Supervised Learning - Assignment 1

21180859 and 21059917
Group: Yes

November 12, 2021

Contents

1 Introduction

2 Part I

In this section we will perform linear regression on the same dataset under different transformations and analyze our results. We define our point vector as $\mathbf{x_i} = x_{i1}, x_{i2}, ..., x_{in}$ where $\mathbf{x_i} \in R^n$. Each $\mathbf{x_i}$ has a corresponding y_i, where $y_i \in R$.

2.1 Linear Regression

In this subsection we will define X to be the $m \times n$ matrix:

$$X = \begin{pmatrix} x_{1,1} & x_{1,2} & ... & x_{1,n} \\ x_{2,1} & x_{2,2} & ... & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & ... & x_{m,n} \end{pmatrix}$$

after a feature map from $\phi : R^n \to R^k$, the transformed dataset can be defined as:

$$\Phi := \begin{pmatrix} \phi_1(x_1) & ... & \phi_k(x_1) \\ \vdots & \ddots & \vdots \\ \phi_1(x_m) & ... & \phi_k(x_m) \end{pmatrix}$$

with a column vector $y = y_1, y_2, ..., y_m$ According to the definition of LSE, the optimal $\mathbf{w}$ can be presented as:

$$\mathbf{w} = (X^T X)^{-1} X^T y$$

2.1.1 Question 1

By placing the dataset into each polynomial base of dimension k=1,2,3,4 for mapping, we will get four different curves.

(a) The plot with k polynomial bases can be seen below in Fig 1:

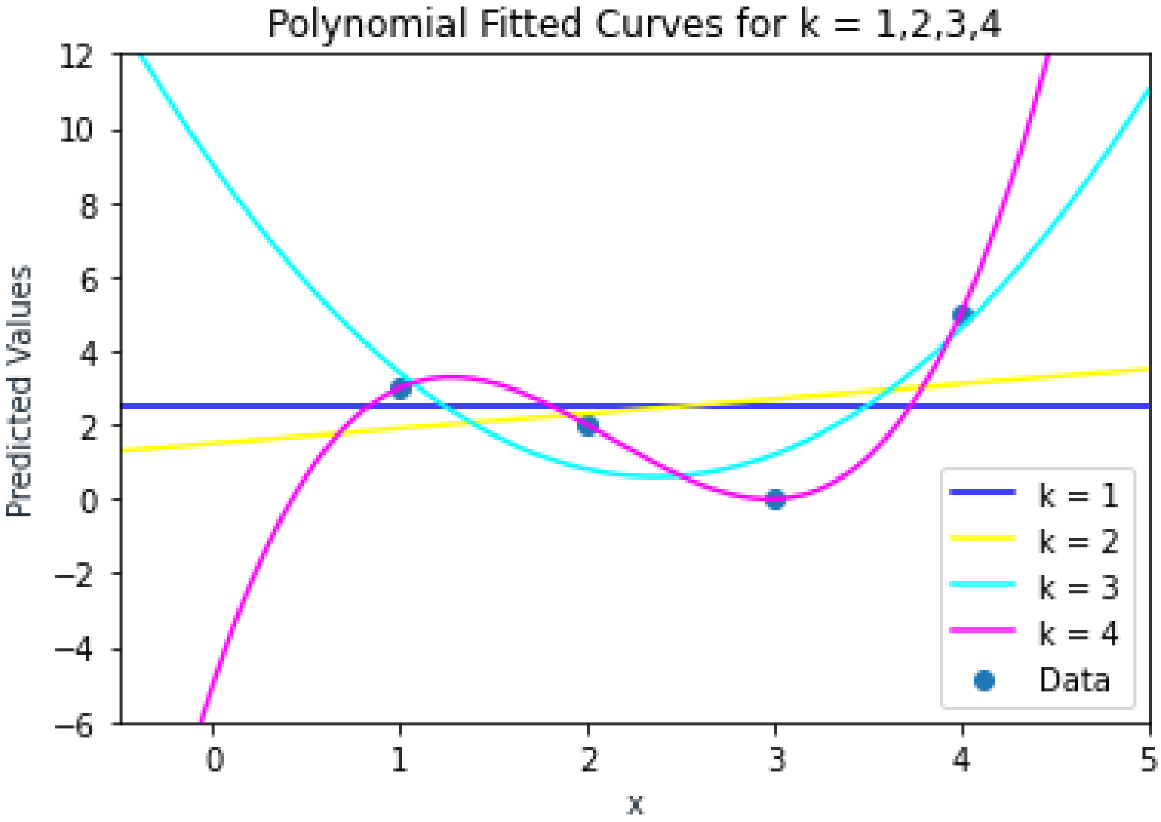

Fig 1: *Four different curves for k=1,2,3,4*

(b) Equations corresponding to the curves fitted for k=1,2,3 are:

$$
\begin{aligned}
k &= 1 : y = 2.5 \\
k &= 2 : y = 1.5 + 0.4x \\
k &= 3 : y = 9 - 7.1x + 1.5x^2
\end{aligned}
$$

(c) MSE of each fitted curve k=1,2,3,4:

$$
\begin{aligned}
MSE &= \frac{1}{m}(\mathbf{Xw\text{-}y})^T(\mathbf{Xw\text{-}y}) \\
k &= 1 : MSE = 3.25 \\
k &= 2 : MSE = 3.05 \\
k &= 3 : MSE = 0.8 \\
k &= 4 : MSE \approx 0
\end{aligned}
$$

2.1.2 Question 2

In this part phenomena of overfitting will be illustrated.

(a) Firstly we define:

$$g_\sigma(x) := sin^2(2\pi x) + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$ and x_i are sampled uniformly at random from the interval [0,1]. In this question we will sample x_i 30 times and apply $g_{0.07}(x_i)$.

