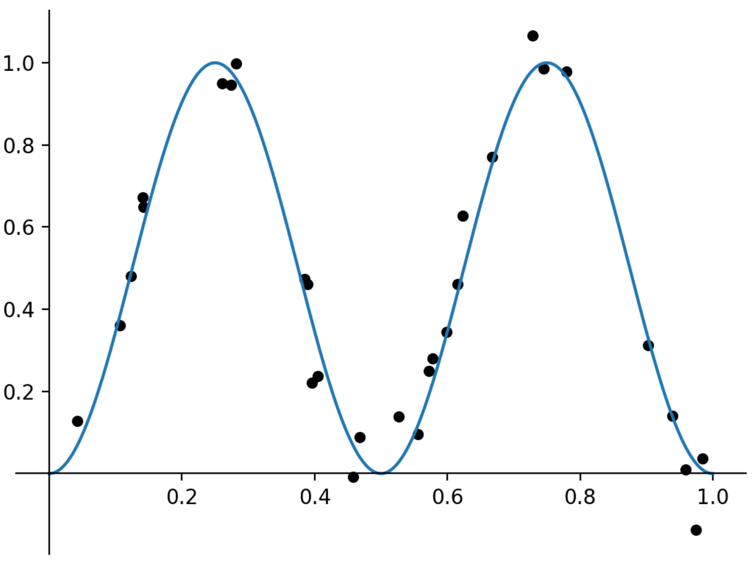

Figure 2: Function $sin^2(2\pi x)$ with random data set.

Q2(a).ii

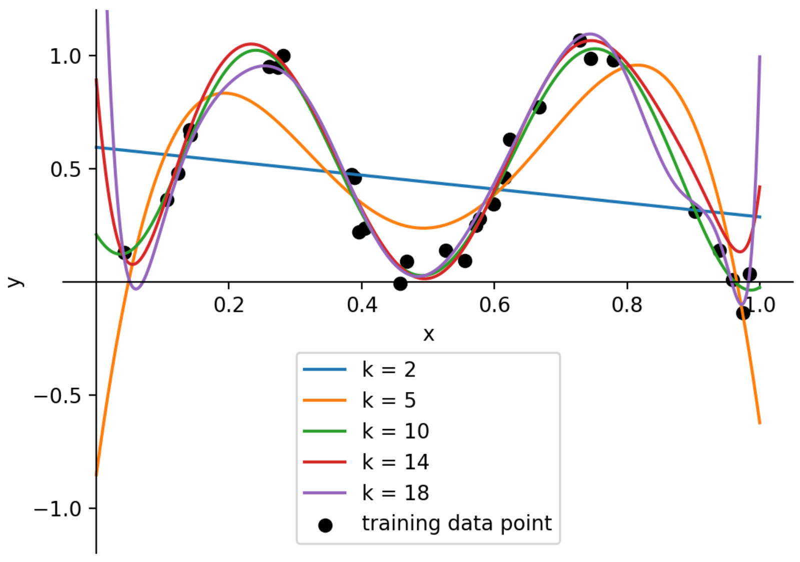

Figure 3: Fit the data set with polynomial bases of dimension k = 2, 5, 10, 14, 18.

Q2(b)

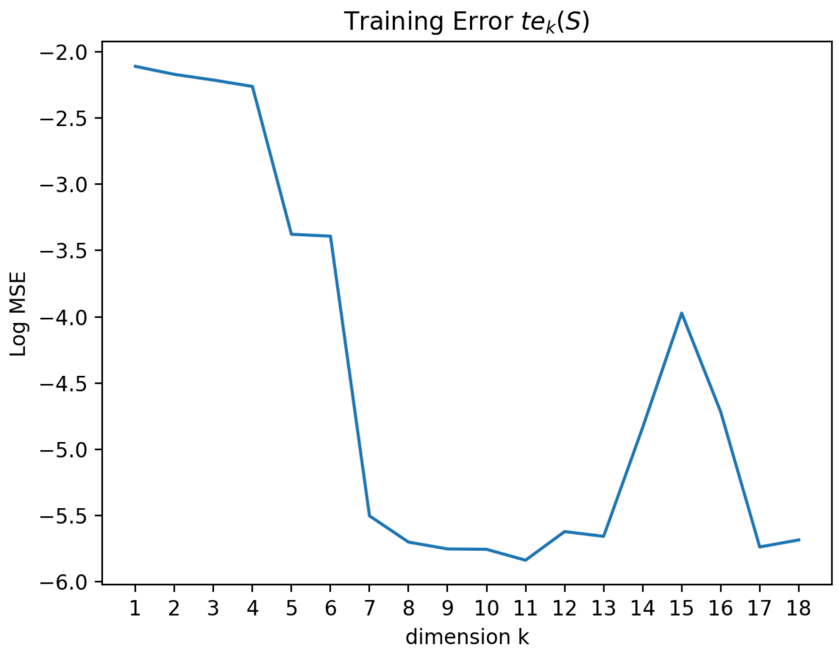

Figure 4: Natural log of the training error versus the polynomial dimension k = 1, . . . , 18.

Q2(c)

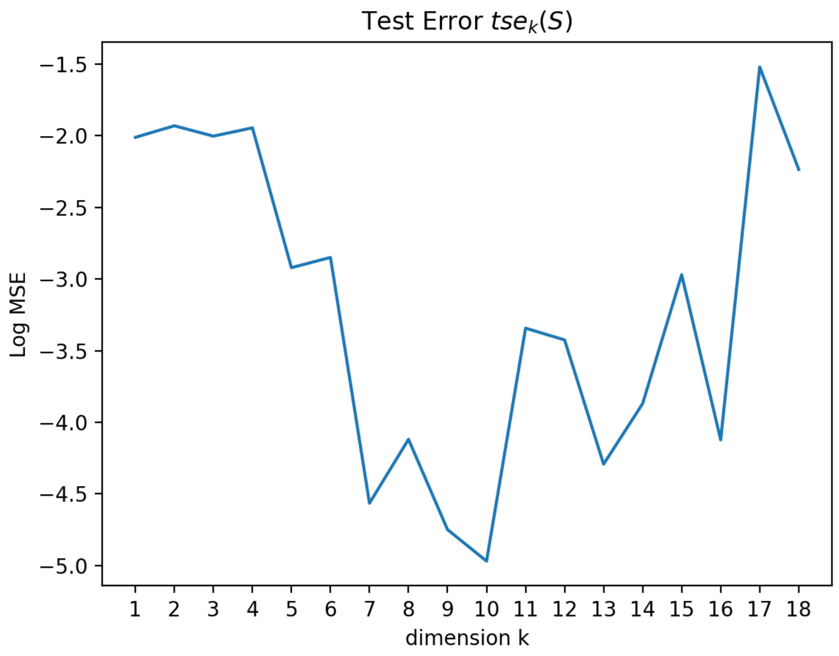

Figure 5: Natural log of the test error versus the polynomial dimension k = 1, . . . , 18

