

EECS 4404

Assignment 1

Name: Bochao Wang
Student ID: 215237902
Prism account: bochao
Date: Feb. 10

1. Bayesian Reasoning I

A terrible crime has been committed and blood is found on the crime scene, that must come from the person who committed the crime. Only 1% of the population (in the city which has 1000000 inhabitants) have this type of blood. A suspect is identified and tested positive for this blood type.

- (a) The prosecutor says: there was only 1% chance that he had this blood type if they were innocent, so there is now 99% chance they are guilty. What is wrong with this argument?

Solve: now we have 2 event need to consider,

event G : guilty; ($\neg G$: innocent)

event B : blood type tested positive (+); ($\neg B$: negative (-))

total people 1M and 1% + blood type and 1 person guilty

now we can have:

	$+ (B)$	$- (\neg B)$	total
guilty (G)	1	0	1
innocent ($\neg G$)	9,999	990,000	999,999
total	10,000	990,000	1,000,000

prosecutor means $P(B| \neg G) = 9,999 / 999,999 \approx 1\%$.

but it means if his blood type tested +, they are guilty chance is not 99%. i.e. $P(G|B) \neq 1 - P(B|\neg G) = 99\%$.

$$P(G|B) = 1 / 10,000 = 0.01\%$$

- (b) The defendant says: There are 10000 people in this city with this blood type, so the chance of being guilty is only 1/10000. What is wrong with this argument? Can you come up with a scenario, where it would be valid?

Solve: the problem is word "only", although 1/10000 chance is guilty in 10000

people with this blood type, it is still an important evidence which link between the suspect and the crime. Because the blood type evidence

prune the number of people who is the guilty probably from 1,000,000 to

10,000. When there is no any other evidence prove the suspect

is the guilty, the it would be valid.

- (c) Further investigations are being conducted, and more evidence collected. The search is narrowed down to 10 suspects. One of these 10 must have committed the crime. A first suspect of these is chosen (at random), the test conducted and it comes back positive. The judge says: "Given how this whole case developed, I have learned my lesson about using Bayes rule now. We can send this person to jail. We know:

$$p(B) = 1/100, \quad p(G) = 1/10,$$

where B is the event that the blood test comes back positive and G is the event that the person was guilty. We also know $p(B|G) = 1$, due to the evidence on the crime scene. Now we get

$$p(G|B) = \frac{p(B|G)p(G)}{p(B)} = \frac{1 \cdot \frac{1}{10}}{\frac{1}{100}} = 10$$

Now this seems convincing..." Is the judge correct?

Solve: No. it is not correct. In this scenario, we can not use $p(B) = 1/100$,

Because these 10 suspects are not select randomly, so the probability of the population have this type of blood is not 1% yet. this probability would increase in these 10 suspects. we can consider $p(B)$ in 10 suspect

$$P(B) = P(B|G)P(G) + P(B|\neg G)P(\neg G), \text{ where } P(B|G) = 1, \quad P(G) = \frac{1}{10}, \quad P(\neg G) = \frac{9}{10}$$

in the scenario. But we don't know $P(B|\neg G)$ exactly. It's

$0/9 \leq P(B|\neg G) \leq 9/9$ by the table. Thus, $P(G|B)$ would

$$P(G|B) = \frac{P(B|G)P(G)}{P(B)} \in [0.1, 1]$$

	+ (B)	- ($\neg B$)	total
guilty (G)	1	0	1
innocent ($\neg G$)	$0 \sim 9$	$9 \sim 0$	9
total	$1 \sim 10$	$9 \sim 0$	10

2. Bayesian Reasoning II

Consider a test which detects if a person has a disease. Let R denote the outcome of the test on a person, D denote whether the person actually has the disease and θ be the likelihood that the test gives the correct result. That is, the probability that it reports that someone has the disease ($R = 1$) when they actually do ($D = 1$), is θ , and the probability that it reports that someone doesn't have the disease when they don't is also θ . Formally:

$$p(R = 1|D = 1) = p(R = 0|D = 0) = \theta$$

Finally, an α -fraction of the population actually has this disease, that is, the prior probability of a person having this disease is $p(D) = \alpha$.

- (a) A patient goes to the doctor, has the test performed and it comes back positive. Derive a general formula for the posterior probability that the person actually has the disease, and simplify it in terms of θ and α . Which value do you get for $\alpha = 0.001$ and $\theta = .95$?