(i) The plot of function $sin^2(2\pi x)$ in the range [0,1] is shown in Fig 2:

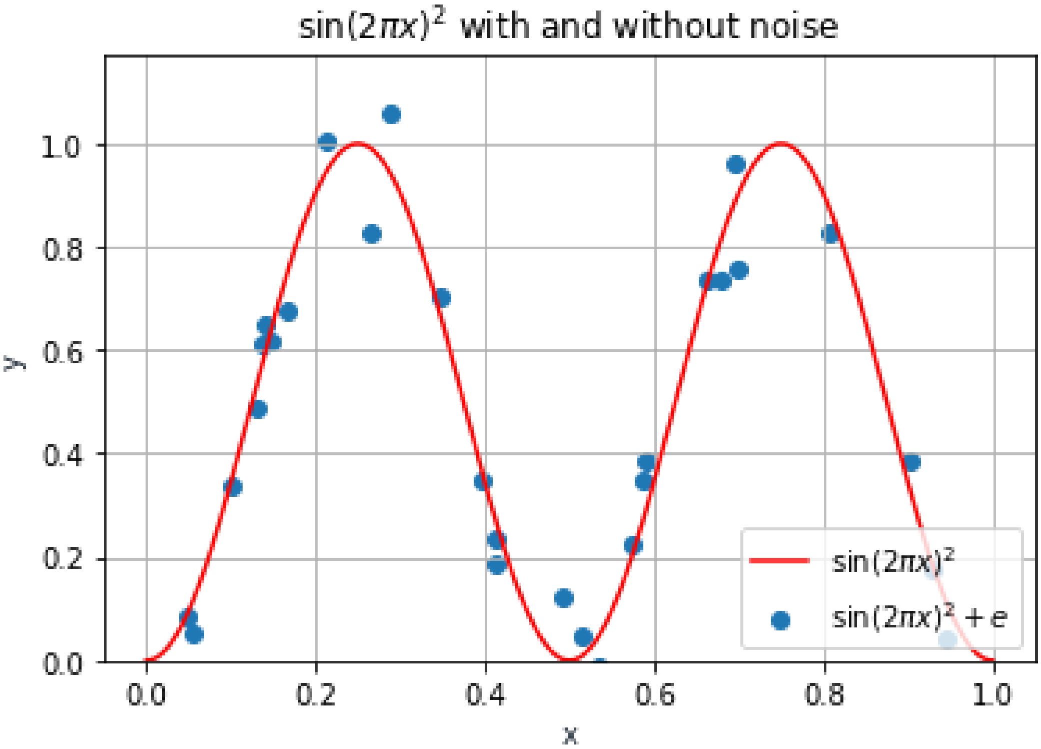

Fig 2: *Curve of* $sin^2(2\pi x)$

(ii) Here we fit the data set with a polynomial bases of dimension k=2,5,10,14,18 as well as a plot of data points. The plot is shown in Fig 3:

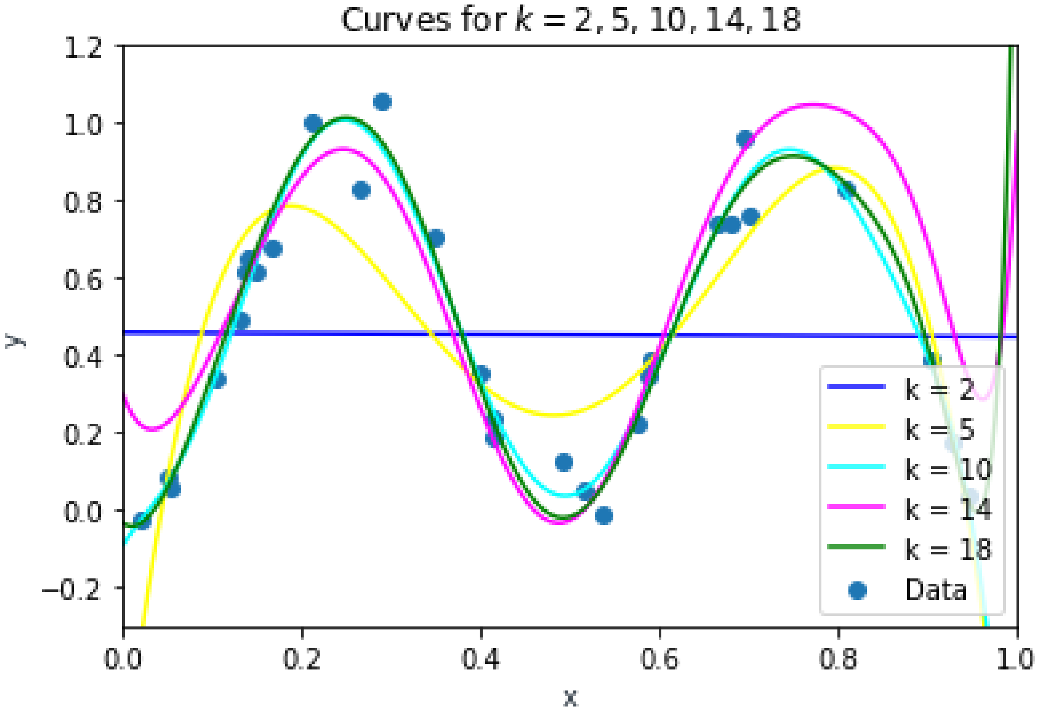

Fig 3: *Curve of mapped* $sin^2(2\pi x)$ *in k=2,5,10,14,18*

(b) First we calculated the MSE for each polynomial degree k from 1 to 18. We then calculated the logarithm of the MSE and plotted it as a function of the polynomial degree k, seen in Fig 4:

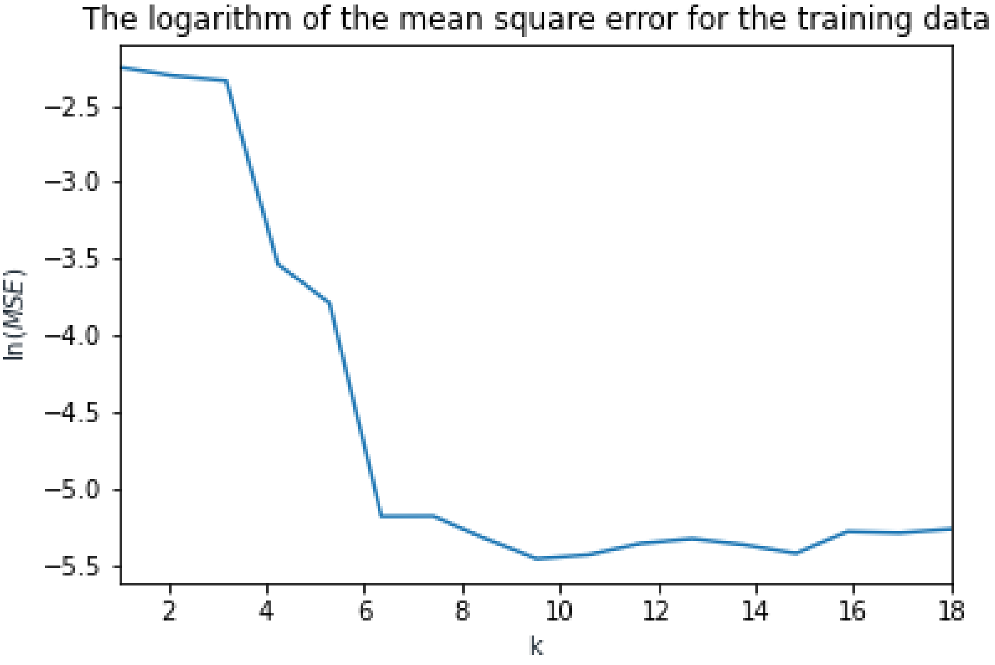

Fig 4: *The logarithm of the MSE for the training data*

As can be seen in Fig 4, there is an overall downward trend in MSE as k increases.

(c) We now sample x_i 1000 times and define $T = \{(x_1, g_{0.07}(x_1), ..., (x_{1000}, g_{0.07t}(x_1 000)\}$ to be our test set. We first train our model and then evaluate it with our test set using the same technique as in (b). The plot of the logarithm of the MSE against k can be seen in Fig 5:

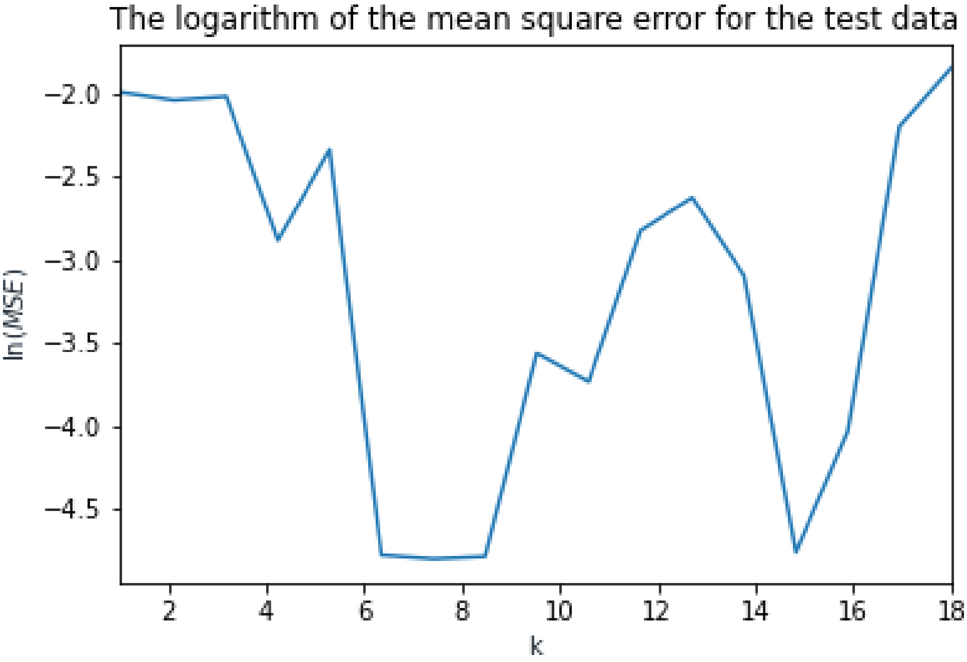

Fig 5: *The logarithm of the MSE for the test data*

It is evident that the error increases as the polynomial degree increases. This is due to overfitting - a phenomenon where the error is small when the model is evaluated using seen data and is large when evaluated using unseen data.

(d) We now do the same as in (b) and (c) except that we average the MSE over 20 runs. Note that we calculate $\ln \frac{\text{MSE}}{20}$. The plot can be seen in Fig 6:

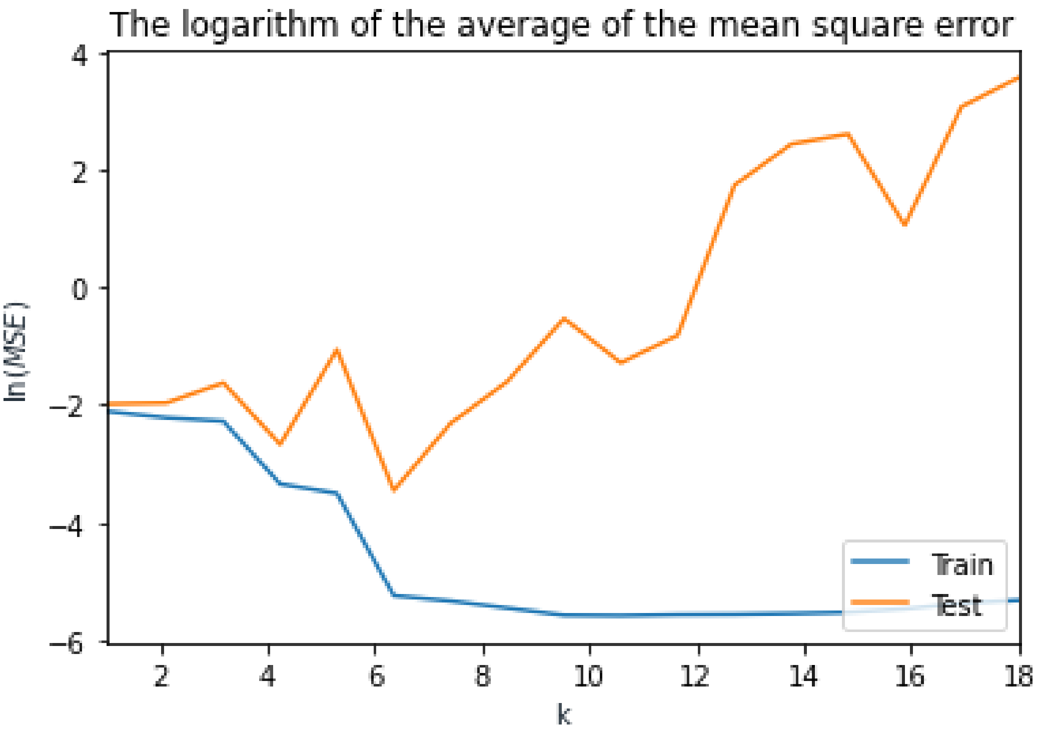

Fig 6: *The logarithm of the average of the MSE for the training and test data*

The plot shows us that on average the test error increases with k and the training error decreases.

