COMP0078 Supervised Learning
Coursework 1

Daniel May
ucabdd3@ucl.ac.uk

Olivier Kraft
ucabpk0@ucl.ac.uk

November 16, 2020

Contents

1 Question 1

(a)

Figure 1 plots the data set with curves for fitted polynomial bases of dimension $k = 1, 2, 3, 4$. We can see that as k increases, the corresponding curve fits the data set increasingly well.

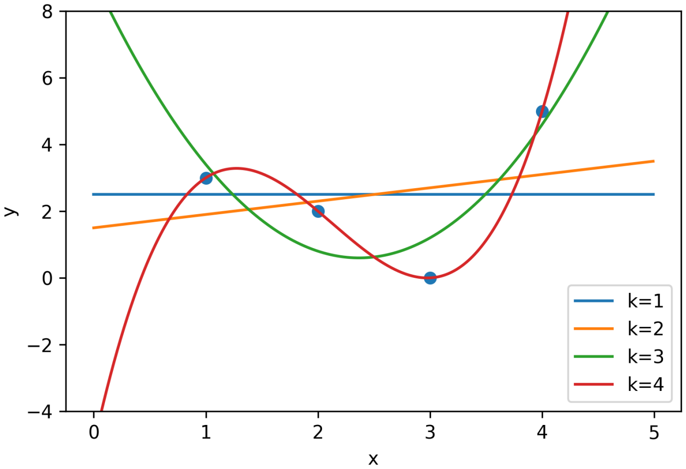

Figure 1: Data set fitted with polynomial bases of dimension $k = 1, 2, 3, 4$

(b)

The equations for the curves fitted for $k = 1, 2, 3$ are:

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

We also confirm that the equation for $k = 4$ is $5 + 15.17x - 8.5x^2 + 1.33x^3$.

(c)

The mean square errors ($MSE = SSE/m$) for $k = 1, 2, 3, 4$ are

- $k = 1 : 3.25$
- $k = 2 : 3.05$
- $k = 3 : 0.80$
- $k = 4 : 1.32e^{-26}$

This accords with (a), where we could see that as k increased, the equation fit the data set increasingly well.

2 Question 2

(a)

(i)

Figure 2 plots our data set of 30 samples superimposed with the curve $sin^2(2\pi x)$ in the range $0 \leq x \leq 1$.

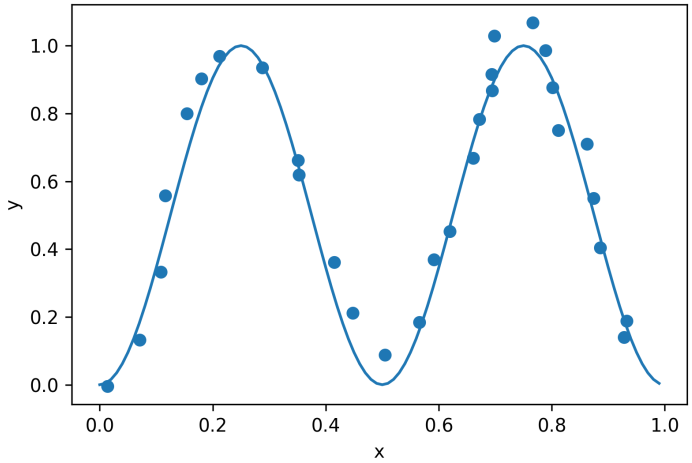

Figure 2: Data set superimposed with curve $sin^2(2\pi x)$ in the range $0 \leq x \leq 1$

(ii)

Figure 3 shows the data set with curves for fitted polynomial bases of dimension $k = 2, 5, 10, 14, 18$. We can see that as k increases, the corresponding curve fits the data set increasingly well.