Solve: posterior probability that the person actually has the disease : $P(D=1|R=1)$

$$P(D=1|R=1) = \frac{P(R=1|D=1)P(D=1)}{P(R=1)}$$

$$P(D=1) = \alpha, P(R=1|D=1) = \theta$$

$$P(R=1) = P(R=1|D=1) \cdot P(D=1) + P(R=1|D=0) \cdot P(D=0)$$

$$\begin{aligned} P(D=1) &= \alpha \\ P(R=1|D=1) &= \theta \\ P(R=0|D=1) &= 1 - \theta \\ P(D=0) &= 1 - \alpha \\ P(R=0|D=0) &= \theta \\ P(R=1|D=0) &= 1 - \theta \end{aligned}$$

$$P(D=0) = 1 - P(D=1) = 1 - \alpha$$

$$P(R=1|D=0) = 1 - \theta$$

$$\text{Thus, } P(D=1|R=1) = \frac{\theta \cdot \alpha}{\theta \cdot \alpha + (1-\theta)(1-\alpha)} = 1.87\%$$

- (b) After the results of the first test come back positive, the doctor runs it a second time. Again, it comes back positive. Derive the posterior probability that the person actually has the disease after this second round of testing assuming the two test results are independent and simplify in terms of θ and α . Again, in addition to the general expression, report the values you get for $\alpha = 0.001$ and $\theta = .95$.

$$P(D=1 | (R_1=1 \cap R_2=1)) = \frac{P((R_1=1 \cap R_2=1)|D=1) \times P(D=1)}{P(R_1=1 \cap R_2=1)}$$

$$= \frac{P((R_1=1 \cap R_2=1)|D=1) \times P(D=1)}{P((R_1=1 \cap R_2=1)|D=1) \times P(D=1) + P((R_1=1 \cap R_2=1)|D=0) \times P(D=0)}$$

where $P(D=1) = \alpha$, $P(D=0) = (1-\alpha)$

$$P((R_1=1 \cap R_2=1) | D=1) = \theta \cdot \theta$$

$$P((R_1=1 \cap R_2=1) | D=0) = (1-\theta) \cdot (1-\theta)$$

$$\begin{aligned} \text{Thus, } P(D=1 | (R_1=1 \cap R_2=1)) &= \frac{\theta \cdot \theta \cdot \alpha}{\theta \cdot \theta \cdot \alpha + (1-\theta)(1-\theta)(1-\alpha)} \\ &= \frac{\theta^2 \cdot \alpha}{\theta^2 + 2\alpha - 2\theta - \alpha + 1} \end{aligned}$$

$$\approx 0.265 = 26.5\%$$

- (c) Analyze under which conditions the posterior probability of having the disease after two positive tests is larger than after only one positive test. How does it depend on α and θ ?

Solve: we know posterior probability \propto prior prob. \times likelihood.

we want to consider the situation that,

$$P(D=1 | (R_1=1 \cap R_2=1)) > P(D=1 | R=1)$$

$$\frac{\theta^2 \cdot \alpha}{\theta^2 + 2\alpha - 2\theta - \alpha + 1} > \frac{\theta \cdot \alpha}{\theta \cdot \alpha + (1-\theta)(1-\alpha)}, \text{ where } \theta > 0, \alpha > 0.$$

$$\frac{\theta}{\theta^2 + 2\alpha - 2\theta - \alpha + 1} > \frac{1}{2\alpha - \theta - \alpha + 1}$$

we just consider $0 \leq \theta \leq 1$, $0 \leq \alpha \leq 1$

$$\text{and } 0 \leq \frac{\theta^2 \cdot \alpha}{\theta^2 + 2\alpha - 2\theta - \alpha + 1} \leq 1, 0 \leq \frac{\theta \cdot \alpha}{\theta \cdot \alpha + (1-\theta)(1-\alpha)} \leq 1$$

we can compute,

$$0 < \alpha < \frac{1}{2}, \frac{1}{4}(3 - \sqrt{\frac{-7\alpha+1}{2\alpha-1}}) < \theta \leq 1 - \sqrt{\frac{\alpha}{2\alpha-1}}$$

analyze this inequality, when α more close to 0

the lower bound of θ would be higher, and close to 0.5

the upper bound of θ would also be higher, and close to 1.

it means when the probability of a disease $P(D=1)$ is lower, $P(R=1 | D=1)$ need higher, i.e. the test need more precise, to guarantee the posterior probability of having the disease after two positive tests is larger than after only one positive test.

But if α more than 0.5, we don't have satisfied θ to guarantee it.

When we just consider $\theta^2 + 2\alpha\theta - \theta - \alpha + 1 > 0$, $2\alpha\theta - \theta - \alpha + 1 > 0$
we can get $0 < \alpha < 1$ and $\frac{1}{2} < \theta < 1$, which more accuracy and satisfy ansise above.