2.1.3 Question 3

We now use the following basis:

$$\{\sin(\pi x), \sin(2\pi x), ..., \sin(k\pi x)\}$$

such that $k \in \{1, ..., 18\}$ and repeat the analysis in 2 (b-d).

We first fit train the model and calculate the training error. In Fig 7 is the logarithm of the MSE plotted against the degree of the polynomial:

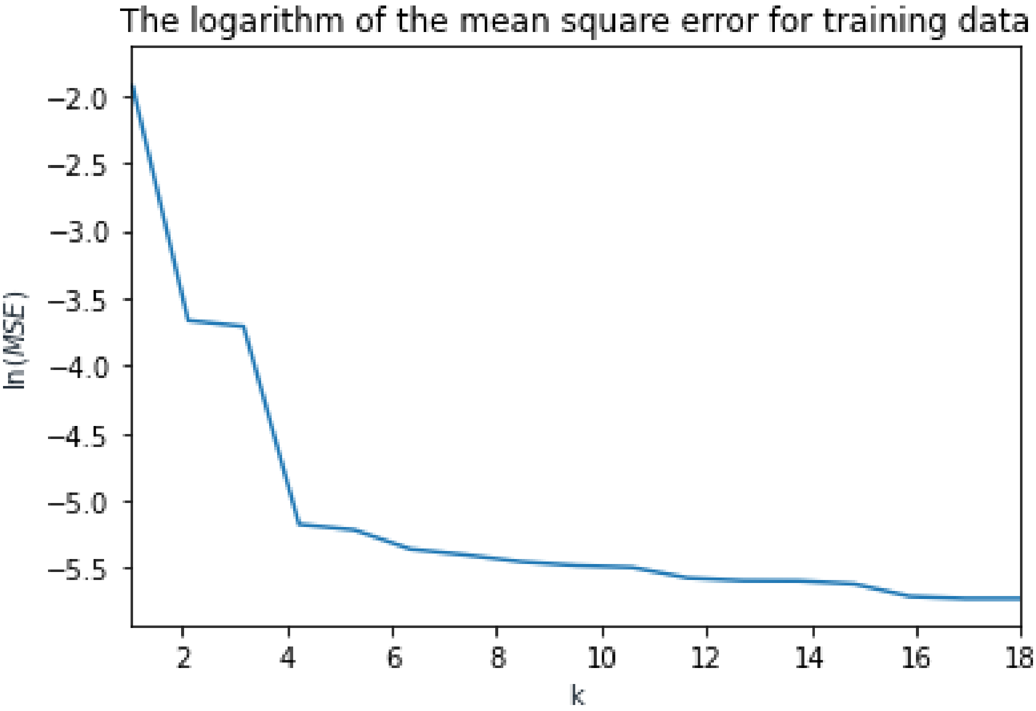

Fig 7: *The logarithm of the MSE for the training data*

In the plot we see that there is a clear downtrend as the polynomial degree k decreases.

We will now see what is the test error:

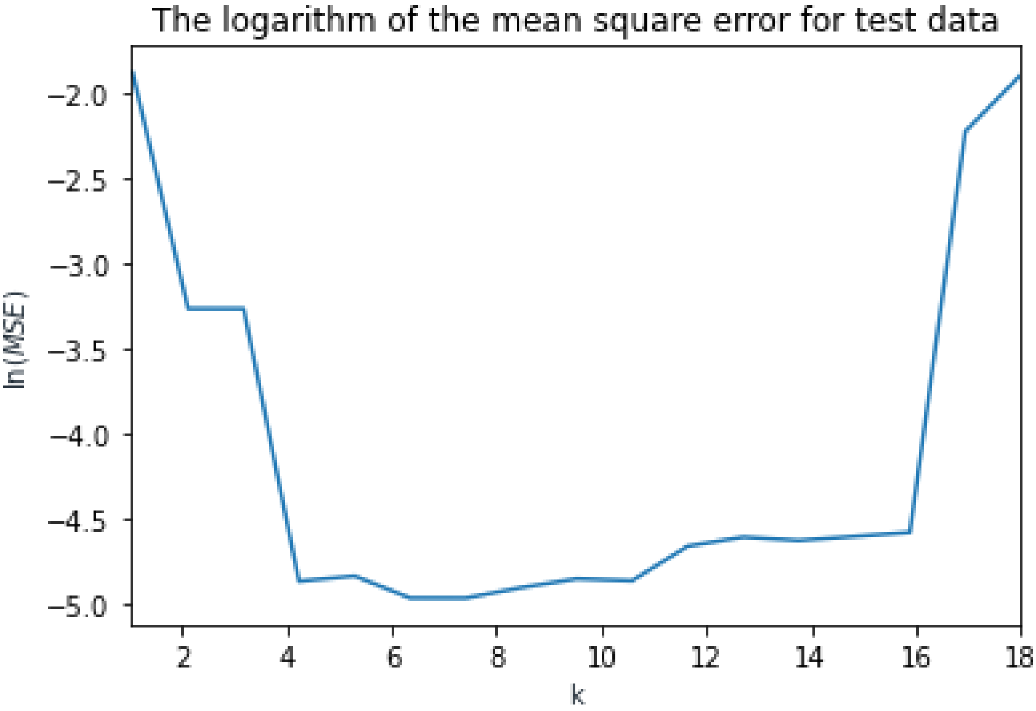

Fig 8: *The logarithm of the MSE for the test data*

At first as k increases the error decreases, however it starts to increase due to overfitting.