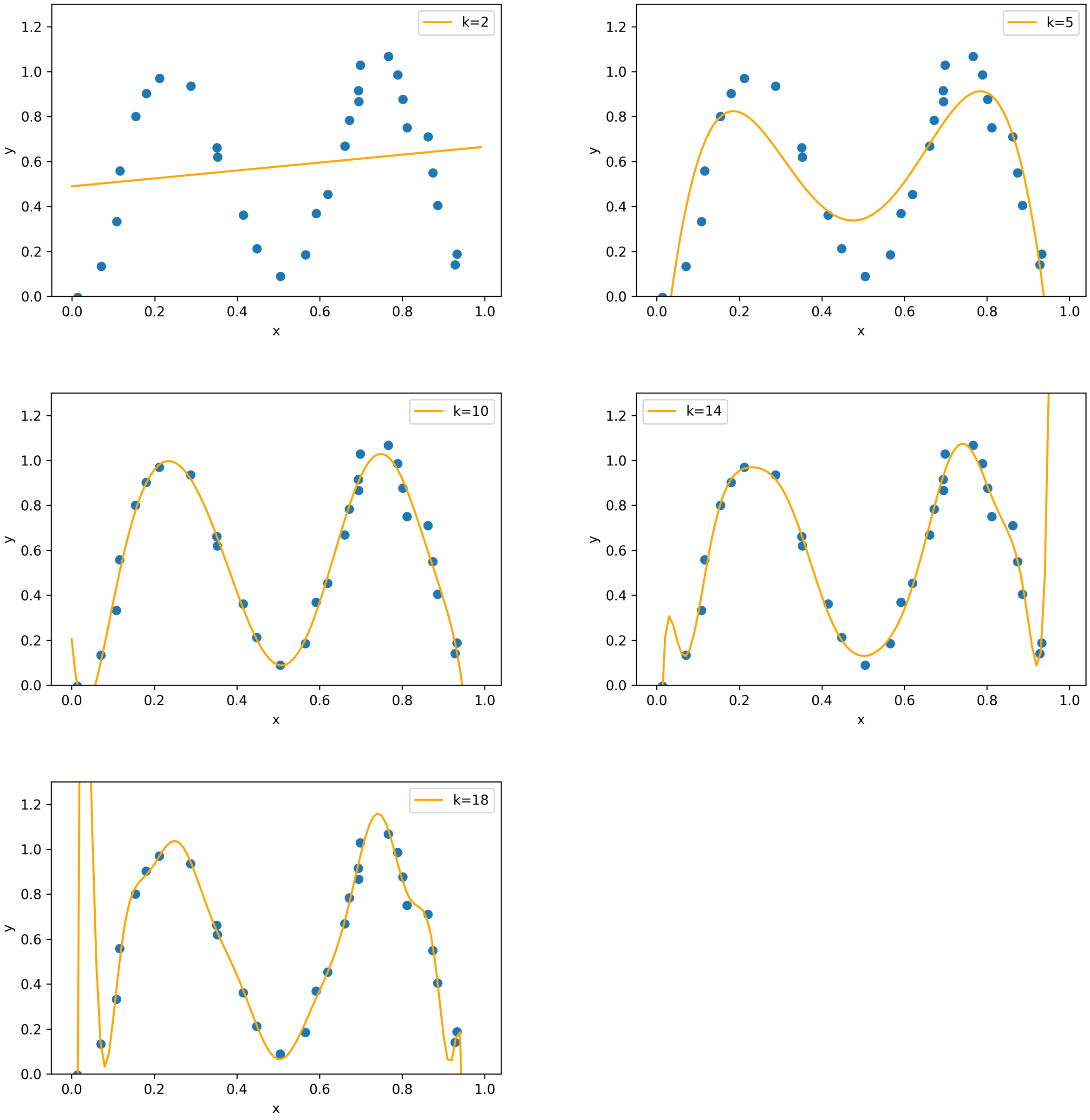

Figure 3: Data set fitted with polynomial bases of dimension $k = 2, 5, 10, 14, 18$

(b)

Figure 4 plots the natural log (ln) of the training error (MSE) versus the polynomial of dimension $k = 1, ..., 18$. As expected from Figure 3, we see that this is a decreasing function of k.

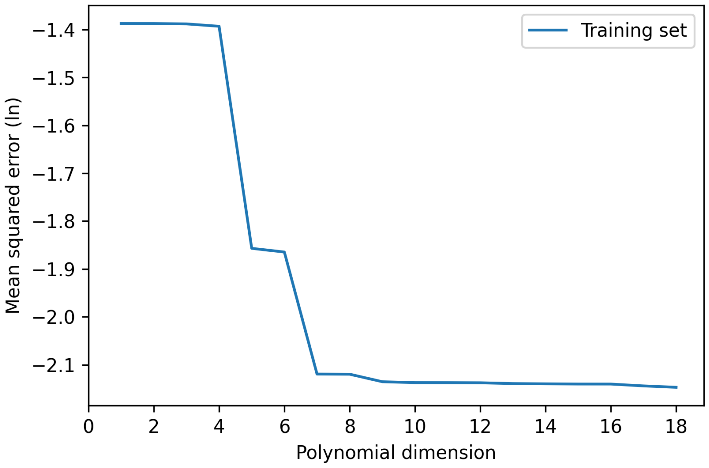

Figure 4: The natural log (ln) of the training error versus the polynomial of dimension $k = 1, ..., 18$

(c)

We generate a test set of a thousand points, then plot the natural log (ln) of the test error (MSE) versus the polynomial of dimension $k = 1, ..., 18$ fitted from the training set, seen in Figure 5. Unlike the training error, the test error eventually increases with increasing k, demonstrating overfitting to the training set.

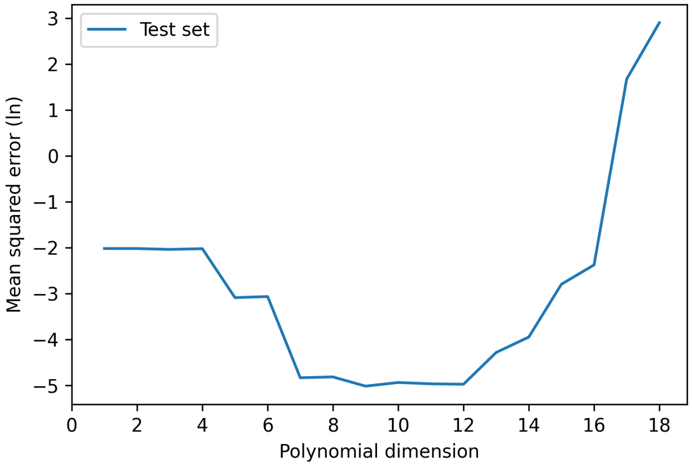

Figure 5: The natural log (ln) of the test error versus the polynomial of dimension $k = 1, ..., 18$ fitted from the training set

(d)

We repeat (b) and (c) and plot the natural log (ln) of the errors averaged over 100 runs, smoothing out the curves, shown in Figure 6. These curves accord with (b) and (c), where the training error is a decreasing function of k, and the test error initially fluctuates erratically with brief dips and rises with increasing k, but eventually clearly increases with increasing k, demonstrating overfitting to the training set.

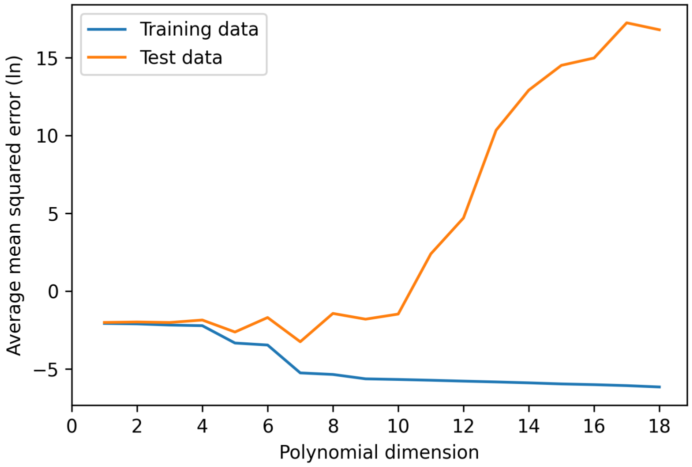

Figure 6: The natural log (ln) of the training and test errors averaged over 100 runs versus the polynomial of dimension $k = 1, ..., 18$ fitted from the training set

3 Question 3

In this question, we repeat the experiments in 2 (b-d) with the basis $sin(1x), sin(2x), sin(3x), ..., sin(kx)$ for $k = 1, ..., 18$.

(b)

Figure 7 plots the natural log (ln) of the training error (MSE) versus the basis of dimension $k = 1, ..., 18$. We see that this is a decreasing function of k.

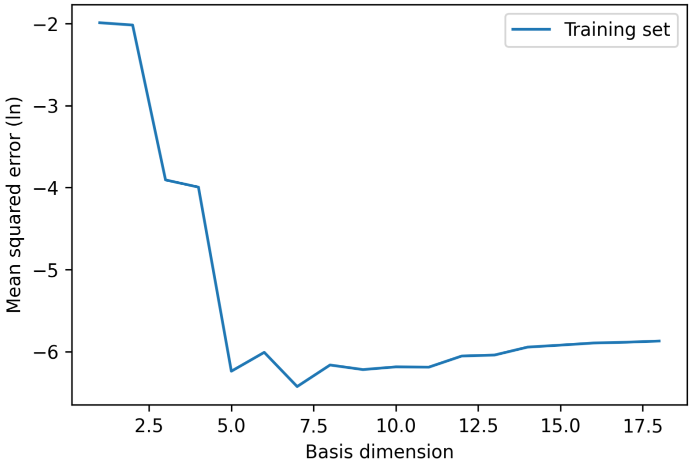

Figure 7: The natural log (ln) of the training error versus the basis of dimension $k = 1, ..., 18$

(c)

We generate a test set of a thousand points, then plot (Figure 8) the natural log (ln) of the test error (MSE) versus the basis of dimension $k = 1, ..., 18$ fitted from the training set.