- (d) Now we would like to use the above insights for employment of intelligent systems: Say security at an airport would like to use a machine learning based system to identify travelers smuggling illegal substances based on expressions of their face while going through security. Say the system was trained to 95% accuracy and we can expect 0.1% of travelers to be smuggling illegal substances. Would installing multiple cameras have the same effect as multiple blood tests? Explain your reasoning!

Solve: if we consider each camera identify travelers smuggling illegal substances is independent, this scenario is similar with multiple blood test. the $\alpha = 0.1\%$ and $\theta = 95\%$.

Based on part(c), we know $0 < \alpha < \frac{1}{2}$ and very close to 0. we need accuracy (i.e. θ), $0 < \theta < \frac{1}{2}$ so α and θ are satisfied

Thus, two camera would have same effect with part(c) scenario.

when cameras is more, we can use similar method to compute, so more cameras would increase accuracy.

3. Linear Algebra

We have seen in class that the solution to regularized least squares regression is given as a solution to the linear system

$$(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id}) \mathbf{w} = \mathbf{X}^T \mathbf{t}$$

where \mathbf{X} is the design matrix and \mathbf{Id} is the identity matrix. In this question you will prove that if $\lambda > 0$, then $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$ is invertible.

- (a) Show that every eigenvector of the matrix $\mathbf{X}^T \mathbf{X}$ is also an eigenvector of the matrix $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$.

Proof: Assume $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id}) \mathbf{v} = \lambda' \mathbf{v}$, where \mathbf{v} is eigenvector of $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$, and

λ' is eigenvalue of $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$.

$$\therefore (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id}) \mathbf{v} = \lambda' \mathbf{v}$$

$$\Leftrightarrow \mathbf{X}^T \mathbf{X} \mathbf{v} + \lambda \mathbf{v} = \lambda' \mathbf{v}, \text{ where } \mathbf{Id} \mathbf{v} = \mathbf{v}.$$

$$\Leftrightarrow \mathbf{X}^T \mathbf{X} \mathbf{v} = (\lambda' - \lambda) \mathbf{v}$$

$\therefore \mathbf{v}$ is eigenvector of $\mathbf{X}^T \mathbf{X}$ and vice versa.

Thus $\forall \mathbf{v} (\mathbf{X}^T \mathbf{X} \mathbf{v} = \lambda' \mathbf{v} \rightarrow (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id}) \mathbf{v} = \lambda'' \mathbf{v})$. The original statement is true.

- (b) Show that all eigenvalues of $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$ are strictly positive.

Proof: we have $\mathbf{X}^T \mathbf{X}$, where $\mathbf{X} \in \mathbb{R}^{m \times n}$ like part (a).

By the properties of symmetric matrix, $\mathbf{X}^T \mathbf{X}$ is a symmetric matrix and $\mathbf{X}^T \mathbf{X}$ is positive semi-definite.

$$\text{we have part (a)} \quad \mathbf{X}^T \mathbf{X} \mathbf{v} = (\lambda' - \lambda) \mathbf{v}$$

$\therefore \mathbf{X}^T \mathbf{X}$ is positive semi-definite.

\therefore the eigenvalues $(\lambda' - \lambda) > 0$

$$\Rightarrow \lambda' > \lambda$$

And we know in $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$, $\lambda > 0$

$$\therefore \lambda' > 0$$

by part (a) $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id}) \mathbf{v} = \lambda' \mathbf{v} \Rightarrow \lambda' > 0$ (proved)

(c) Use the above results to conclude that $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$ is invertible.

Proof. By part (b) we know all eigenvalues of $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$ are strictly positive.

Thus, $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$ is positive definite matrix.

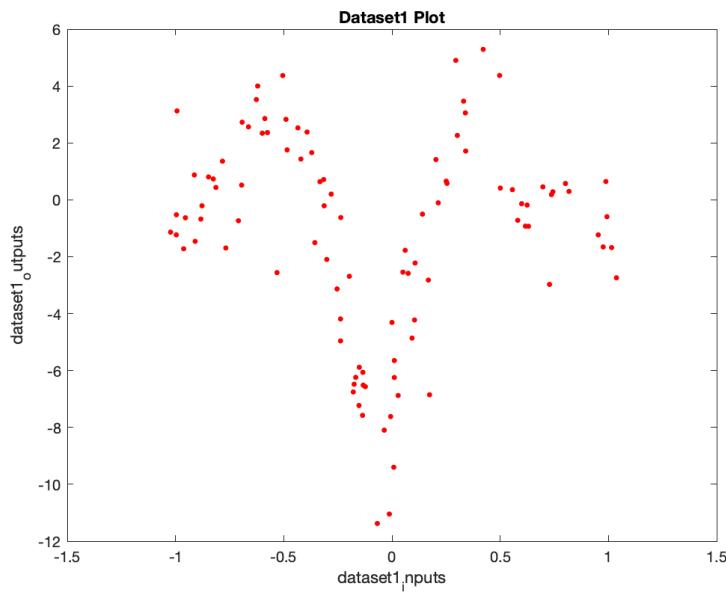
The number 0 is not an eigenvalue of $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$.