Lastly, we will see the average over 20 runs in Fig 8:

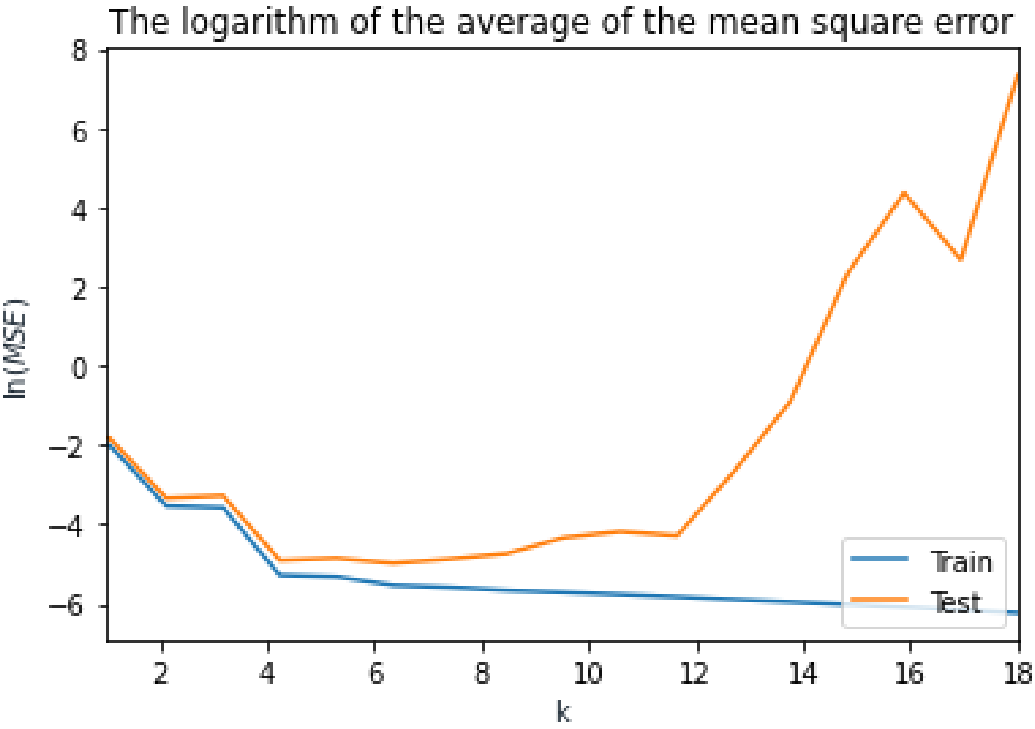

Fig 8: *The logarithm of the average of the MSE for the train and test data*

As expected, there is an indication of overfitting at higher values of k.

2.2 Filtered Boston housing and kernels

2.2.1 Question 4

In this section we will perform linear regression, where 'MEDV' is treated as the target variables and other variables as the predictors. We will be using 2/3 of the data as the training set and the remaining is the test set. We will first fit the target on a vector of ones, not using any of the predictors. Consequently, we will fit each predictor separately and finally we will use all predictors together.

(a) By using a vector of ones that is the same length of the training set and testing set respectively, we will fit the data with a constant function, then calculate the MSE on both the training and test sets:

$$training : MSE = 82.98542016747528$$

$$testing : MSE = 85.72325798887627$$

(b) We know that the optimal bayes estimator (when the loss function is the squared loss function) $f^*(x)$ is the expected value of y given the training data, i.e, $f^*(x) = E[y|x]$. Since x is a constant $f^*(x) = E[y]$. So our estimates are the mean of the target vector.

(c) We are now going to show our results of fitting each predictor separately with a bias term incorporated as well. We are going to show the values of the coefficients β_0 and β_1 from this equation:

$$y = \beta_0 + \beta_1 \text{predictor}$$

Table 1: Coefficient and MSE for each attribute

	CRIM	**ZN**	**INDUS**	**CHAS**	**NOX**	**RM**
beta_0	-0.3886	0.121062	-0.72599	5.717128	-33.1886	9.094418
beta_1	23.79606	21.1348	30.11669	22.24809	40.84501	-34.3391
mse_train	71.67505	79.45107	62.56643	81.91043	65.82976	49.78207
mse_test	67.65445	76.31561	72.39579	84.54537	60.14709	48.14559
	AGE	**DIS**	**RAD**	**TAX**	**PTRATIO**	**LSTAT**
beta_0	-0.1462	1.012527	-0.37105	-0.02441	-2.01504	-1.02289
beta_1	32.65194	18.68324	26.44481	31.9763	59.8812	35.5928
mse_train	69.34691	78.42583	77.79362	61.13353	67.6091	38.27399
mse_test	70.42391	73.80487	87.84329	63.85667	67.35361	33.83725

(d) We are now going to fit all of the predictors at once and see how they perform.

$$training : MSE = 24.41872832965957$$

$$testing : MSE = 26.20069052815768$$

It is evident that fitting all of the predictors at once is superior to fitting any of the predictors by their own.

2.2.2 Question 5

In this part we will do kernel ridge regression. For nonlinear regression, the dual representation in combination with the kernel trick will be used to predict the data.

For a given kernel function K define the kernel matrix $\mathbf{K}$ for a training set of size l elementwise via

$$K_{i,j} := K(\mathbf{x}_i, \mathbf{x}_j)$$

After kernelization the dual optimisation is:

$$\alpha^* = argmin_{\alpha \in R^l} \frac{1}{l} \sum_{i=1}^{l} (\sum_{j=1}^{l} \alpha_j K_{i,j} - y_i)^2 + \gamma \alpha^T K \alpha$$

dual optimisation formulation could also be represented as:

$$\alpha^* = (K + \gamma l I_l)^{-1} y$$

where I_l denotes the $l \times l$identity matrix. The evaluation of the regression function can be represented as:

$$y_{test} = \sum_{i=1}^{l} \alpha_i^* K(x_i, x_{test})$$

In this part we will perform kernel ridge regression with the Gaussian kernel,

$$K(x_i, x_j) = exp(-\frac{\| x_i - x_j \|^2}{2\sigma^2})$$

(a) Let $\gamma = [2^{-40}, 2^{-39}, ..., 2^{-26}]$ and $\sigma = [2^7, 2^{7.5}, ..., 2^{13}]$. We will now perform kernel ridge regression on the training set using five-fold cross-validation with all pair of α and σ and choose the pair which performs the best. By best performing we mean the pair (γ, σ) which yields the lowest MSE. Below you will see the best pair:

$$\begin{aligned} \gamma &: 2^{-27} \\ \sigma &: 2^{9} \end{aligned}$$

(b) We will now calculate the values of the cross validation error for all of the pairs of (γ, σ) and plot them in a heatmap plot. The interpretation of this plot is that the lighter the value of a grid the higher the error. Each gird represents the value of the cross validation error for each (γ, σ) pair, where γ represents the horizontal axis and σ the vertical axis.