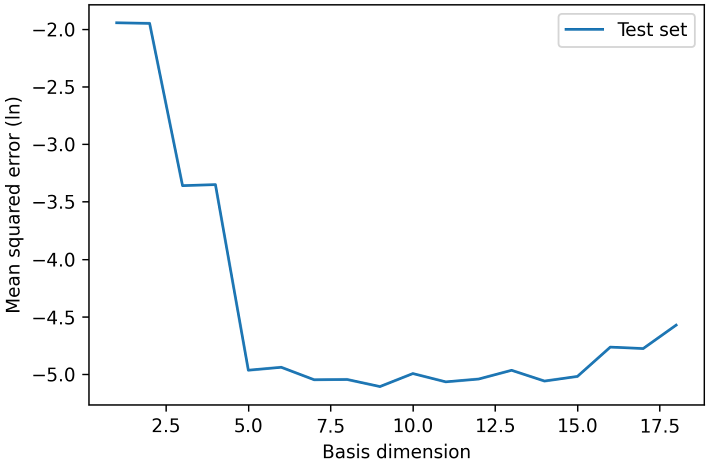

Figure 8: The natural log (ln) of the test error versus the basis of dimension $k = 1, ..., 18$ fitted from the training set

(d)

We repeat (b) and (c) and plot the natural log (ln) of the errors averaged over 100 runs, smoothing out the curves, shown in Figure 9.

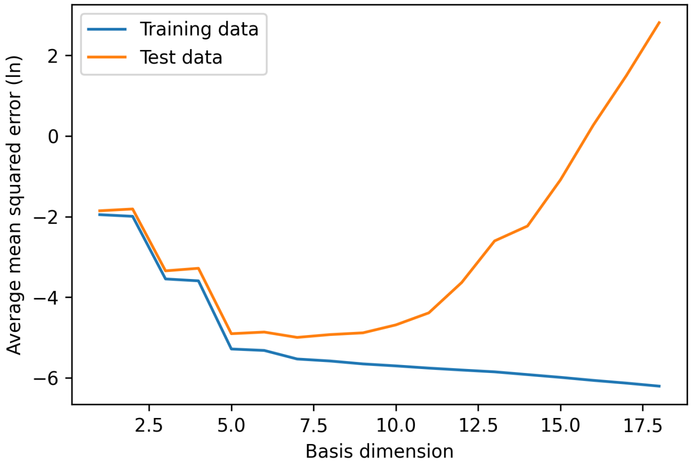

Figure 9: The natural log (ln) of the training and test errors averaged over 100 runs versus the basis of dimension $k = 1, ..., 18$ fitted from the training set

4 Question 4

(i) Preliminary remarks

- Values included in the report for Questions 4 and 5 are rounded to two decimals.
- We begin by defining a function that randomly splits the data into a training set (that includes 2/3 of data points) and a test set (that consists of the remaining data points).
- We define n_{train} and n_{test} as follows:

$$n_{train} = \text{number of data points in the training set} \tag{1}$$
$$n_{test} = \text{number of data points in the test set} \tag{2}$$

- We define y_{train} and y_{test} as follows:

$$\boldsymbol{Y}_{train} = \text{response variables for the training set} \tag{3}$$
$$\boldsymbol{Y}_{test} = \text{response variables for the test set} \tag{4}$$

(a) Naive Regression

- Let $\boldsymbol{C_1}$ be a vector of ones of length n_{train}. We can then compute the vector of coefficients w using the general formula for linear regression:

$$\boldsymbol{w} = (\boldsymbol{C_1}^T\boldsymbol{C_1})^{-1}\boldsymbol{C_1}^T\boldsymbol{Y}_{train} \tag{5}$$

- We can then use $\boldsymbol{w}$ to compute the mean square error (MSE):

$$MSE_{train} = \frac{||\boldsymbol{Y}_{train} - \boldsymbol{w}\boldsymbol{C_1}||^2}{n_{train}} \tag{6}$$
$$MSE_{test} = \frac{||\boldsymbol{Y}_{test} - \boldsymbol{w}\boldsymbol{C_1}||^2}{n_{test}} \tag{7}$$

- Using this method, we obtain the following results:
  - MSE on the training set over 20 runs: 83.01
  - MSE on the test set over 20 runs: 87.46.

(b) Interpretation of constant function

The constant function is equivalent to the **mean value of column 13 for the training set**.

This can be derived from the expression used to determine the regression coefficients: the first factor (X^TX) is the inverse of the number of observations in the training set, whereas the second term (X^TY) is the sum of the values of column 13 (i.e. median house price) in the training set. In other words, we divide the total by the number of observations, which is equivalent to calculating the mean.

(c) Linear Regression with single attributes

- For linear regression with a single feature k ($k \in 1, 12$), the feature matrix X_k is matrix of dimension (n_{train}, 2):
  - the first column of X_k is a column of 1's, corresponding to the bias term
  - the second column consists of the values of the relevant attribute for all data points from the training set.

- We can the compute the vector of coefficients $\boldsymbol{w}$ using the same method as above:

$$\boldsymbol{w} = (X_k^T X_k)^{-1} X_k^T \boldsymbol{Y}_{train} \tag{8}$$

- Based on $\boldsymbol{w}$ we can compute the MSE on the training and test sets as follows:

$$MSE_{train} = \frac{||\boldsymbol{Y}_{train} - X_k \boldsymbol{w}||^2}{n_{train}} \tag{9}$$

$$MSE_{test} = \frac{||\boldsymbol{Y}_{test} - X_k^{(test)} \boldsymbol{w}||^2}{n_{test}} \tag{10}$$

  Where $X_k^{(test)}$ is defined analogously to X_k, using the features of the data points from the test set instead of the training set.