Q2(d)

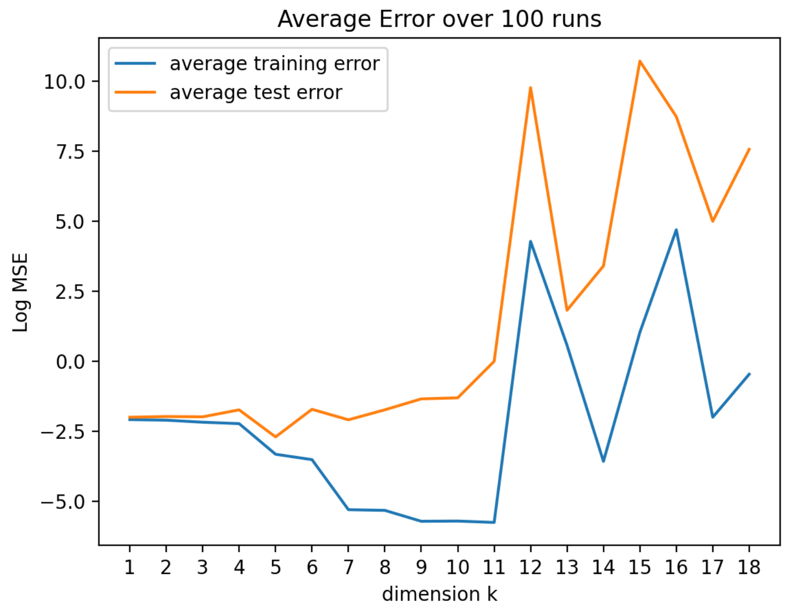

Figure 6: Natural log of the average training and test error versus the polynomial dimension k = 1, . . . , 18 over 100 runs.

Q3(b)

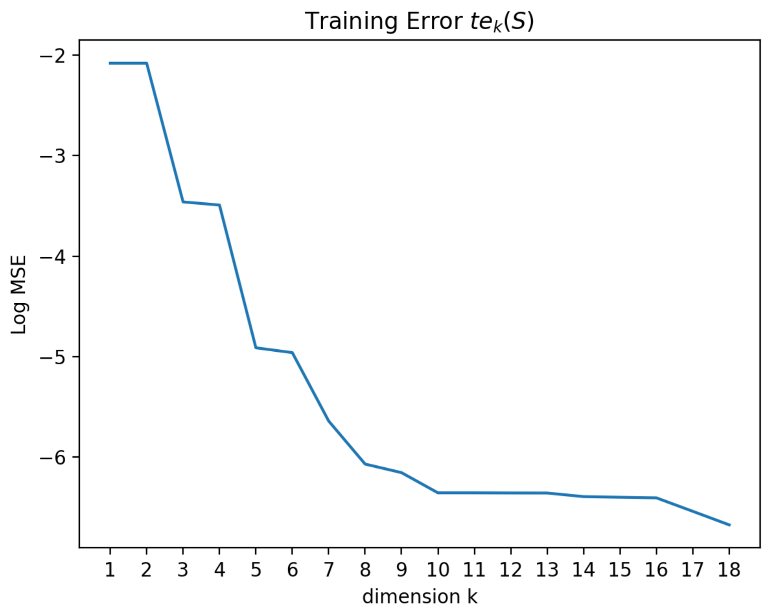

Figure 7: Natural log of the training error versus the polynomial dimension k = 1, . . . , 18 with new basis.

Q3(c)

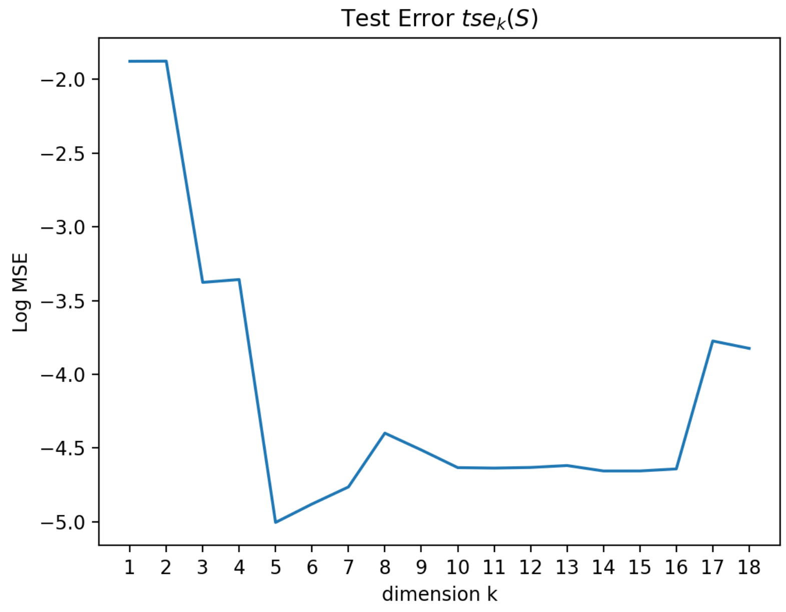

Figure 8: Natural log of the test error versus the polynomial dimension k = 1, . . . , 18 with new basis.

Q3(d)

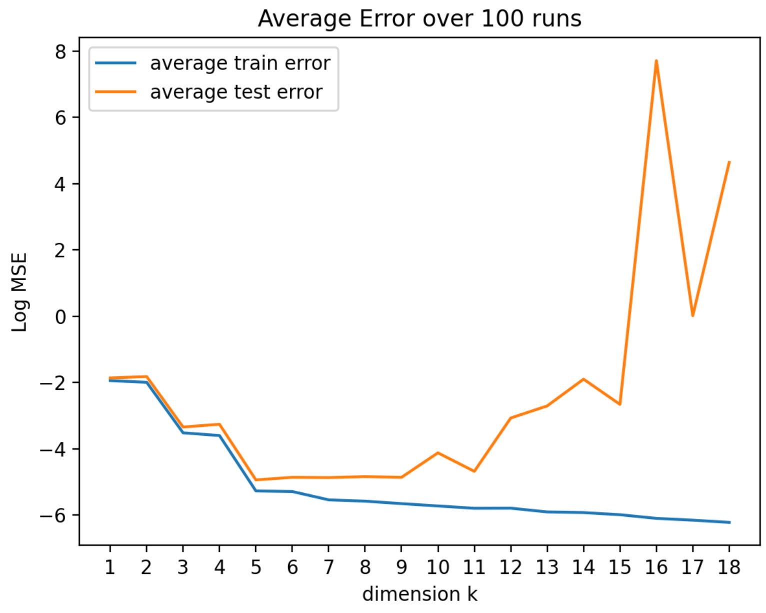

Figure 9: Natural log of the average training and test error versus the polynomial dimension k = 1, . . . , 18 over 100 runs with new basis.