$(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id}) \mathbf{x} = 0$ has no non-trivial solution.

Thus, $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{Id})$ is invertible.

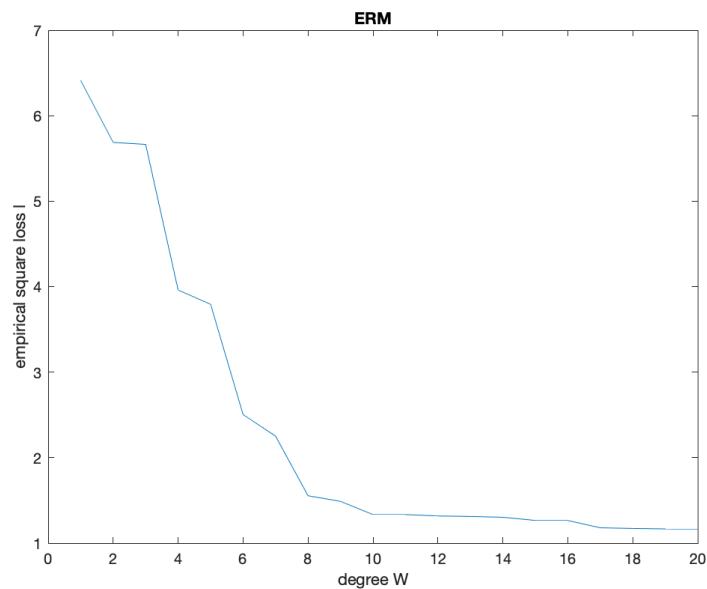
4. Liner Regression

Step 1 - load the data



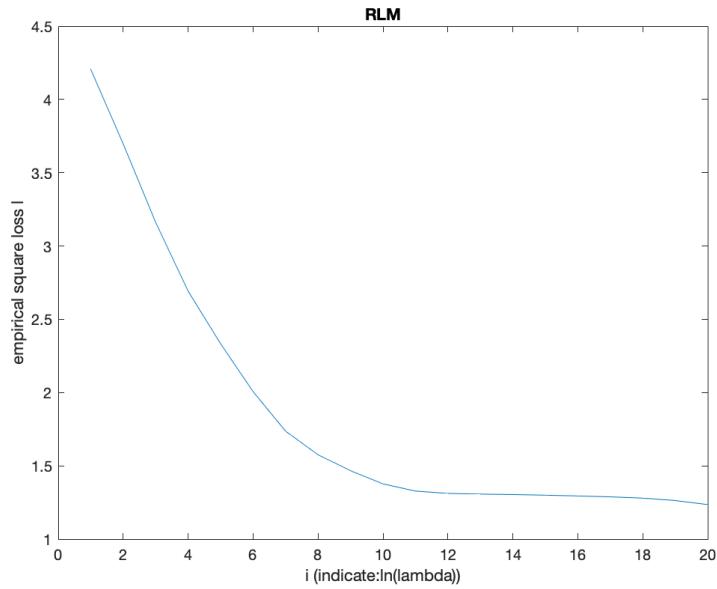
Step 2 - ERM

Plot the empirical square loss on the data with degrees $W = 1, \dots, 20$.



Based on the plot, when degree $W > 10$ the loss would be much less than other degree. $W = 20$ has least loss, it is suitable when we do not consider overfit so far.