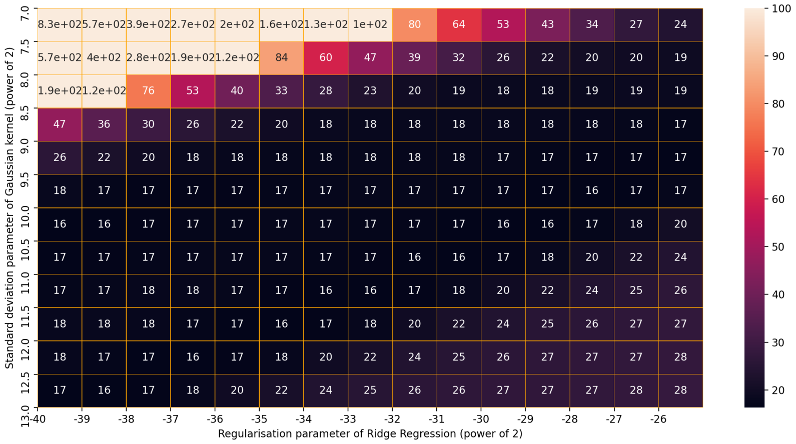

Fig 9: *Cross validation error*

We can see that for values of $\gamma < 2^{-32}$ and $\sigma < 2^{8.5}$ the error start increasing.

(c) We now calculate the MSE on the training and test sets for the best γ and σ pair.

$$\begin{aligned} training: MSE &= 8.350504249448724 \\ testing: MSE &= 10.933745518684931 \end{aligned}$$

(d) We repeat step a,b,c and record the train/test error and standard deviations of the train/test errors show in table 2.

Table 2: MSE of train/test in different regressions

	MSE train	**MSE test**
Naive Regression	83.8282+-5.5942	85.435+-11.9205
CRIM	71.2003+-4.976	71.8127+-8.412
ZZN	72.808+-5.2065	78.3452+-9.0971
INDUS	64.8486+-4.5231	67.4516+-9.8674
CHAS	79.8741+-5.358	81.3827+-7.3982
NOX	68.2813+-5.4806	68.1039+-9.0205
RM	43.4394+-3.1459	43.5684+-6.1201
AGE	71.8738+-4.8933	75.5024+-11.2398
DIS	78.7194+-4.8933	75.9108+-11.6784
RAD	73.5864+-5.0625	73.2694+-9.5224
TAX	66.0772+-4.0677	64.5588+-9.1009
PTRATIO	63.768+-4.6077	62.0114+-6.3922
LSTAT	37.7003+-2.0734	41.8196+-4.7855
Kernel Ridge Regression	7,6951+-0.8128	10.0794+-2.3829

3 Part II

In this section we will implement the k-NN algorithm and analyse the performance.

A voted-center hypothesis is a function $h_{S,v} : [0,1]^2 \to \{0,1,\}$ where a set of labeled centers $S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_{|s|}, y_{|S|})\}$. Each $\mathbf{x}_i \in [0,1]^2$ and $y_i \in [0,1]$.

3.1 Question 6

A visualisation of an $h_{S,v}$ where S=100 and v=3 is shown in Fig 10:

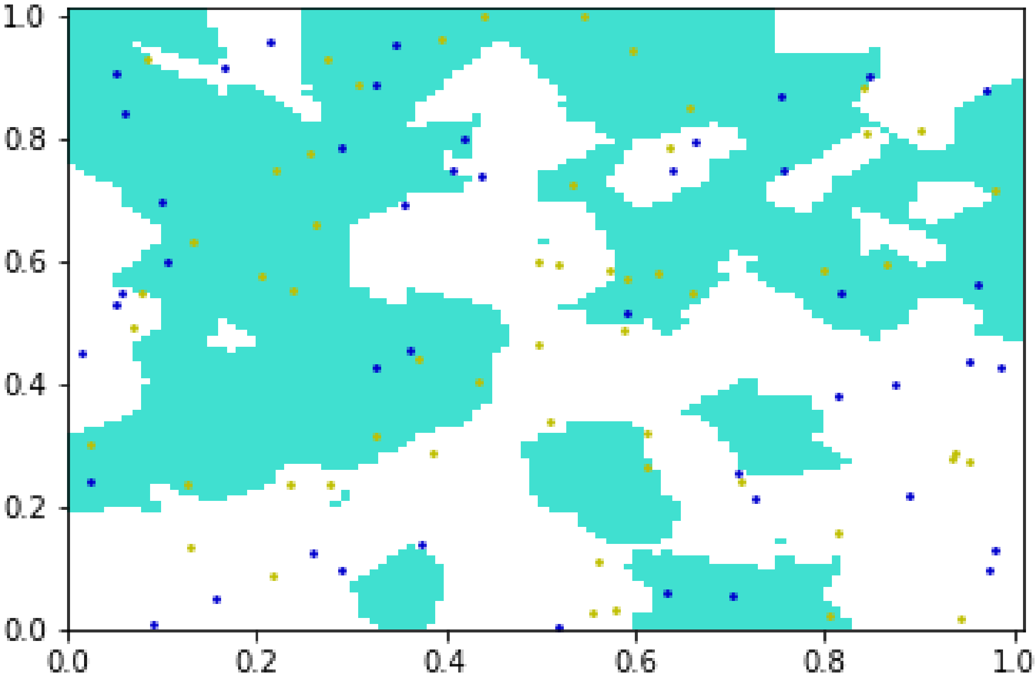

Fig 10: *Visualisation of an $h_{S,v}$*

3.2 Question 7

Given an $h_{S,v}$ the underlying probability distribution $p_h(\mathbf{x}, y)$ is determined my sampliing an $\mathbf{x}$ uniformly at random from $[0,1]^2$ and then its corresponding y value is then generated by flipping biased coin P(heads)=0.8. If heads comes up then y=$h_{S,v}(\mathbf{x})$ otherwise y is sampled uniformly at random from 0,1. A visualisation of the error using Protocol A is shown in Fig 12. The vertical axis corresponds to the error which is the percentage of incorrect predictions and the horizontal axis is the value of k - the number of neighbours.