1.2 Filtered Boston housing and kernels

Q4(a) Naive Regression

Training Error = 84.327
Testing Error = 85.109

Q4(b) Interpretation of the constant function

The constant function represents the average of the training data. By using this approach, any new data point will be predicted as this average, regardless of its features.

Q4(c) Linear Regression (single attribute)

Attribute	**Training Error**	**Test Error**
CRIM	71.607	73.050
ZN	73.161	74.411
INDUS	64.297	65.750
CHAS	81.773	82.461
NOX	68.633	70.080
RM	43.575	44.175
AGE	71.897	73.936
DIS	79.166	79.555
RAD	72.228	72.220
TAX	65.951	66.028
PTRATIO	63.408	61.425
LSTAT	38.471	38.762

Table 1: Training error and test error obtained from linear regression with single attribute

Q4(d) Linear Regression (all attributes)

Training Error = 21.832
Testing Error = 25.074

1.3 Kernelised ridge regression

Q5(a)

Optimal parameters $\gamma = 2^{-28}$ and $\sigma = 2^{8}$

Q5(b)

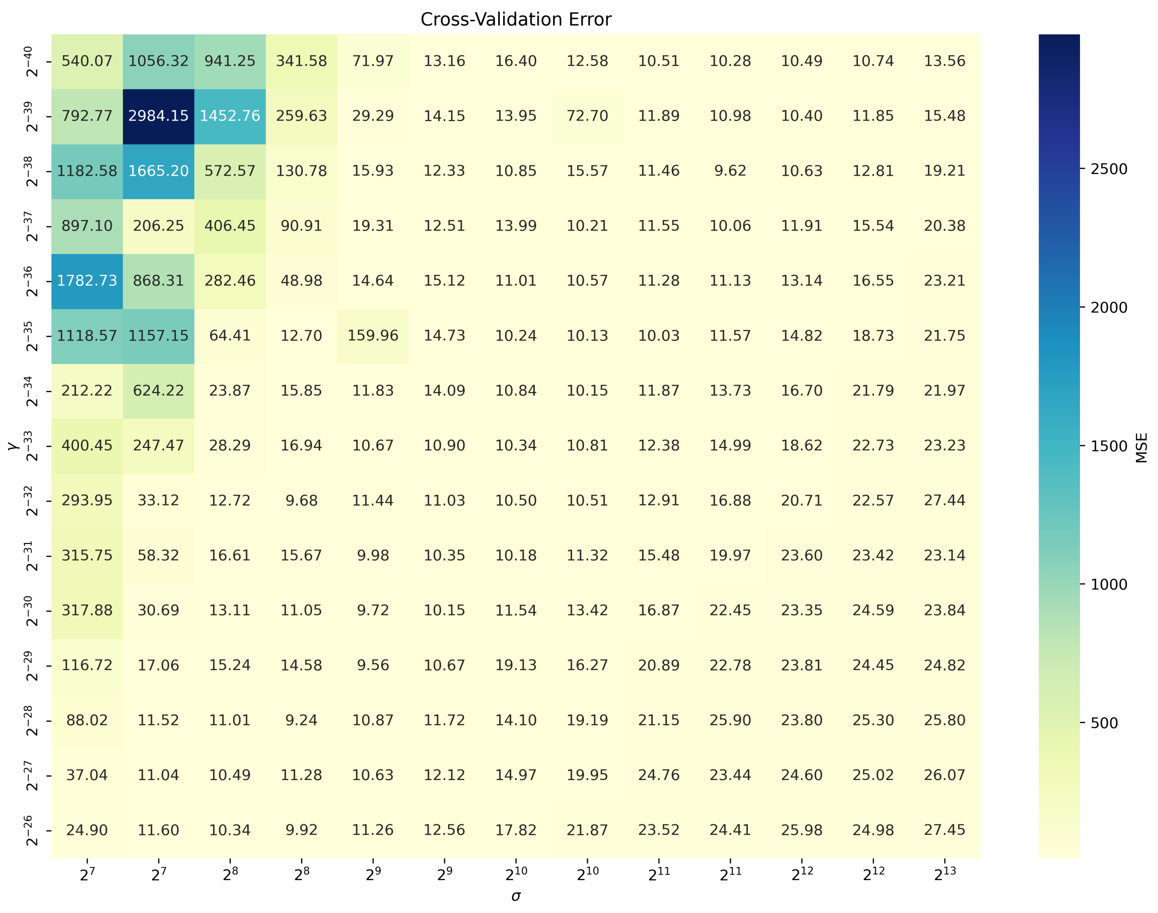

Figure 10: Cross validation error over different combinations of γ and σ.

Q5(c)

Best Parameter MSE – Training: 5.442, Test: 19.266

Q5(d)

Method	MSE train	MSE test
Naive Regression	86.208 ± 4.495	81.116 ± 8.978
Linear Regression (CRIM)	73.191 ± 4.545	69.799 ± 9.590
Linear Regression (ZN)	75.043 ± 4.298	70.749 ± 8.690
Linear Regression (INDUS)	66.474 ± 4.439	61.594 ± 9.019
Linear Regression (CHAS)	82.775 ± 4.431	80.737 ± 9.150
Linear Regression (NOX)	70.667 ± 4.629	66.063 ± 9.318
Linear Regression (RM)	45.453 ± 3.216	40.227 ± 6.498
Linear Regression (AGE)	74.309 ± 4.825	69.153 ± 9.743
Linear Regression (DIS)	80.968 ± 4.813	75.997 ± 9.752
Linear Regression (RAD)	74.179 ± 4.892	68.390 ± 9.891
Linear Regression (TAX)	67.874 ± 4.597	62.219 ± 9.351
Linear Regression (PTRATIO)	64.328 ± 3.806	59.643 ± 7.714
Linear Regression (LSTAT)	38.999 ± 2.333	37.850 ± 4.723
Linear Regression (all attributes)	22.903 ± 1.529	22.668 ± 3.461
Kernel Ridge Regression	7.779 ± 1.599	11.896 ± 2.259

Table 2: Summary of training error and test error with standard deviation for each method over 20 random splits of data.