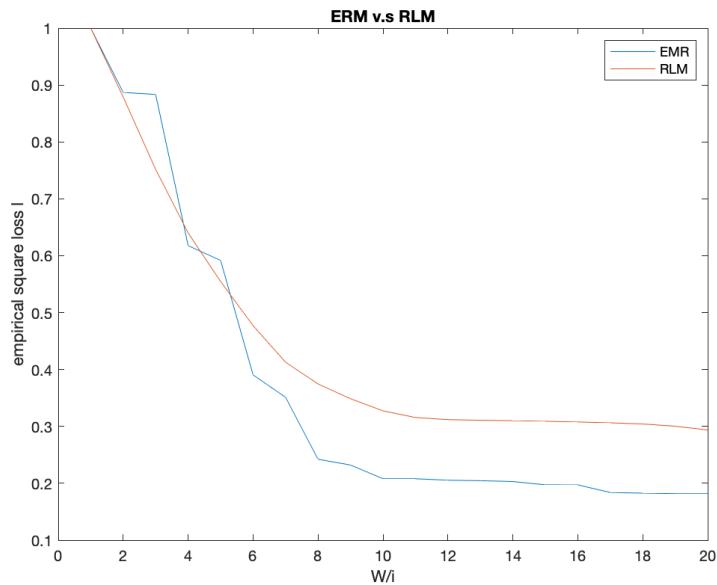
Step 3 - RLM



based on the plot, when degree $W = 20$ and the $\ln(\lambda) = -1, -2, \dots, -20$, after $\ln(\lambda) < -10$, loss would be much less than other dergree. $\ln(\lambda) = -20$ has least loss, it is suitable when we do not consider overfit so far.

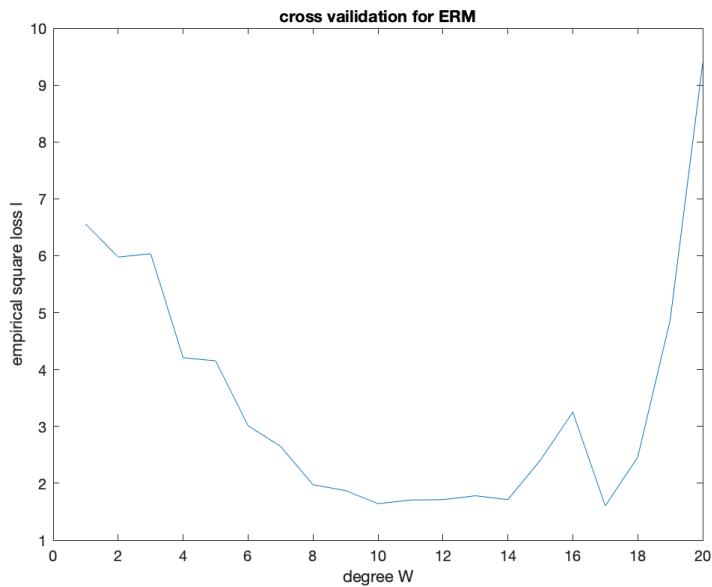
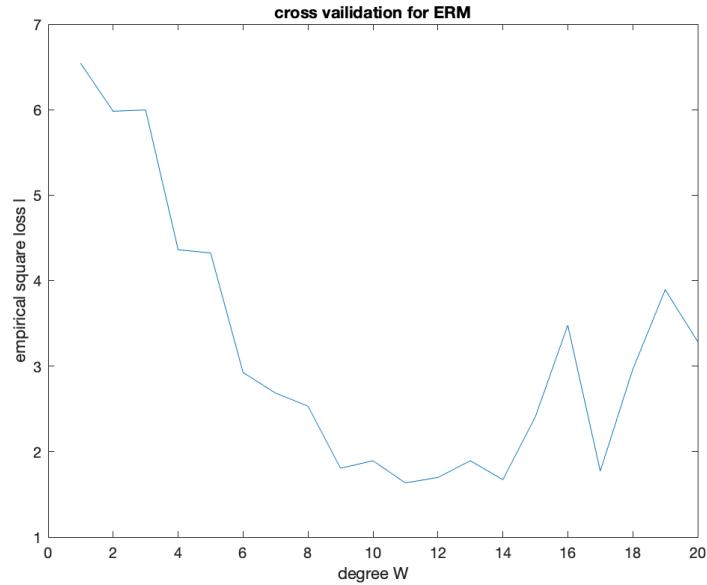
compare ERM and RLM

Note: we nomalize the loss into $(0, 1)$, for comparing ERM and RLM



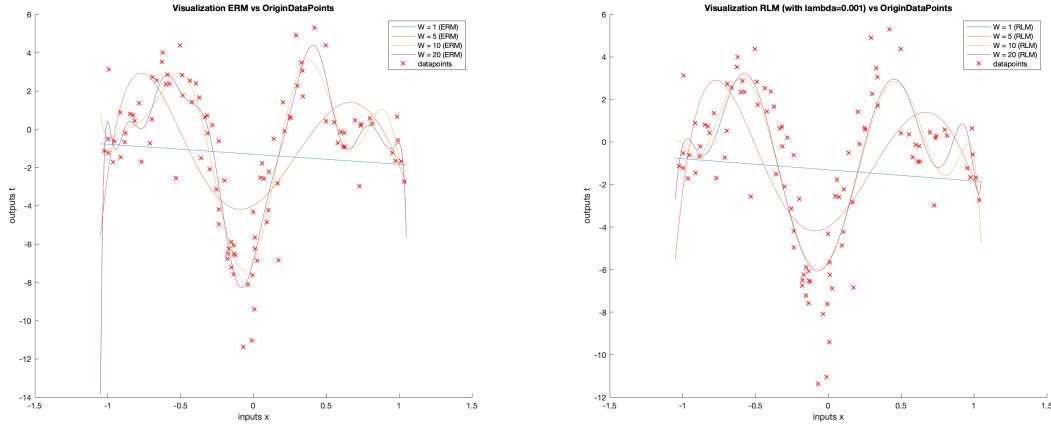
When degree $W = 20$, $\ln(\lambda) = -1, -2, \dots, -20$ to regularize. We can see after regularization, the empirical square loss would be higher than before regularization. Although, the loss is a little bit higher, but it would balance the loss and regularizer, which would avoid the overfitting problem.