- We obtain the following results:
  - Attribute 1 (CRIM)
    - Training MSE over 20 runs: 69.30
    - Test MSE over 20 runs: 77.31
  - Attribute 2 (ZN)
    - Training MSE over 20 runs: 71.50
    - Test MSE over 20 runs: 77.79
  - Attribute 3 (INDUS)
    - Training MSE over 20 runs: 65.96
    - Test MSE over 20 runs: 62.47
  - Attribute 4 (CHAS)
    - Training MSE over 20 runs: 81.06
    - Test MSE over 20 runs: 83.75
  - Attribute 5 (NOX)
    - Training MSE over 20 runs: 70.83
    - Test MSE over 20 runs: 65.61
  - Attribute 6 (RM)
    - Training MSE over 20 runs: 44.81
    - Test MSE over 20 runs: 41.55
  - Attribute 7 (AGE)
    - Training MSE over 20 runs: 69.09
    - Test MSE over 20 runs: 79.59
  - Attribute 8 (DIS)
    - Training MSE over 20 runs: 80.00
    - Test MSE over 20 runs: 77.85
  - Attribute 9 (RAD)
    - Training MSE over 20 runs: 73.66
    - Test MSE over 20 runs: 69.44
  - Attribute 10 (TAX)
    - Training MSE over 20 runs: 66.08
    - Test MSE over 20 runs: 65.86
  - Attribute 11 (PTRATIO)

        - ∗ Training MSE over 20 runs: 63.38
        - ∗ Test MSE over 20 runs: 61.65
    - Attribute 12 (LSTAT)
        - ∗ Training MSE over 20 runs: 38.62
        - ∗ Test MSE over 20 runs: 38.59

- Taken in isolation, Attributes 6 and 12 provide the most accurate prediction for the value of the 13th column.

(d) Linear Regression using all attributes

- We follow a similar as above, except that the feature matrices X_{train} and X_{test} now have 13 columns:
    - A column of ones
    - One column for each feature.
- We use the same steps as above to compute the vector of coefficients $\boldsymbol{w}$ based on X_{train} and $\boldsymbol{Y}_{train}$:

$$\boldsymbol{w} = (X_{train}^T X_{train})^{-1} X_{train}^T \boldsymbol{Y}_{train} \tag{11}$$

- We can then compute the MSEs:

$$MSE_{train} = \frac{||\boldsymbol{Y}_{train} - X_{train}\boldsymbol{w}||^2}{n_{train}} \tag{12}$$

$$MSE_{test} = \frac{||\boldsymbol{Y}_{test} - X_{test}\boldsymbol{w}||^2}{n_{test}} \tag{13}$$

- We obtain the following results:
    - Training MSE over 20 runs: 21.98
    - Test MSE over 20 runs: 24.99

  The MSEs for both the training and test sets are lower than for any of the models based on a single feature.

5 Question 5

(a) Cross-validation

Method: For the cross-validation, we split the training set into 5 random folds of same size. Then, for each pairing of γ and σ, we proceed as follows:

- Step 1: select a fold f. This fold f will be referred to as validation fold in the next steps. The remaining 4 folds are the training folds.
- Step 2: compute the Kernel matrix based on all data points from the training folds, using the given value of sigma. The Kernel matrix is a square matrix with a number of rows and column equal to the number of data points in the training folds. The elements of the Kernel matrix are determined by the Kernel of all pair-wise combinations of points in the training folds.
- Step 3: compute the alpha vector, using the given value of gamma.
- Step 4: for every data point from the **validation** fold, use the alpha vector to compute the predicted value of the response variable. Then compute and retain the prediction error, ie the absolute value of the difference between the prediction obtained and the true value of the response variable.
- Step 5: repeat Steps 1 through 4 for every fold.
- Step 6: use the values of all prediction errors to compute the mean error for a given pairing of γ and σ.

Using this method, we obtain a minimum mean error of 2.46 with

$$\gamma = 2^{-26} \tag{14}$$

$$\sigma = 2^{8} \tag{15}$$

(b) Plotting the cross-validation error

To obtain the cross-validation error, we divide the mean error above by the number of folds, i.e. 5.

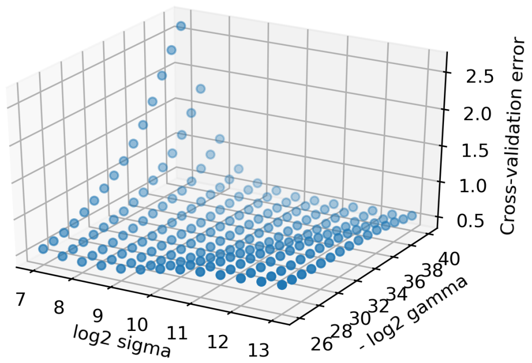

Figure 10: Cross-validation error for all pairings of γ and σ

(c) MSE for best values of γ and σ

Using the values mentioned above for gamma and sigma, we now repeat the kernel ridge regression on the training and test sets.

- Step 1: compute the Kernel matrix based on all data points from the training set, using $\sigma = 2^8$.
- Step 3: compute the alpha vector, using $\gamma = 2^{-26}$.
- Step 4: for every data point from the **training** set, use the alpha vector to compute the predicted value. Then compute and retain the squared error.
- Step 5: compute the mean squared error on the training set.
- Step 6: repeat Steps 4 to 5 with data points from the test set.

We obtain the following values:

- MSE on training set is 6.85
- MSE on test set is 13.21

(d) Comparison of all methods

Using the Boston housing data, we generate 20 different random splits into training and test set. For every split, we repeat the methods described above and compute the MSE on the training and test set.

After the 20 runs, we obtain the following values for the mean and standard deviation of the MSE:

Method	MSE train	MSE test
Naive Regression	83.19 ± 4.38	87.02 ± 8.83
Linear regression (attribute 1)	70.37 ± 4.02	75.02 ± 8.08
Linear regression (attribute 2)	71.91 ± 4.86	76.94 ± 9.86
Linear regression (attribute 3)	62.88 ± 5.20	68.62 ± 10.54
Linear regression (attribute 4)	80.59 ± 4.05	84.98 ± 8.30
Linear regression (attribute 5)	67.53 ± 5.19	72.26 ± 10.46
Linear regression (attribute 6)	41.49 ± 3.86	48.29 ± 7.81
Linear regression (attribute 7)	70.64 ± 5.20	76.32 ± 10.60
Linear regression (attribute 8)	77.49 ± 5.29	82.94 ± 10.92
Linear regression (attribute 9)	70.35 ± 4.39	76.07 ± 8.90
Linear regression (attribute 10)	64.07 ± 5.00	69.97 ± 10.18
Linear regression (attribute 11)	61.36 ± 2.45	65.65 ± 5.01
Linear regression (attribute 12)	38.31 ± 2.47	39.07 ± 4.94
Linear regression (all attributes)	21.37 ± 1.61	26.01 ± 3.55
Kernel Ridge Regression	7.81 ± 1.66	11.66 ± 1.97