2 Part II

Q6

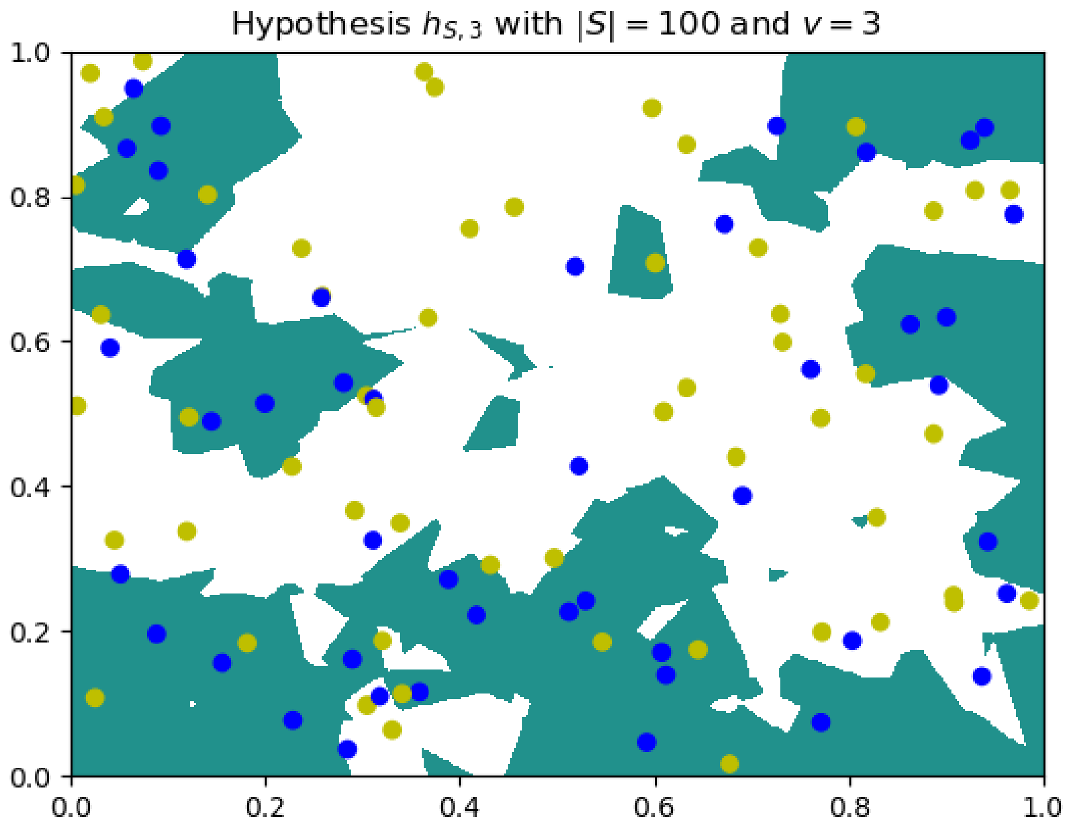

Figure 11: A hypothesis h_S, v visualized with $|S| = 100$ and $v = 3$.

Q7

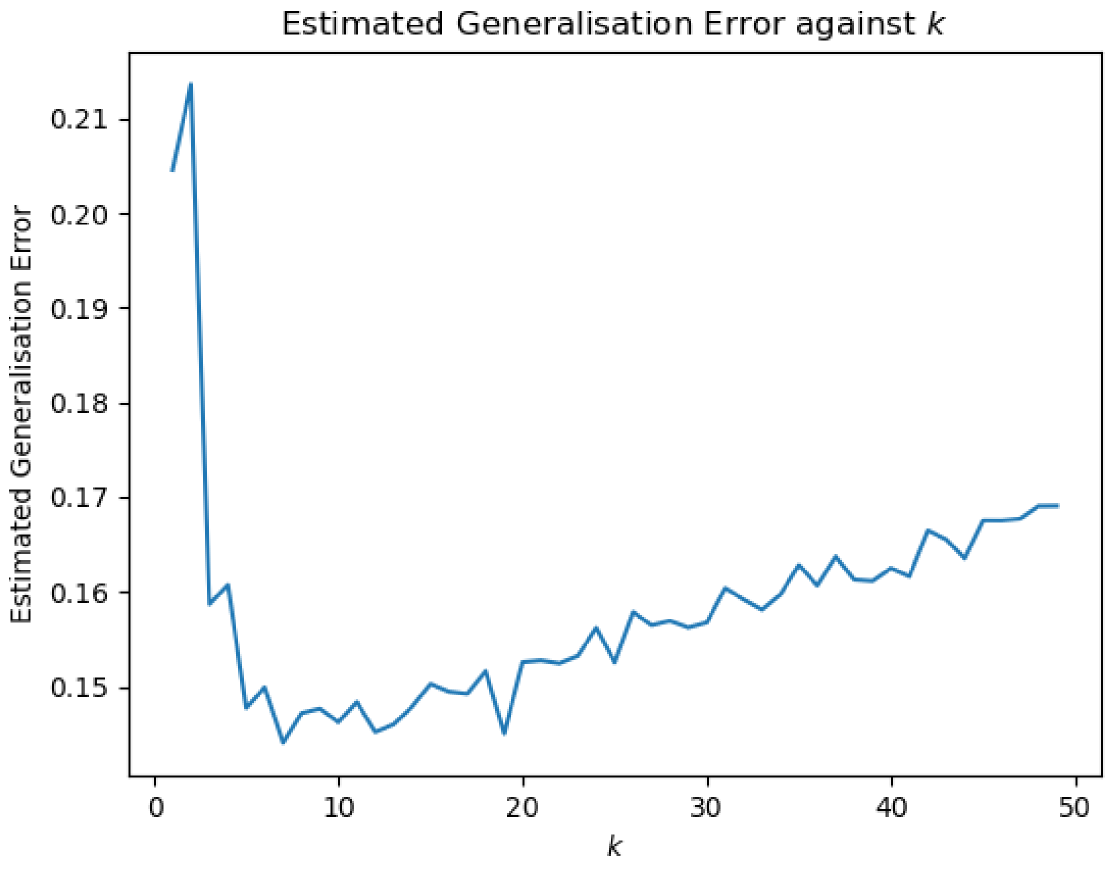

Figure 12: Estimated Generalisation Error against k.