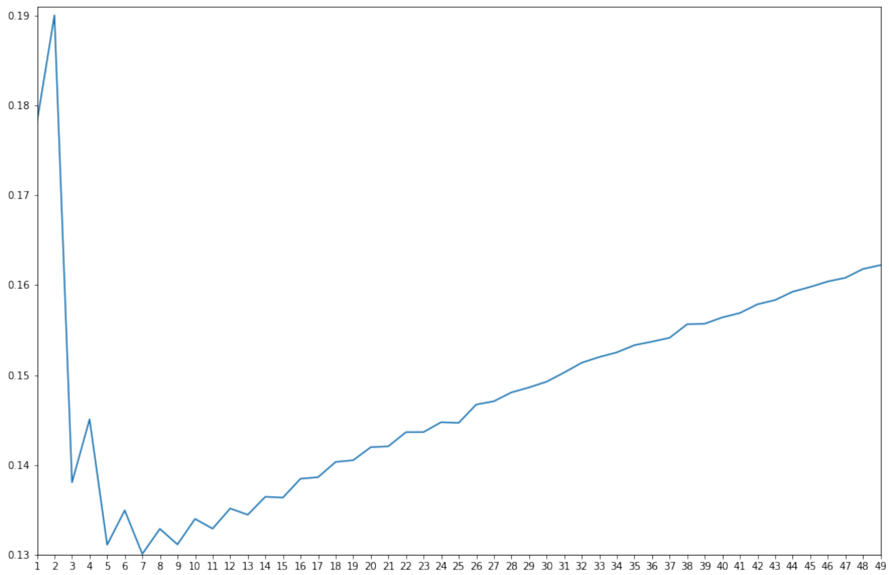

Fig 11: *Visualisation using Protocol A*

As Fig 11 shown above, as the value of k grows, the MSE experiences a sharp drop which is then followed by a slow rise after k=7. Let m denote the number of examples in the data. When ratio $\frac{m}{k}$ is too large, the model is prone to being overly complex, which may lead to overfitting, conversely, when $\frac{m}{k}$ is too small, the model could be underfitting. It follows that when doing k-NN classification, there should be a reasonable ratio between m and k.

3.3 Question 8

We will now calculate the optimal k for each m, where m is the number of training points and takes values in $\{100, 500, 1000, ..., 4000\}$. We plot the results in Fig 12:

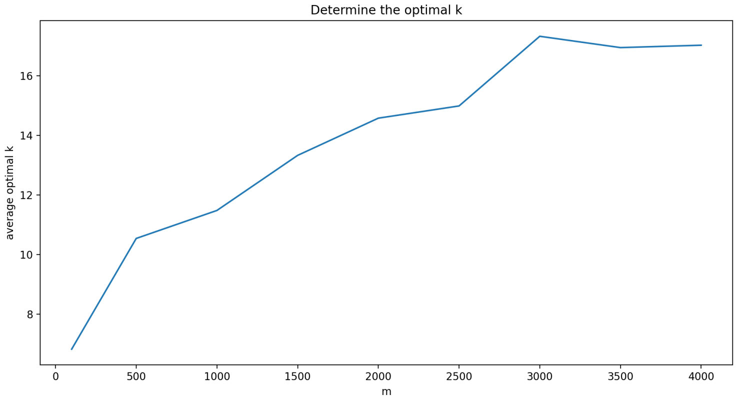

Fig 12: *Visualisation using Protocol B*

We can see that as the number of examples m increase, the optimal k increases as well, indicating that there is a ratio $\frac{m}{k}$ which is relatively consistent throughout the range of m. In other words, the ratio $\frac{m}{k}$ when $m = 500$ is close in value when $m = 2000$. Additionally, when we generate more training data points inside a unit square and generate 1000 testing data points, the distance between training data and testing data might be closer. As a result, larger k is needed to ensure accuracy.

4 Part III

4.1 Question 9

(a) We first show that the function $K_c(\mathbf{x}, \mathbf{z}) := c + \sum_{i=1}^{n} x_i z_i$ is symmetric:

$$\begin{aligned} K_c(x,z) &= c + \sum_{i=1}^{n} x_i z_i \\ &= c + (x_1 z_1 + x_2 z_2 + ... + x_n z_n) \\ &= c + (z_1 x_1 + z_2 x_2 + ... + x_n z_n) \\ &= c + \sum_{i=1}^{n} z_i x_i \\ &= K_c(z,x) \end{aligned}$$

To show for which values of $c \in R$ the function $K_c(\mathbf{x}, \mathbf{z}) := c + \sum_{i=1}^{n} x_i z_i$ is a positive semidefinite kernel, we need to show for which values of c the following inequality:

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

is true for all $a_i \in R, i \in \{1, ..., m\}$ and for all $\mathbf{x} \in R^n$.

We have that

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

$$\sum_{i,j} a_i a_j \left(c + \sum_{k=1}^{n} x_{ik} x_{jk} \right) \geq 0$$

$$c \sum_{i,j} a_i a_j + \sum_{i,j} a_i a_j \sum_{k} x_{ik} x_{jk} \geq 0$$

$$c \left(\sum_{i} a_i \right)^2 + \sum_{k} \left(\sum_{i} a_i x_{ik} \right) \left(\sum_{j} a_j x_{jk} \right) \geq 0$$

$$c \left(\sum_{i} a_i \right)^2 + \langle \sum_{i} a_i \mathbf{x}_i, \sum_{j} a_j \mathbf{x}_j \rangle \geq 0$$

$$c\left(\sum_i a_i\right)^2 + \| \sum_i a_i \mathbf{x}_i \|^2 \geq 0$$

We have that both $\left(\sum_i a_i\right)^2$ and $\| \sum_i a_i \mathbf{x}_i \|^2$ are non-negative for all $\mathbf{x}_i$ and a_i.