Step 4 - cross validation



We can see when we use cross validation for ERM, $W = 20$ is not lowest loss yet, because of overfitting degree larger than 10. Although each time the curves have some difference, the tendency would be same. Thus, $W = 10 \sim 12$ is suitable after consider the overfitting.

Step 5 - visualization



we can compare two figures, ERM fits data very well as larger degree, but there is overfitting issue. After regularization, in some degree, it would avoid overfitting when the degree W is larger. Thus, adding regularizer ($\lambda = 0.001$), as $W = 20$ is most suitable.

Step 6 - bonus

Because there is no dataset2 , we just give intuition: we can repeate step 1 - step 5, and consider choose more suitable degree W , nerrow down from $\{1, 2, \dots, 20\}$ to smaller range such as $\{5, 6, \dots, 9\}$, use `train_6.m` with cross validation to find minimum loss and avoid overfitting as possible. Call this function repeatedly to choose minimum loss, and plot the fitting curves and original data to compare. From this step, we can analyze figures to choose ERM or RLM so that avoid overfitting. We found RLM is more suitable, so we change a routine in `train_6.m`, just training our model with RLM, and find minimum loss and output the most suitable degree W .

Source Code

Step 1-5

`main.m`

```

1 % STEP 1 load data
2 [x, t] = load_data(["dataset1_inputs.txt","dataset1_outputs.txt"]);
3 fprintf("step1: Finish load data and plot, please press enter to
4 continue!\n")
5 pause
6 clc;close all;
7
8 % STEP 2 minimizer of the empirical risk (ERM). compute the empirical
9 % square
10 % loss (Loss) on the data and plot it as a function of degree W
11 loss_erm = zeros(20,1);
12 % compute weight with ERM and loss, which is with each degree W
13 for d = 1:20

```

```

12     w = erm_w(x, t, d);
13     loss_erm(d) = q_loss(w, x, t);
14 end
15 % Normalization for loss
16 loss_erm = loss_erm/max(loss_erm);
17
18 % plot the loss_erm graph with degree W
19 plot(loss_erm);
20 title('ERM');
21 ylabel('empirical square loss l');
22 xlabel('degree W');
23 fprintf("step2: Finish compute and plot empirical square loss on the data,
24 please press enter to continue!\n")
24 pause;
25 clc;close all;
26
27 % STEP 3 minimizer of the regularized risk (RLM). compute regularized least
28 % squares regression on the data and plot the empirical loss as a function
29 % of i. compare ERM and RLM
30 loss_rlm = zeros(20,1);
31 % compute weight with RLM and loss, which is with each degree W
32 for i = 1:20
33     w = rlm_w(x, t, 20, -i);
34     loss_rlm(i) = q_loss(w, x, t);
35 end
36 % Normalization for loss
37 loss_rlm = loss_rlm/max(loss_rlm);
38 % plot the loss_rlm graph with degree W
39 plot(loss_rlm);
40 title('RLM');
41 ylabel('empirical square loss l');
42 xlabel('i (indicate:ln(lambda))');
43 fprintf("step3.1: Finish compute and plot empirical square loss on the
44 data, please press enter to continue!\n")
44 pause;
45 plot(loss_erm);
46 hold on
47 plot(loss_rlm);
48 legend('ERM', 'RLM');
49 title('ERM v.s RLM');
50 ylabel('empirical square loss l');
51 xlabel('W/i');
52 fprintf("step3.2: Finish compare ERM and RLM, please press enter to
53 continue!\n")
53 pause;
54 clc; close all;