6 Question 6

(a)

We define the predictor function f such that $f(x) = \hat{y}$. The error rate of f with respect to the loss function L_c is equal to:

$$\begin{aligned} \epsilon(f) &= \sum_{x,y} L_c(y, f(x))p(x,y) \\ &= \sum_{x \in X} \left\{ \sum_{y \in Y} L_c(y, f(x))p(y|x) \right\} p(x) \end{aligned} \tag{16}$$

The Bayes estimator is the function f^* that minimises the error rate:

$$f^* = \text{argmin}_f \epsilon(f) \tag{17}$$

Minimising the error rate for a given value of x, denoted as x', is equivalent to minimising the following expression:

$$\sum_{y \in Y} L_c(y, f(x'))p(y|x') \tag{18}$$

$$= \sum_{y \in Y} [y \neq f(x')] c_y p(y|x') \tag{19}$$

$$= [1 \neq f(x')]c_1 Pr(y = 1|x') + [2 \neq f(x')]c_2 Pr(y = 2|x') + ... + [k \neq f(x')]c_k Pr(y = k|x') \tag{20}$$

The factor $[y \neq f(x')]$ is equal to 0 when $[y = f(x')]$, and equal to 1 for all other values of y. In other words, one of the terms in the above expression will be equal to 0, whereas the remaining terms will be equal to $c_y p(y|x')$.

To minimise the sum, we therefore need to predict $f(x')$ such that $[y \neq f(x')]$ is equal to 0 (i.e. $y = f(x')$) when the factor $c_y p(y|x')$ takes its maximum value.

Therefore the Bayes' estimator is given by:

$$f^*(x) = \arg\max_y c_y p(y|x)$$

(b)

(i)

If $F(x) = x^2$, then $F'(x) = 2x$. The loss function becomes:

$$\begin{aligned} L_F(y, \hat{y}) &= \hat{y}^2 - y^2 + (y - \hat{y})2y \\ &= \hat{y}^2 - 2y\hat{y} + y^2 \\ &= (\hat{y} - y)^2 \end{aligned} \tag{21}$$

In this case, the loss function is the squared error.

(ii)

We begin by proving that, if F is strictly convex differentiable, then $F'(x)$ is strictly increasing.

We consider any two numbers $a, b \in \mathbb{R}$. We need to show that:

$$a < b \Rightarrow F'(a) < F'(b) \tag{22}$$

Since F is strictly convex, we know that:

$$F(\alpha a + (1-\alpha)b) < \alpha F(a) + (1-\alpha)F(b), \forall \alpha : 0 < \alpha < 1 \tag{23}$$

The slope of the straight line connecting a and b is equal to $\frac{F(b)-F(a)}{b-a}$.

In order to show that $F'(a) < F'(b)$, it is possible to prove by contradiction that the slope is greater than $F'(a)$ and less than $F'(b)$.

For the proof, we assume first that $F'(a)$ is greater than the slope. This means that:

$$lim_{h\to 0^+} \frac{F(a+h)-F(a)}{h} \geq \frac{F(b)-F(a)}{b-a} \tag{24}$$

(We only consider positive values of h, since we are interested in values that lie between a and b.)
Multiplying both sides by h (positive) and adding $F(a)$, we obtain:

$$lim_{h\to 0^+} F(a) + h\frac{F(a+h)-F(a)}{h} \geq lim_{h\to 0^+} F(a) + h\frac{F(b)-F(a)}{b-a} \tag{25}$$

$$\Rightarrow lim_{h\to 0^+} F(a+h) \geq lim_{h\to 0^+} [(1-\frac{h}{b-a})F(a) + \frac{h}{b-a}F(b)] \tag{26}$$

At the same time, since h is strictly positive and less than b-a (positive based on the definition of a and b) when it tends to 0^+, we can say that: $0 < 1 - \frac{h}{b-a} < 1$
Therefore, $1 - \frac{h}{b-a}$ meets the definition of α for which inequality (3) should hold.
Replacing α with $1 - \frac{h}{b-a}$ in (3), we obtain:

$$F(1 - \frac{h}{b-a}a + \frac{h}{b-a}b) < (1 - \frac{h}{b-a})F(a) + \frac{h}{b-a}F(b) \tag{27}$$

$$\Rightarrow F(a - \frac{ah}{b-a} + \frac{bh}{b-a}) < (1 - \frac{h}{b-a})F(a) + \frac{h}{b-a}F(b) \tag{28}$$

$$\Rightarrow F(a + \frac{h(b-a)}{b-a}) < (1 - \frac{h}{b-a})F(a) + \frac{h}{b-a}F(b) \tag{29}$$

$$\Rightarrow F(a+h) < (1 - \frac{h}{b-a})F(a) + \frac{h}{b-a}F(b) \tag{30}$$

The inequalities (6) and (10) are incompatible. Therefore, we reject the initial assumption that $F'(a)$ is greater than the slope, and conclude that:

$$F'(a) < \frac{F(b)-F(a)}{b-a} \tag{31}$$

Using the same approach, and evaluating $lim_{h\to 0^-} \frac{F(b+h)-F(b)}{h}$ we establish that:

$$F'(b) > \frac{F(b)-F(a)}{b-a} \tag{32}$$

From (11) and (12), we conclude that:

$$a < b \Rightarrow F'(a) < F'(b) \tag{33}$$

In other words, **F'(x) is a strictly increasing function.**

Prove that $y = \hat{y} \Leftrightarrow L_F(y, \hat{y}) = 0$ for strictly convex differentiable $F : \mathbb{R} \to \mathbb{R}$.

Step 1: prove that $y = \hat{y} \Rightarrow L_F(y, \hat{y}) = 0$
Replacing $\hat{y}$ with y, it is easy to show that $L_F(y, \hat{y}) = F(y) - F(y) + 0 * F'(y) = 0$.

Step 2: prove that $L_F(y,\hat{y}) = 0 \Rightarrow y = \hat{y}$

We prove this by contradiction, assuming first that $L_F(y,\hat{y}) = 0$, but that $y < \hat{y}$. In that case, based on the proof above, we know that $F'(y)$ is less than the slope between y and $\hat{y}$:

$$F'(y) < \frac{F(\hat{y}) - F(y)}{\hat{y} - y} \tag{34}$$

Multiplying by $\hat{y} - y$ (which is positive since $y < \hat{y}$ by assumption), we obtain:

$$\begin{aligned} F'(y)(\hat{y} - y) &< F(\hat{y}) - F(y) && (35)\\ \Rightarrow 0 &< F(\hat{y}) - F(y) - F'(y)(\hat{y} - y) && (36)\\ \Rightarrow 0 &< F(\hat{y}) - F(y) + F'(y)(y - \hat{y}) && (37)\\ \Rightarrow 0 &< L_F(y,\hat{y}) && (38) \end{aligned}$$

This contradicts the initial assumption that $L_F(y,\hat{y}) = 0$. We therefore conclude that, if $L_F(y,\hat{y}) = 0$, then $\hat{y}$ cannot be greater than y.