Comment:

The estimated generalisation error is high when k is small, due to overfitting where the models only consider the nearest 1 or 2 neighbours.As k increases up to $k = 10$, the estimated generalisation error decreases, since to the models now consider larger number of neighbour around the test point, resulting in higher accuracy. There is a relatively large decrease in the estimated generalisation error from $k = 2$ to $k = 3$ because the training and test points are sampled from the 3-NN model with probability of 0.8, i.e. majority of the training and test points are sampled from the 3-NN model. Therefore, the k-NN models where k is around 3 will have a better performance. However, as k increases from 10 onwards, the estimated generalisation error starts to increase because now the number of neighbours that the models considered are too large that it leads to underfitting and lower accuracy.

Q8

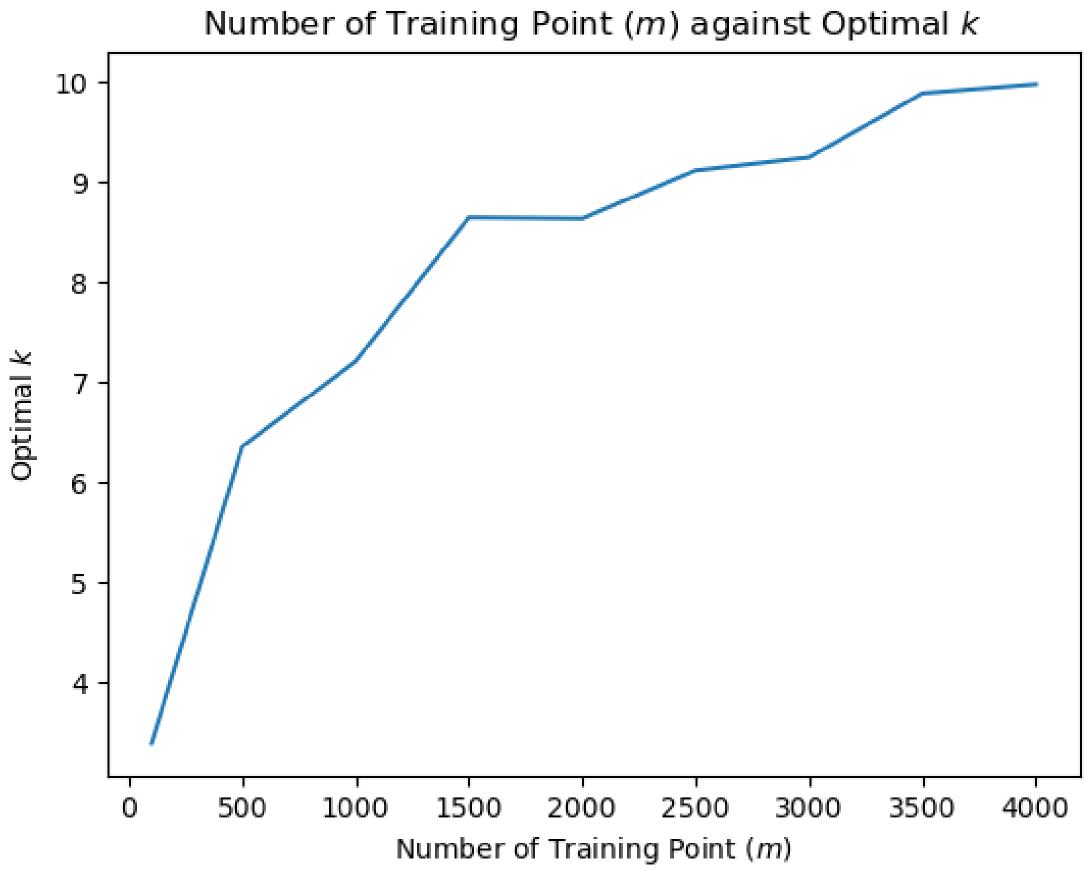

Figure 13: Number of Training Point (m) against Optimal K.

As the number of training points (m) increases, the optimal k increases because the relationship between the training points become more complex and it required a higher complexity model to capture such relationship. At the beginning when m is small, the optimal k is near to 3 because this a better representation of the 3-NN model where all training and test points are sampled from the 3-NN model with probability 0.8. From the graph, we can see that the rate of increase of the optimal k is larger at the beginning when m is low, where the gradient of the curve is steeper, and the rate of increase of the optimal k is lesser when m is large. This can due to at the beginning when m is low, more neighbours need to be considered in estimating the test points and as m starts to get larger, only a sufficiently large number of neighbours is needed to be considered to represent the test point.

3 Part III

Q9(a)

Given $K_c(\mathbf{X}, \mathbf{z}) := c + \sum_{i=1}^{n} x_i z_i$ where $\mathbf{x}, \mathbf{z} \in \mathbb{R}^n$, denote the matrix $\mathbf{A}$ such that $(\mathbf{A})_{ij} = K_c(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{A} \in \mathbb{R}^{m \times m}$ (assuming m training points).

K_c is a kernel function since it can expressed as $K_c(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$ where $\phi : \mathbb{R}^n \to \mathbb{R}^{n+1}$ such that $\phi(\mathbf{x})^\top = (\sqrt{c}, x_1, \ldots, x_n)$.

By definition, a kernel function $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is positive semidefinite if it is symmetric and the matrix $(K(\mathbf{x}_i, \mathbf{x}_j) : i, j = 1, \ldots, k)$ is positive semidefinite for every $k \in \mathbb{N}$ and every $\mathbf{x}_1, \ldots, \mathbf{x}_k \in \mathbb{R}^n$.

K_c is symmetric. Proof of K_c is symmetric:

$$K_c(\mathbf{x}, \mathbf{z}) = c + \sum_{i=1}^{n} x_i z_i = c + \sum_{i=1}^{n} z_i x_i = K_c(\mathbf{z}, \mathbf{x})$$

Hence, for K_c to be a positive semidefinite (PSD) kernel, $\mathbf{A}$ must be a PSD matrix. $\mathbf{A}$ is PSD matrix if and only if $b^\top \mathbf{A} b \geq 0 \quad \forall b \in \mathbb{R}^m$.