We consider four cases:

Case 1:

If $\left(\sum_i a_i\right)^2 = 0$ and $\| \sum_i a_i \mathbf{x}_i \|^2 = 0$ then can take any value, i.e., $c \in R$

Case 2:

If $\left(\sum_i a_i\right)^2 > 0$ and $\| \sum_i a_i \mathbf{x}_i \|^2 = 0$ then has to be greate than or equal to 0 i.e., $c \geq 0$

Case 3:

If $\left(\sum_i a_i\right)^2 = 0$ and $\| \sum_i a_i \mathbf{x}_i \|^2 > 0$ then $c \in R$

Case 4:

If $\left(\sum_i a_i\right)^2 > 0$ and $\| \sum_i a_i \mathbf{x}_i \|^2 > 0$ then $c \geq 0$

Hence,

$$c \in \{(-\infty, \infty) \cap (0, \infty)\}$$

Therefore for $c \geq 0$ the function $K_c(\mathbf{x}, \mathbf{z})$ is a PSD kernel.

b) For Ridge regression, we have that the dual of the optimization formulation after kernelization is

$$\alpha^* = \text{argmin}_{\alpha \in R^l} \frac{1}{l} \sum_{i=1}^{l} \sum_{j=1}^{l} K_{i,j} - y_i)^2 + \gamma \alpha^T K \alpha$$

For least squares linear regression, we set $\gamma = 0$, and then we obtain the following optimization formulation:

$$\alpha^* = \text{argmin}_{\alpha \in R^l} \frac{1}{l} \sum_{i=1}^{l} \left(\sum_{j=1}^{l} K_{i,j} - y_i \right)^2$$

Plugging in the kernel function $K_c(\mathbf{x}, \mathbf{z}) := c + \sum_{i=1}^{n} x_i z_i = c + K(x, z)$, where $K(x, z) = x^t z$ we derive the following expression:

$$\alpha^* = \text{argmin}_{\alpha \in R^l} \frac{1}{l} \sum_{i=1}^{l} \left(\sum_{j=1}^{l} \alpha_j (K_{c_{i,j}}) - y_i \right)^2$$

$$= \text{argmin} \frac{1}{l} \sum_{i=1}^{l} \left(\sum_{j=1}^{l} \alpha_j (c + K_{i,j}) - y_i \right)^2$$

$$= \text{argmin} \frac{1}{l} \sum_{i=1}^{l} \left(c \sum_{j=1}^{l} \alpha_j + \sum_{j=1}^{l} \alpha_j K_{i,j} - y_i \right)^2$$

$$= \text{argmin}\left(\frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j K_{i,j} - y_i\right)^2 + c^2\left(\sum_{j=1}^{l}\alpha_j\right)^2 + 2c\left(\sum_{j=1}^{l}\alpha_j\right)\frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j K_{i,j} - y_i\right)\right)$$

In the above expression if we set
$f(\alpha) = 2\left(\sum_{j=1}^{l}\alpha_j\right)\frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_j K_{i,j} - y_i\right)$
and $g(\alpha) = \left(\sum_{j=1}^{l}\alpha_j\right)^2$
then the expression becomes

$$\alpha^* = \text{argmin}_{\alpha\in R^l}\frac{1}{l}\sum_{i=1}^{l}\left(\sum_{j=1}^{l}\alpha_i K_{i,j} - y_i\right)^2 + c^2 g(\alpha) + cf(\alpha)$$

The c terms serves as a regularization term for α^*. Setting $c = 0$ we get the expression for α^* which is the solution to the least squares optimization formulation with $K(x,z) = x^T z$. When $c > 0$, the value of α^* will be restricted to a smaller hypothesis space. The higher the c, the smaller the hypothesis space of α^* becomes.

4.2 Question 10

It is clear that when $\beta \to \infty$:

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

It then follows that

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

Where $x_k = t$. Therefore,

$$\text{sign}(f(t)) = \text{sign}(\alpha_k K_\beta(x_k, t))$$

Now, since $K_\beta \geq 0$, it follows from above that $\text{sign}(f(t)) = \text{sign}(\alpha_k)$.
We should also note that when $\beta \to \infty$, the targets y_i are equal to α_i since

$$y_i = \sum_{j=1}^{m}\alpha_j K_\beta(x_j, x_i) = \alpha_i K_\beta(x_i, x_i) = \alpha$$

Hence, $\text{sign}(y_i) = \text{sign}(\alpha_i)$.
Using these two derivations we can see that it follows that

$$\text{sign}(f(t)) = \text{sign}(\alpha_k) = \text{sign}(y_k)$$

Where y_k is the class of the point x_k which is the closest training point to t, the point we are aiming to classify. Therefore when $\beta \to \infty$, the kernel regression with the Gaussian kernel $K_\beta = \exp(-\beta\|x-t\|^2)$ approximates a 1-NN classifier.

4.3 Question 11

Assume that our $n \times n$ grid can be represented by a matrix defined as

$$A = \begin{pmatrix} A_{1,1} & \cdots & A_{1,n} \\ \vdots & \ddots & \vdots \\ A_{n,1} & \cdots & A_{n,n} \end{pmatrix}$$

Such that $A \in F_2^{n\times n}$ where $F^{n\times n}{}_2$ is the finite field of 2 elements matrices and $A_{i,j} = 1$ if there is a mole sticking out of the (i,j)th hole of the grid and 0 otherwise. Our goal is to find a way to make all of the entries in this matrix equal to 0, i.e. turn A into the zero-matrix in an efficient way.

Now, define $W_{i,j}$, where $(i,j) \in \{1,...,n\} \times \{1,...,n\}$ as the matrix which represents the action of whacking a specific entry, (i,j) of the grid matrix A. The matrix $W_{i,j}$ is added to A and this operation represents the hole (i,j) being hit. When we hit a hole in A, we get a modified matrix $A' = A + W_{i,j}$. Define $h_{i,j}$ as the coefficients which indicate which holes were hit. We calculate $h_{i,j}$ by setting up the following equation:

$$A + \sum_{i,j} h_{i,j} W_{i,j} = \mathbf{0}$$

Where $\mathbf{0}$ is the $n \times n$ zero-matrix. Using the properties of F_2 we have that $A = -A$, hence:

$$\sum_{i,j} h_{i,j} W_{i,j} = A$$

Which can be further transformed into

$$\begin{pmatrix} W_{1,1} \\ W_{1,2} \\ \vdots \\ W_{n,n} \end{pmatrix} \cdot \begin{pmatrix} h_{1,1} \\ h_{1,2} \\ \vdots \\ h_{n,n} \end{pmatrix} = \begin{pmatrix} A_{1,1} \\ A_{1,2} \\ \vdots \\ A_{n,n} \end{pmatrix}$$

Where $W_{1,1}, ..., W_{n,n}$ represent the vectors of all possible places that is possible to hit on the A matrix. Solving for h requires solving a system of n^2 equations via Gaussian elimination which is done in $O(n^6)$ time.