Using the same method, we find that, with F convex:

$$y > \hat{y} \Rightarrow 0 > L_F(y,\hat{y}) \tag{39}$$

Therefore, if $L_F(y,\hat{y}) = 0$, then y cannot be greater than $\hat{y}$.

We conclude that $L_F(y,\hat{y}) = 0 \Rightarrow y = \hat{y}$.

From steps 1 and 2, we conclude that:

$$y = \hat{y} \Leftrightarrow L_F(y,\hat{y}) = 0 \tag{40}$$

(iii)

We distinguish three possibilities:

- $y = \hat{y}$
- $y > \hat{y}$
- $y < \hat{y}$

If $y = \hat{y}$, then we know from (ii) above that $L_F(y,\hat{y}) = 0$.

If $y < \hat{y}$, then we know from the proof in (ii) that $F'(y)$ is less than the slope of the straight line connecting y and $\hat{y}$:

$$F'(y) < \frac{F(\hat{y}) - F(y)}{\hat{y} - y} \tag{41}$$

As we showed under (ii) above, this implies that $L_F(y,\hat{y}) > 0$.

If $y > \hat{y}$ (i.e. $y - \hat{y} > 0$), we know from the proof above that $F'(y)$ is greater than the slope of the straight line connecting $\hat{y}$ and y:

$$\begin{aligned} F'(y) &> \frac{F(y) - F(\hat{y})}{y - \hat{y}} && (42)\\ \Rightarrow F'(y)(y - \hat{y}) &> F(y) - F(\hat{y}) && (43)\\ \Rightarrow F(\hat{y}) - F(y) + F'(y)(y - \hat{y}) &> 0 && (44)\\ \Rightarrow L_F(y,\hat{y}) &> 0 && (45) \end{aligned}$$

In all three scenarios, we find that $L_F(y,\hat{y}) \geq 0$.

(iv)

We define the predictor function f such that $f(x) = \hat{y}$. The error rate of f with respect to the loss function L_F is equal to:

$$\begin{aligned}\epsilon(f) &= E[F(f(x)) - F(y) + (y - f(x))F'(y)] \\ &= \sum_{x\in X}\sum_{y\in Y}[F(f(x)) - F(y) + (y - f(x))F'(y)]p(x,y)\end{aligned} \tag{46}$$

The Bayes estimator is the function f^* that minimises the error rate:

$$f^* = \text{argmin}_f \epsilon(f) \tag{47}$$

Using Bayes' theorem, we can rewrite $\epsilon(f)$ as follows:

$$\begin{aligned}\epsilon(f) &= \sum_{x\in X}\sum_{y\in Y}[F(f(x)) - F(y) + (y - f(x))F'(y)]p(x,y) \\ &= \sum_{x\in X}\left\{\sum_{y\in Y}[F(f(x)) - F(y) + (y - f(x))F'(y)]p(y|x)\right\}p(x)\end{aligned} \tag{48}$$

We now attempt to minimise $\epsilon(f)$ at a given value of x, denoted as x'. We denote $f(x')$ as z, and the expected error at x' as e. $p(x')$ being a constant, e is proportional to $\sum_{y\in Y}[F(z)-F(y)+(y-z))F'(y)]p(y|x')$.

Therefore, in order to compute $f^*(x')$, we need to determine the value of z that minimises e. By definition, F is strictly convex differentiable, and we can therefore differentiate e with respect to z.

$$\begin{aligned}\frac{\partial e}{\partial z} &= \frac{\partial}{\partial z}\sum_{y\in Y}[F(z) - F(y) + (y - z))F'(y)]p(y|x') \\ &= \sum_{y\in Y}[F'(z) - F'(y)]p(y|x')\end{aligned} \tag{49}$$

Setting the derivative to zero, we obtain:

$$\frac{\partial e}{\partial z} = 0 \tag{50}$$

$$\begin{aligned}&\Rightarrow \sum_{y\in Y}[F'(z) - F'(y)]p(y|x') = 0 \\ &\Rightarrow F'(z) = E[F'(y)|x'] \\ &\Rightarrow z = (F')^{-1}(E[F'(y)|x'])\end{aligned} \tag{51}$$

Therefore, the Bayes' estimator which minimises the error rate is given by

$$f^* = (F')^{-1}(E[F'(y)|x']) \tag{52}$$

For the special case $F(x) = |x|^p$, $p > 1$, we have:

$$\begin{aligned}F'(x) &= px^{p-1} \text{ if } x > 0 \\ F'(x) &= p(-x)^{p-1} \text{ if } x < 0\end{aligned} \tag{53}$$

The inverse is given by:

$$\begin{aligned}(F')^{-1}(x) &= \sqrt[p-1]{\frac{x}{p}} \text{ if } x > 0 \\ (F')^{-1}(x) &= -\sqrt[p-1]{\frac{x}{p}} \text{ if } x < 0\end{aligned} \tag{54}$$

Plugging these into Equation 52, we get the Bayes' estimator:

$$\begin{aligned}f^* &= (F')^{-1}(E[F'(y)|x']) \\ &= \sqrt[p-1]{\frac{\sum_{y \in Y} px^{p-1}P(y|x')}{p}} \\ &= \sqrt[p-1]{\sum_{y \in Y} x^{p-1}P(y|x')}\end{aligned} \tag{55}$$

Using this formula to calculate the Bayes' estimator for the square loss function in Equation (i), where $F(x) = x^2$, will give us some confidence that our general Bayes' estimator for the F-loss is correct.

The Bayes' estimator for the square loss is known to be $E(y|x)$, and choosing $p = 2$, we have $F(x) = |x|^2 = x^2$. Hence, we can use Equation 55 with $p = 2$ and check it equals $E(y|x)$:

$$\begin{aligned}f^* &= \sqrt[2-1]{\sum_{y \in Y} y^{2-1}P(y|x')} \\ &= \sum_{y \in Y} yP(y|x') \\ &= E(y|x)\end{aligned} \tag{56}$$

Therefore, we gain confidence that our solution for the Bayes' estimator for general $F(x) = |x|^p$ is correct.

7 Question 7

(a)

Answer: The function $K_c(x,z)$ is a positive semi-definite kernel if and only if $c \geq 0$.