$$\begin{aligned}
b^\top \mathbf{A} b &= \sum_{i,j=1}^{m} b_i b_j K_c(\mathbf{x_i}, \mathbf{x_j}) \\
&= \sum_{i,j=1}^{m} b_i b_j \left(c + \sum_{k=1}^{n} x_{i,k} x_{j,k} \right) \\
&= \sum_{i,j=1}^{m} b_i b_j c + \sum_{i,j=1}^{m} b_i b_j \sum_{k=1}^{n} x_{i,k} x_{j,k} \\
&= c \sum_{i=1}^{m} b_i \sum_{j=1}^{m} b_j + \sum_{i,j=1}^{m} b_i b_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\
&= c \left\langle \sum_{i=1}^{m} b_i, \sum_{i=1}^{m} b_j \right\rangle + \left\langle \sum_{i=1}^{m} b_i \mathbf{x}_i, \sum_{j=1}^{m} b_j \mathbf{x}_j \right\rangle \\
&= c \left\| \sum_{i=1}^{m} b_i \right\|^2 + \left\| \sum_{i=1}^{m} b_i \mathbf{x}_i \right\|^2
\end{aligned}$$

$$b^\top \mathbf{A} b \geq 0 \Rightarrow c \left\| \sum_{i=1}^{m} b_i \right\|^2 + \left\| \sum_{i=1}^{m} b_i \mathbf{x}_i \right\|^2 \geq 0$$

The above is true, i.e. $\mathbf{A}$ is PSD, only when $c \geq 0$

$\therefore K_c$ is a PSD kernel if and only if $c \geq 0$.

Q9(b)

For linear regression with kernel function K_c, we will minimise the emperical error of the learning algorithm where we select $f(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x})$ and the loss function is:

$$\mathcal{E}_{emp}(\mathcal{S}, \mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 = \frac{1}{m}\sum_{i=1}^{m}(y_i - \mathbf{w} \cdot \phi(\mathbf{x}_i))^2$$

Since $\mathcal{E}_{emp}(\mathcal{S}, \mathbf{w})$ is differentiable with respect to $\mathbf{w} \cdot \phi(\mathbf{x})$ and $\mathbf{w}$ is a minimizer of $\mathcal{E}_{emp}$, then by Representor Theorem, $\mathbf{w}$ has the form

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i \phi(x_i) \implies f(\mathbf{z}) = \langle \mathbf{w}, \phi(\mathbf{z}) \rangle = \sum_{i=1}^{m} \alpha_i K(\mathbf{x}_i, \mathbf{z})$$

$$\begin{aligned}
f(\mathbf{z}) &= \sum_{i=1}^{m} \alpha_i K(\mathbf{x}_i, \mathbf{z}) \\
&= \sum_{i=1}^{m} \alpha_i \left(c + \sum_{j=1}^{n} x_{i,j} z_j \right) \\
&= \sum_{i=1}^{m} \alpha_i c + \sum_{i=1}^{m} \alpha_i \left(\sum_{j=1}^{n} x_{i,j} z_j \right) \\
&= \sum_{i=1}^{m} \alpha_i \mathbf{x}_i \cdot \mathbf{z} + c \sum_{i=1}^{m} \alpha_i
\end{aligned}$$

From the above formula, we can see that c acts as a regularisation term that regularises the predict value. A higher value of c leads to stronger regularisation and penalises large coefficients. As a result, the coefficients will be lower and some may even be zero. If $\alpha_i = 0$, this means that the input $\mathbf{x}_i$ has no influence on the predicted value. When $c = 0$, there is no regularisation and the solution will be the same as that of a least squares problem.

Q10

Given Gausian kernel $K_\beta(\mathbf{x},\mathbf{t}) = \exp(-\beta\|\mathbf{x}-\mathbf{t}\|^2)$ and we perform linear regression with the Gaussian kernel, then a function $f:\mathbb{R}^n \to \mathbb{R}$ will have the form $f(\mathbf{t}) = \sum_{i=1}^m \alpha_i K_\beta(\mathbf{x},\mathbf{t})$. This form is obtained by using the Representor Theorem. The corresponding classifier is then $\text{sign}(f(t))$ i.e.

$$\text{sign}(f(\mathbf{t})) = \text{sign}\left(\sum_{i=1}^m \alpha_i K_\beta(\mathbf{x},\mathbf{t})\right) = \text{sign}\left(\sum_{i=1}^m \alpha_i \sum_{j=1}^n \exp(-\beta\|\mathbf{x}_j-\mathbf{t}\|^2)\right)$$

To simulate a 1-Nearest Neighbor Classifier, we will only consider the training point nearest to the test point $\mathbf{t}$. This means we want $\exp(-\beta\|\mathbf{x}_j-\mathbf{t}\|^2)$ to be 0 when $j \neq k$ where $\mathbf{x}_k$ has the lowest distance to $\mathbf{t}$ i.e.

$$\min_{j=1,\dots,m} \|\mathbf{x}_j-\mathbf{t}\|^2 = \|\mathbf{x}_k-\mathbf{t}\|^2$$

For $\exp(-\beta\|\mathbf{x}_j-\mathbf{t}\|^2)$ to be 0, β has to be ∞. For $\exp(-\beta\|\mathbf{x}_k-\mathbf{t}\|^2)$, β can be any number in $\mathbb{R}^+$ except $+\infty$. We will set $\beta = 0$ for simplicity in calculations. Hence, β will have the form

$$\beta = \begin{cases} 0 & \text{if } \|\mathbf{x}_k-\mathbf{t}\|^2 = \min_{j=1,\dots,m}\|\mathbf{x}_j-\mathbf{t}\|^2 \\ \infty & \text{otherwise} \end{cases}$$

The above β function is in terms of $\mathbf{x}_1,\dots,\mathbf{x}_m$ and test point $\mathbf{t}$, and indicates that $\beta = 0$ for $\exp(-\beta\|\mathbf{x}_k-\mathbf{t}\|^2)$, where $\mathbf{x}_k$ is the point that is closest to $\mathbf{t}$, and $\beta = \infty$ for all other points.

Intuitively, the Gaussian kernel represents the distance between vectors (i.e. the squared norm of their distance). If the vectors are closer to each other, then $\|\mathbf{x}-\mathbf{t}\|^2$ will be smaller and $\exp(-\beta\|\mathbf{x}-\mathbf{t}\|^2)$ will be larger. This means that closer points will have higher influence on the test point. In our case, only the closest point will have influence on the test point as we are simulating the 1-Nearest Neighbor Classifier.