```

```

55
56 % STEP 4 cross vailidation. Implement 10-fold cross validation for ERM.
57 % concat pair of inputs and outputs
58 concat = horzcat(x,t);
59 % % rank data randomly
60 % init some para.
61 loss_cross_val = zeros(20,1);
62 fold = 10;
63 % compute loss with cross vailidation, which is with each degree W
64 for d = 1:20
65     % rank data randomly
66     rowrank = randperm(size(concat, 1));
67     rank_data = concat(rowrank, :);
68     loss_cross_val(d) = cross_vailidation_erm(rank_data,d,fold);
69 end
70 % Normalization for loss
71 loss_cross_val = loss_cross_val/max(loss_cross_val);
72 % plot the loss_cross_val graph with degree W
73 plot(loss_cross_val);
74 title('cross vailidation for ERM');
75 ylabel('empirical square loss l');
76 xlabel('degree W');
77 fprintf("step4: Finish cross vailidation and plot empirical square loss on
the data, please press enter to continue!\n")
78 pause;
79 clc;close all;
80
81 % STEP 5 visualization.
82 % init some setting
83 degrees = [1 5 10 20];
84 interval = -1.05:0.01:1.05;
85 label_erm = string(zeros(length(degrees)+1,1));
86 label_rlm = string(zeros(length(degrees)+1,1));
87 % load labels
88 n = 1;
89 for i = degrees
90     label_erm(n) = ("W = " +num2str(i)+" (ERM)");
91     n = n + 1;
92 end
93 label_erm(n) = "datapoints";
94 n = 1;
95 for i = degrees
96     label_rlm(n) = ("W = " +num2str(i)+" (RLM)");
97     n = n + 1;
98 end
99 label_rlm(n) = "datapoints";

```

```

100 % plot the data along with the ERM learned models
101 figure;
102 subplot(1,2,1)
103 hold on;
104 for d = degrees
105     w_erm_vis = erm_w(x, t, d);
106     plot(interval,func(w_erm_vis,interval));
107 end
108 plot(x,t,'rx');
109 title('Visualization ERM vs OriginDataPoints');
110 ylabel('outputs t');
111 xlabel('inputs x');
112 legend(label_erm);
113 subplot(1,2,2)
114 hold on
115 % plot the data along with the RLM learned models
116 for d = [1 5 10 20]
117     w_rlm_vis = rlm_w(x, t, d, log(0.001));
118     plot(interval,func(w_rlm_vis,interval));
119 end
120 % plot origin dataset
121 plot(x,t,'rx');
122 title('Visualization RLM (with lambda=0.001) vs OriginDataPoints');
123 ylabel('outputs t');
124 xlabel('inputs x');
125 legend(label_rlm);
126 fprintf("step5: Finish visualization with ERM and RLM, please press enter
127 to continue!\n")
128 pause;
129 clc;close all;

```

`functions()`

```

1 function [x, t] = load_data(a)
2 % Load Data
3 input = load(a(1));
4 output = load(a(2));
5 x = input(:, 1); t = output(:, 1);
6 % Plot Data
7 fprintf('Plotting Data ... \n');
8 plot(x, t, 'r.', 'MarkerSize', 10);
9 title('Dataset1 Plot');
10 ylabel('dataset1_outputs');
11 xlabel('dataset1_inputs');

```

```

1 function w = erm_w(x, t, d)
2 N = size(x, 1);
3 % design matrix X of the data,
4 % where N is size of the data and d is degree of poly
5 % |x1^0 x1^1 ... x1^d|
6 % |x2^0 x2^1 ... x2^d|
7 % |...| = X
8 % |...|
9 % |xN^0 xN^1 ... xN^d|
10 X = zeros(N,d);
11 for r = 1:N
12     for c = 1:d
13         X(r, c) = x(r)^c;
14     end
15 end
16 % first column would be constant
17 X = [ones(N,1), X];
18 % vector w that solves the unregularized least squares linear regression
problem
19 % ERM solution w = (X'*X)^-1 * X' * t from slide,
20 % where X is design matrix of the data
21 % w = (X' * X)^-1 * X' * t;
22 w = pinv(X' * X) * X' * t;

```

```

1 function w = rlm_w(x, t, d, ln_lambda)
2 N = size(x, 1);
3 % d = 2;
4 % ln_lambda = -2
5 lambda = exp(ln_lambda);
6 % design matrix X of the data,
7 % where N is size of the data and d is degree of poly
8 % |x1^0 x1^1 ... x1^d|
9 % |x2^0 x2^1 ... x2^d|
10 % |...| = X
11 % |...|
12 % |xN^0 xN^1 ... xN^d|
13 X = zeros(N,d);
14 for r = 1:N
15     for c = 1:d
16         X(r, c) = x(r)^c;
17     end
18 end
19 % first column would be constant
20 X = [ones(N,1), X];
21 Id = eye(size(X' * X));

```

```

22 % vector w that solves the regularized least squares linear regression
23 % problem
24 % RLM solution w = (X'*X+lambda*Id)^-1 * X' * t from slide,
25 % where X is design matrix of the data
26 % w = (X' * X + lambda * Id)^-1 * X' * t;
27 w = pinv(X' * X + lambda * Id) * X' * t;