Proof:

Step 1: We begin by showing that $K_c(x,z)$ is a positive semi-definite kernel for any $c \geq 0$.
We define the kernel matrix K as the square (k-by-k) matrix such that:

$$K_{ij} = K_c(x_i, x_j) : i,j = 1,...,k, k \in \mathbb{N}, x_i, x_j \in \mathbb{R}^n \tag{57}$$

For any k, x_i and x_j, we also define:

- K' as the matrix such that:
$$K'_{ij} =< x_i, x_j >: i,j = 1,...,k \tag{58}$$

- C as the matrix such that:
$$C_{ij} = c : i,j = 1,...,k \tag{59}$$

To prove that $K_c(x,z)$ is a positive semi-definite kernel, we need to show that, for any $k \in \mathbb{N}$ and $x_i, x_j \in \mathbb{R}^n$, K is a positive semi-definite matrix. This is equivalent to showing that:

$$z^T K z \geq 0, \forall z \in \mathbb{R}^k \tag{60}$$

From the definition of the function K_c, we can observe that $(i,j)^{th}$ element of K is equal to the $(i,j)^{th}$ element of K' plus the constant c.
Considering that K, K' and C all have the same dimension, we can rewrite $z^T K z$ as follows:

$$\begin{aligned} z^T K z &= z^T (K' + C) z \\ &= z^T K' z + z^T C z \end{aligned} \tag{61}$$

Since K' is the matrix corresponding to the linear kernel, we know that it is positive semi-definite and therefore $z^T K' z \geq 0$.
Next, we need to determine the sign of $z^T C z$.

$$\begin{aligned} z^T C z &= c z^T C_1 z \qquad \text{(where } C_1 \text{ is a k-by-k matrix of ones} \\ &= c z^T \begin{pmatrix} z_1 + z_2 ... + z_k \\ z_1 + z_2 ... + z_k \\ ... \\ z_1 + z_2 ... + z_k \end{pmatrix} \\ &= c[z_1(\sum_{i=1}^k z_i) + z_2(\sum_{i=1}^k z_i) ... + z_n(\sum_{i=1}^k z_i)] \\ &= c(\sum_{i=1}^k z_i)^2 \end{aligned} \tag{62}$$

We conclude that, if $c \geq 0$, then $z^T C z \geq 0$.
For $c \geq 0$, $z^T K z$ is the sum of two positive numbers, and is therefore positive. This means that K is a positive semi-definite matrix, and (x,z) is a positive semi-definite function.

Step 2: In order to show that $K(x,z)$ is positive semi-definite only for positive values of c, we need to prove that if $c < 0$, then K is not positive semi-definite.

For $k = 1$ and x_1 equal to the zero vector, the kernel matrix K (as defined above) is a one-by-one matrix with a single element equal to c. If $c < 0$, that matrix is not positive semi-definite, and therefore K(x,z) is not a positive semi-definite kernel.

Conclusion: We have shown that the function $K_c(x, z)$ is a positive semi-definite kernel if and only if $c \geq 0$.

(b)

A kernel can be viewed as a measure of similarity. $K_c(x, z)$ is higher when x and z are similar than when they are different. $K_c(x, z)$ reaches its minimum value when x and z are orthogonal, at which point the dot product is equal to zero and $K_c(x, z)$ is therefore equal to c.

In the context of linear regression, this generally means that data points that are more similar to the one for which we are predicting an output variable will be given more weight.

However, as c increases, its relative weight in the value of $K_c(x, z)$ increases, and, conversely, the dot product between vectors has less and less impact on the value of $K_c(x, z)$ (in relative terms). In other words, increasing c has the effect of reducing the weight of the measure of similarity. This means that the predictions will become smoother, and eventually get closer to an average of the existing data, rather than making predictions based on the similarity with individual data points.

8 Question 8

Let the k-th example from the data set be the nearest neighbour of t. Accordingly, the Euclidean norm of $x_i - t$ is the smallest for $i = k$:

$$
\begin{aligned}
&|||x_k - t|| < ||x_i - t|| && \forall i \in \{0, m\} - k \\
&\Rightarrow ||x_k - t||^2 < ||x_i - t||^2 \\
&\Rightarrow -\beta||x_k - t||^2 > -\beta||x_i - t||^2 && \forall \beta > 0 \\
&\Rightarrow exp(-\beta||x_k - t||^2) > exp(-\beta||x_i - t||^2) \\
&\Rightarrow K_\beta(x_k, t) > K_\beta(x_i, t)
\end{aligned}
$$

Accordingly, the value of the kernel $K_\beta(x_i, t)$ is the highest for $i = k$. In addition, we can observe that increasing β has the effect of amplifying the difference between $K_\beta(x_k, t)$ and $K_\beta(x_i, t)$ for other values of i. In other words, the relative weight of the nearest neighbour will strictly increase as a function of β. (By contrast, if β were to tend to 0, all the kernels would tend to 1.)

The Gram matrix for the Gaussian kernel is given by

$$
K_\beta(\boldsymbol{x}, \boldsymbol{x}) = \begin{pmatrix} 1 & exp(-\beta||x_1 - x_2||^2) & ... & exp(-\beta||x_1 - x_m)||^2) \\ exp(-\beta||x_2 - x_1)||^2) & 1 & ... & ... \\ ... & ... & 1 & ... \\ exp(-\beta||x_m - x_1)||^2) & ... & ... & 1 \end{pmatrix}
$$

This is a positive definite matrix. Its inverse is therefore a symmetrical matrix that can be written as:

$$
K_\beta(\boldsymbol{x}, \boldsymbol{x})^{-1} = c * \begin{pmatrix} 1 & c_{12} & ... & c_{1m} \\ c_{21} & 1 & ... & c_{2m} \\ ... & ... & 1 & ... \\ c_{m1} & ... & ... & 1 \end{pmatrix}
$$

(where c is a normalising constant for the diagonal elements, and all elements c_ij $(i, j \in [1, m])$ are constants that depend on $\boldsymbol{x_i}$ and β)

As β increases, we note that all non-diagonal elements of the diagonal matrix tend to zero, and the Gram matrix therefore tends to the identity matrix. If a matrix tends to the identity matrix, the same is necessarily true for its inverse. We can therefore find a value of β such that the normalising constant c is close to 1, and therefore positive for any larger values of β.