Q11(a)

For f_i,

$$\mathcal{E}_{\rho_i}(f_i) = \sum_{k=1}^{|C|} \mathbf{1}_{\{f_i(x_k)\neq y_k\}} \times \rho_i(\{(x_k, y_k)\}) = \sum_{k=1}^{2n} \mathbf{1}_{\{f_i(x_k)\neq y_k\}} \times \frac{1}{2n}\mathbf{1}_{\{f_i(x_k)=y_k\}} = 0$$

With probability $\frac{1}{2n}$, $\mathbf{1}_{\{f_i(x_k)\neq y_k\}} = 0$ since $f_i(x) = y$ while with probability 0, $\mathbf{1}_{\{f_i(x_k)\neq y_k\}} = 1$.
$\therefore \mathcal{E}_{\rho_i}(f_i) = 0$
$\forall f : \mathcal{X} \to \mathcal{Y}$ where $f \neq f_i$,

$$\mathcal{E}_{\rho_i}(f) = \sum_{k=1}^{|C|} \mathbf{1}_{\{f(x_k)\neq y_k\}} \times \rho_i(\{(x_k, y_k)\}) = \sum_{k=1}^{2n} \mathbf{1}_{\{f(x_k)\neq y_k\}} \times \frac{1}{2n}\mathbf{1}_{\{f_i(x_k)=y_k\}}$$

$\exists$ at least one $x' \in \mathcal{X}$ such that $f(x') \neq f_i(x')$ and $f_i(x') = y \implies f(x') \neq y$.
$\therefore \mathcal{E}_{\rho_i}(f) \geq \frac{1}{2n}$

$$0 < \frac{1}{2n} \implies \mathcal{E}_{\rho_i}(f_i) < \mathcal{E}_{\rho_i}(f) \implies \mathcal{E}_{\rho_i}(f_i) = \inf_{f:\mathcal{X}\to\mathcal{Y}} \mathcal{E}_{\rho_i}(f) = 0$$

Q11(b)

Since we have defined the S_j and S_j^i, if the distribution is p_i, then the possible training sets A can receive are $S_1^i, ..., S_k^i$, and all these training sets have the same probability of being sampled. Therefore, we can show that

$$\mathbb{E}_{S\sim\rho_i^n}\mathcal{E}_{\rho_i}(A(S)) = \frac{1}{k}\sum_{j=1}^{k} \mathcal{E}_{\rho_i}\left(A\left(S_j^i\right)\right) \tag{1}$$

Using the fact the for any set of scalars $\alpha_1, ..., \alpha_m$, we have $\max_\ell \alpha_\ell \geq \frac{1}{m}\sum_{\ell=1}^m \alpha_\ell \geq \min_\ell \alpha_\ell$. We then can show

$$\begin{aligned}\max_{i=1,...,T} \mathbb{E}_{S\sim\rho_i^n}\mathcal{E}_{\rho_i}(A(S)) &\geq \frac{1}{T}\sum_{i=1}^{T}\frac{1}{k}\sum_{j=1}^{k} \mathcal{E}_{\rho_i}\left(A\left(S_j^i\right)\right) \\ &= \frac{1}{k}\sum_{j=1}^{k}\frac{1}{T}\sum_{i=1}^{T} \mathcal{E}_{\rho_i}\left(A\left(S_j^i\right)\right) \quad . \\ &\geq \min_{j=1,...,K}\frac{1}{T}\sum_{i=1}^{T} \mathcal{E}_{\rho_i}\left(A\left(S_j^i\right)\right)\end{aligned} \tag{2}$$

Q11(c)

For every function $f : \mathcal{X} \to \mathcal{Y}$ and every i, we have

$$\mathcal{E}_{\rho_i}(A(S_j^i)) = \frac{1}{2n} \sum_{x \in C} \mathbf{1}_{\{A(S_j^i)(x) \neq f_i(x)\}}. \tag{3}$$

We can get the lower bound of the risk with respect to the errors only over $S_j^{'}$. And given that $|C| = 2n$, $S_j = (x_1, \ldots, x_n)$ and $S_j^{'} = \{v_1, \ldots, v_p\}$ is the subset of points of C that do not belong to S_j, it is clear that $n = p$. Then we have

$$\begin{aligned} \mathcal{E}_{\rho_i}(A(S_j^i)) &\geq \frac{1}{2n} \sum_{r=1}^{p} \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} \\ &\geq \frac{1}{2p} \sum_{r=1}^{p} \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}}. \end{aligned} \tag{4}$$

Again, use the fact that "maximum" is greater or equal to "average" and that "average" is greater or equal to "minimum", we can get

$$\begin{aligned} \frac{1}{T} \sum_{i=1}^{T} \mathcal{E}_{\rho_i}\left(A\left(S_j^i\right)\right) &\geq \frac{1}{T} \sum_{i=1}^{T} \frac{1}{2p} \sum_{r=1}^{p} \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} \\ &= \frac{1}{2p} \sum_{r=1}^{p} \frac{1}{T} \sum_{i=1}^{T} \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} \\ &\geq \frac{1}{2} \min_{r=1,\ldots,p} \frac{1}{T} \sum_{i=1}^{T} \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}}. \end{aligned} \tag{5}$$

Q11(d)

For any $r = 1, \ldots, p$, we partition the f in $\mathcal{Y}^{\mathcal{C}}$ into $\frac{T}{2}$ pairs $(f_i, f_{i'})$ such that if $f_i(x) = 1$ then $f_{i'}(x) = -1$ and if $f_i(x) = -1$ then $f_{i'}(x) = 1$. This is possible since $T = |\mathcal{Y}^{\mathcal{C}}| = 2^{2n}$ and for every additional x' added into $\mathcal{C}$, there will be 2 times more f added into $\mathcal{Y}^{\mathcal{C}}$ which correspond to $f(x') = 1$ and $f(x') = -1$. Hence, $\forall$ v_r where $r = 1, \ldots, p$, we can partition $\mathcal{Y}^{\mathcal{C}}$ into $\frac{T}{2}$ pairs $(f_i, f_{i'})$ such that $f_i(v_r) \neq f_{i'}(v_r)$.

Since $|S_j^i| = |S_j^{i'}| = n$ and each pair of $(x_k, f_i(x_k))$ for $k = 1, \ldots, n$ in S_j^i and $S_j^{i'}$ is independently sampled from ρ, then $A(S_j^i)(x) = A(S_j^{i'})(x)$

$$\begin{aligned} A(S_j^i)(x) = A(S_j^{i'})(x) \implies & \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} + \mathbf{1}_{\{A(S_j^{i'})(v_r) \neq f_{i'}(v_r)\}} \\ &= \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} + \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_{i'}(v_r)\}} = 1 \end{aligned}$$

since $f_i(v_r) \neq f_{i'}(v_r)$, then $A(S_j^i)(v_r)$ surely not equal to either $f_i(v_r)$ or $f_{i'}(x)$. This indicates that in half of the cases $\mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} = 1$ and in the other half of the cases $\mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} = 0$.