```

```

1 function Loss = q_loss (w, x, t)
2 N = size(x,1);
3 Loss = 0;
4 for i = 1:N
5     Loss = Loss + 1/2 * (func(w,x(i)) - t(i))^2;
6 end
7 Loss = (1/N) * Loss;

```

```

1 % yw(xi) = (Xw)i = w0x0+w1x1+..+wdx
2 function y = func(w,x)
3 w_inverse = zeros(1,size(w,1));
4 for i = 1:size(w)
5     w_inverse(i) = w(size(w,1)-i+1);
6 end
7 y = polyval(w_inverse,x);

```

```

1 function avg_loss = cross_vailidation_erm(rank_data,degree,fold)
2 chunck = size(rank_data,1)/fold; % the number of times of testing
3 tot_loss = 0; % init total loss
4 for i = 1:fold
5     n=1;
6     testing = zeros(chunck, 2);
7     % load testing set
8     for j = 1+(i-1)*chunck : i*chunck
9         testing(n,:) = rank_data(j,:);
10        n=n+1;
11    end
12    % load remaining rank_data for training set
13    training = rank_data(~ismember(rank_data,testing,'rows'),:);
14    % training our model
15    w = erm_w(training(:,1), training(:,2), degree);
16    % compute the total loss
17    tot_loss = tot_loss + q_loss(w, testing(:,1), testing(:,2));
18 end
19 avg_loss = tot_loss/fold;

```

Step 6 - bonus

main.m

```
1 % STEP 6 bonus
2 x_b = load("dataset2_inputs.txt");
3 t_b = load("dataset2_outputs.txt");
4 interval = -1:0.01:1.25;
5 [opt_w, min_loss] = train_6(x_b,t_b);
6 for i = 1:5
7     [w,l] = train_6(x_b,t_b);
8     if l < min_loss
9         min_loss = l;
10        opt_w = w;
11    end
12 end
13 opt_w
14 plot(interval,func(opt_w,interval));
15 hold on
16 plot(x_b,t_b,'rx');
17 title('Visualization Model vs OriginDataPoints');
18 ylabel('outputs t');
19 xlabel('inputs x');
20 save('opt_w_s6.txt','opt_w');
21 fprintf("step6: Finish training model, please press enter to continue!\n")
22 pause;
23 clc;close all;
```

function()

```
1 function [opt_w,min_loss] = train_6 (x_b,t_b)
2 % x_b = load("dataset2_inputs.txt");
3 % t_b = load("dataset2_outputs.txt");
4 %concat pair of inputs and outputs
5 concat = horzcat(x_b,t_b);
6 % rank data randomly
7 rowrank = randperm(size(concat, 1));
8 rank_data = concat(rowrank, :);
9 % init some para.
10 fold = 10;
11 chunck = size(rank_data,1)/fold; % the number of times of testing
12 min_loss = inf;
13 % compute loss with cross vailidation, which is with each degree W
14 loss = zeros(1,fold);
15 for degree = 5:9
16     w = zeros(degree+1,fold);
17     for i = 1:fold
```

```

18 n=1;
19 testing = zeros(chunck, 2);
20 % load testing set
21 for j = 1+(i-1)*chunck : i*chunck
22     testing(n,:) = rank_data(j,:);
23     n=n+1;
24 end
25 % load remaining rank_data for training set
26 training = rank_data(~ismember(rank_data,testing, 'rows'),:);
27
28 % training our model
29 % w(:,i) = erm_w(training(:,1), training(:,2), degree);
30 % compute the total loss
31 % loss(:,i) = q_loss(w(:,i), testing(:,1), testing(:,2));
32
33 n = 1;
34 loss_rlm = zeros(1,20);
35 w_rlm = zeros(degree+1,20);
36 for ln_lambda = 1:20
37     % training our model
38     w_rlm(:,n) = rlm_w(training(:,1), training(:,2), degree, -
ln_lambda);
39     % compute the total loss
40     loss_rlm(:,n) = q_loss(w_rlm(:,n), testing(:,1), testing(:,2));
41     n=n+1;
42 end
43 if min(loss_rlm) < min_loss
44     min_loss = min(loss_rlm);
45     index = loss_rlm==min(loss_rlm);
46     opt_w = w_rlm(:,index);
47     %flag="RLM";
48 end
49 end
50
51 % if min(loss) < min_loss
52 %     min_loss = min(loss);
53 %     index = loss==min(loss);
54 %     opt_w = w(:,index);
55 %     %flag="EMR";
56 % end
57 end

```