We can use the inverse of the Gram matrix to compute the α vector:

$$
\alpha_\beta = c \begin{pmatrix} 1 & c_{12} & ... & c_{1m} \\ c_{21} & 1 & ... & c_{2m} \\ ... & ... & 1 & ... \\ c_{m1} & ... & ... & 1 \end{pmatrix} \times \begin{pmatrix} y_1 \\ y_2 \\ ... \\ y_m \end{pmatrix}
$$

$$
= \text{c} \begin{pmatrix} y_1 + y_2 \cdot c_{12} + ... + y_m \cdot c_{1m} \\ y_1 \cdot c_{21} + y_2 + ... + y_m \cdot c_{2m} \\ ... \\ y_1 \cdot c_{m1} + y_2 \cdot c_{m2} + ... + y_m \end{pmatrix}
$$

We can then insert this expression into the formula for $f(t)$:

f(t) $= \sum_{i=1}^{m} \alpha_i K_\beta(\boldsymbol{x_i}, \boldsymbol{t})$

= c[(y$_1$ $+ y_2 \cdot c_{12} + ... + y_n \cdot c_{1m}) \cdot K_\beta(\boldsymbol{x_1}, \boldsymbol{t}) + (y_1 \cdot c_{21} + y_2 + ... + y_m \cdot c_{2m}) \cdot K_\beta(\boldsymbol{x_2}, \boldsymbol{t})$ $+ ... + (y_1 \cdot c_{m1} + y_2 \cdot c_{m2} + ... + y_m) \cdot K_\beta(\boldsymbol{x_n}, \boldsymbol{t})]$

Since the prediction is based on the sign of f(t), and c is positive, we can divide f(t) by c without changing the prediction. We also re-arrange the terms and group them as factors of all values of y:

f(t) = y$_1$$[K_\beta(\boldsymbol{x_1}, \boldsymbol{t}) + c_{21} \cdot K_\beta(\boldsymbol{x_2}, \boldsymbol{t}) + ... + \cdot c_{m1} \cdot K_\beta(\boldsymbol{x_m}, \boldsymbol{t}] + ... + y_k[c_{1k} \cdot K_\beta(\boldsymbol{x_1}, \boldsymbol{t}) + ... + K_\beta(\boldsymbol{x_k}, \boldsymbol{t}) +$ $... + c_{nk} \cdot K_\beta(\boldsymbol{x_m}, \boldsymbol{t})] + ... + y_n[c_{1m} \cdot K_\beta(\boldsymbol{x_1}, \boldsymbol{t}) + ... + K_\beta(\boldsymbol{x_m}, \boldsymbol{t})]$

Let u_i be the factor associated with y_i in the expression above. All values of u_i are functions of ($\boldsymbol{x_1}$, ... $\boldsymbol{x_m}$, $\boldsymbol{t}$).

For any values of ($\boldsymbol{x_1}$, $\boldsymbol{x_m}$, $\boldsymbol{t}$), there exists a function $\hat{\beta}$ to obtain a value of β_0 such that:
$|u_k| = \sum_{i \in [1,m]-k} |u_i|$

For β_0, the k-th example from the data set has the same weight as the combination of all other points. For any value of β greater than β_0, the weight of k-th will only increase (as we explained above), meaning that it will always 'prevail' and the prediction for t will be y_k.

9 Question 9

Let

$$S = \begin{pmatrix} M_{n^2-n+1} & M_{n^2-n+2} & ... & M_{n^2} \\ ... & ... & ... & ... \\ M_{n+1} & M_{n+2} & ... & M_{2n} \\ M_1 & M_2 & ... & M_n \end{pmatrix}$$

be a matrix representing the state of the board, where $M_i = 1$ if the mole is visible and $M_i = 0$ if the mole is hidden.

Let $w_i \in \{0, 1\}$ be an indicator function which is 1 if the mole in hole M_i is whacked, and 0 otherwise. This action makes the mole hide underground (setting $M_i := 0$) and flips the immediate four adjacent holes (setting $M_j := 1 - M_j$).

Since we know that $M_i \in \{0, 1\}$, these operations for updating the board state are equivalent to performing $M_j := M_j + 1 \bmod 2$ on the index that is whacked and its adjacent neighbours.

To the extent that whacking a mole flips a binary state of the mole and its neighbours, hitting the same mole twice (assuming it has 're-emerged' in the meantime) is a neutral move.

In addition, we can observe that the impact of a given move (i.e. flipping the state of a mole and its neighbours) is not affected by previous moves. Therefore, once a winning sequence has been identified, the order in which the fields are hit does not matter. In practice, the only constraint that may determine the order is that the player can only hit a hole with a mole that is not hiding.

We can write each hole state M_i as a sum of whacks (modulo 2). For example, looking at S above, we would have $M_1 = w_1 + w_{n+1} + w_2$, and $M_{n+2} = w_{n+2} + w_{2n+2} + w_{2n} + w_2 + w_{n+1}$.

This gives us a system of n^2 equations, with one for each hole M_i:

$$\begin{aligned} M_1 &= w_1 + w_2 + w_{n+1} \\ M_2 &= w_1 + w_2 + w_n + w_{n+2} \\ &\vdots \\ M_{n^2} &= w_{n^2-n} + w_{n^2-1} + w_{n^2} \end{aligned}$$

In order to solve this system of linear equations, we can the current state of the board in matrix form, as follows:

$$\begin{pmatrix} 1 & 1 & ... & 0 \\ 1 & 1 & ... & 0 \\ \vdots & \vdots & ... & \vdots \\ 0 & 0 & ... & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_{n^2} \end{pmatrix} = \begin{pmatrix} M_1 \\ M_2 \\ \vdots \\ M_{n^2} \end{pmatrix} \tag{63}$$

In the i-th row (representing the i-th field of the board), the following elements are equal to 1:

- (i, i)
- $(i, i + 1)$, unless i is a multiple of n (in which case (i,i+1) = 0).
- $(i, i - 1)$, unless $i - 1$ is a multiple of n (in which case $(i, i - 1) = 0$)
- $(i, i - n)$, unless $i \leq n$ (in which case $(i, i - n) = 0$)
- $(i, i + n)$, unless $i \geq n^2 - n$ (in which case $(i, i + n) = 0$)

All other elements of the matrix are equal to 0 (since the relevant fields are not affected by the corresponding w_i's).

For an n x n matrix, a system of linear equations can be solved in $O(n^3)$ time using Gaussian elimination. The matrix given by Equation 63 is n^2 x n^2, so assuming there is a solution, we can solve for the vector of whacks (w_i, $i \in [1, n^2]$) in $O(n^6)$ time, which is polynomial in n. If the board state is not reachable, there will be no solution to these equations.

The list of indices i where $w_i = 1$ correspond to the holes which we need to whack to empty the board. We can do this in any order with the constraint that whacks cannot be done to holes where a mole is currently hiding.