Hence

$$\frac{1}{T} \sum_{i=1}^{T} \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} = \frac{1}{T} \left(\frac{T}{2}\right) = \frac{1}{2}$$

Q11(e)

The Markov's inequality is saying that if Z is a nonnegative random variable and $a > 0$, then the probability that Z is at lease a is at most the expectation of Z divided by a:

$$\mathbb{P}(Z \geq a) \leq \frac{\mathbb{E}(\mathbb{Z})}{a}. \tag{6}$$

Since $Z : [0, 1]$ and $\mathbb{E}[Z] = \mu$, then $(1 - Z)$ is also between 0 to 1 and $\mathbb{E}[1 - Z] = 1 - \mu$. Thus, we can apply the Markov's inequality to $(1 - Z)$

$$\begin{aligned} \mathbb{P}(1 - Z \geq a) &\leq \frac{\mathbb{E}[1 - Z]}{a} \\ &= \frac{1 - \mu}{a}. \end{aligned} \tag{7}$$

By swapping the position between inequality, we have

$$\mathbb{P}(Z \leq 1 - a) \leq \frac{1 - \mu}{a} \tag{8}$$

We want to find $\mathbb{P}(Z > 1 - a)$, thus we need to take the complement of equation 8

$$\begin{aligned} \mathbb{P}(Z > 1 - a) &\geq 1 - (\frac{1 - \mu}{a}) \\ &= \frac{a - 1 + \mu}{a} \\ &= \frac{\mu - (1 - a)}{a}. \end{aligned} \tag{9}$$

Q11(f)

By the result of Q11(e),

$$\mathbb{P}_{S \sim \rho^n}\left(\mathcal{E}_\rho(A(S)) > \frac{1}{8}\right) = \mathbb{P}_{S \sim \rho^n}\left(\mathcal{E}_\rho(A(S)) > 1 - \frac{7}{8}\right) \geq \frac{\mathbb{E}_{S \sim \rho^n}\mathcal{E}_\rho(A(S)) - \frac{1}{8}}{\frac{7}{8}}$$

By the results of Q11(b), (c) and (d), $\exists$ a distribution ρ over $\mathcal{X} \times \mathcal{Y}$ such that

$$\begin{aligned} \mathbb{E}_{S \sim \rho^n}\mathcal{E}_\rho(A(S)) &\geq \min_{j=1,\ldots,k} \frac{1}{T}\sum_{i=1}^{T} \mathcal{E}_{\rho_i}(A(S_j^i)) \\ &\geq \frac{1}{2} \min_{r=1,\ldots,p} \frac{1}{T}\sum_{i=1}^{T} \mathbf{1}_{\{A(S_j^i)(v_r) \neq f_i(v_r)\}} \\ &= \frac{1}{2}\left(\frac{1}{2}\right) \\ &= \frac{1}{4} \end{aligned}$$

$\therefore \mathbb{E}_{S \sim \rho^n}\mathcal{E}_\rho(A(S)) \geq \frac{1}{4} \implies \mathbb{P}_{S \sim \rho^n}\left(\mathcal{E}_\rho(A(S)) > \frac{1}{8}\right) \geq \frac{\frac{1}{4} - \frac{1}{8}}{\frac{7}{8}} = \frac{1}{7}$

Q11(g)i.

For any learning algorithm $A : S \mapsto (f : \mathcal{X} \to \mathcal{Y})$ for the task of binary classification and for any integer $n \in \mathbb{N}$, there exists a distribution ρ over $\mathcal{X} \times \mathcal{Y}$ such that

- There exists a classification function where the error of that function is zero, i.e.

$$\inf_{f:\mathcal{X}\to\mathcal{Y}} \mathcal{E}_\rho(f) = 0$$

- The probability of the error of a classification obtained from a learning algorithm A is greater than $\frac{1}{8}$ is at least $\frac{1}{7}$, i.e.

$$\mathbb{P}_{S\sim\rho^n}\left(\mathcal{E}_\rho(A(S)) > \frac{1}{8}\right) \geq \frac{1}{7}$$

Q11(g)ii.

By definition of learnable space of function,

$$\begin{aligned}
&\mathbb{P}_{S\sim\rho^n}\left(\mathcal{E}_\rho(A(S)) - \inf_{f\in\mathcal{H}} \mathcal{E}_p(f) \leq \epsilon\right) \geq 1-\delta \\
&\implies 1 - \mathbb{P}_{S\sim\rho^n}\left(\mathcal{E}_\rho(A(S)) - \inf_{f\in\mathcal{H}} \mathcal{E}_p(f) > \epsilon\right) \geq 1-\delta \\
&\implies \mathbb{P}_{S\sim\rho^n}\left(\mathcal{E}_\rho(A(S)) - \inf_{f\in\mathcal{H}} \mathcal{E}_p(f) > \epsilon\right) \leq \delta
\end{aligned}$$

By contradiction, assume $\epsilon < \frac{1}{8}$ and $\delta < \frac{1}{7}$. Then

$$\mathbb{P}_{S\sim\rho^n}\left(\mathcal{E}_\rho(A(S)) > \frac{1}{8} > \epsilon\right) \geq \frac{1}{7} > \delta \implies \mathbb{P}_{S\sim\rho^n}\left(\mathcal{E}_\rho(A(S)) > \epsilon\right) > \delta$$

By the No-Free-Lunch Theorem, $\inf_{f\in\mathcal{H}} \mathcal{E}_p(f) = 0$, then

$$\mathbb{P}_{S\sim\rho^n}\left(\mathcal{E}_\rho(A(S)) - \inf_{f\in\mathcal{H}} \mathcal{E}_p(f) > \epsilon\right) > \delta$$

which contradicts the definiton of learnable space of function. Hence, the No-Free-Lunch theorem imply that the space $\mathcal{Y}^\mathcal{X}$ of all functions $f : \mathcal{X} \to \mathcal{Y}$ is not learnable.

Q11(g)iii.

The No-Free-Lunch theorem implies that no learning algorithm exists such that the error of the learning algorithm is zero, i.e. a universal learning algorithm that can succeed on all learning tasks does not exist. This leads to the Bias Variance Dilemma where Bias and Variance tend to trade off against each other.

When designing a machine learning algorithm, we need to select a hypothesis space for the function $f : \mathcal{X} \to \mathcal{Y}$ with either high or low complexity. In general, a low complexity hypothesis space will have high bias but low variance. This leads to underfitting and the functions are too simple to capture the complex relationships within the data. As we increase the complexity, the bias will decrease and variance will increase. This leads to overfitting and the functions are sensitive to noise.