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: nocent)

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 bettem 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 vaild.

(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 probility of the population have this type of blood is not 1% yet. this probility would increase in these 10 suspects, we can consider $P(B)$ in 10 suspect

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

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. Thuse, $P(G|B)$ would

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

	+(B)	−(¬B)	total
guilty (G)	1	0	1
innocent(¬G)	0 ~ 9	9 ~ 0	9
total	1 ~ 10	9 ~ 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 doesnt 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 $p(D) = \alpha$.

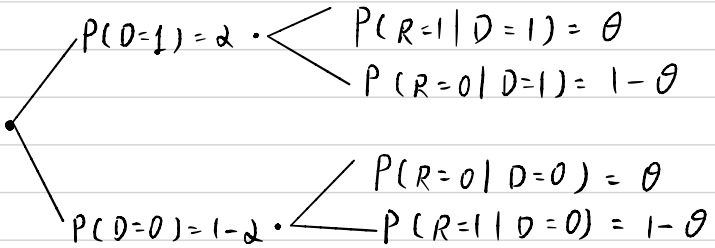
(a) A patient goes to the doctor, has the test performed and it comes back positive. Derive a general formula for the posterior probability that the person actually has the disease, and simplify it in terms of θ and α. Which value do you get for $\alpha = 0.001$ and $\theta = .95$?

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

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

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

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



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

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

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

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

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

Thus, $P(D=1 \mid (R_1=1 \cap R_2=1)) = \dfrac{\theta \cdot \theta \cdot \alpha}{\theta \cdot \theta \cdot \alpha + (1-\theta)(1-\theta)(1-\alpha)}$

$= \dfrac{\theta^2 \cdot \alpha}{\theta^2 + 2\theta\alpha - 2\theta - \alpha + 1}$.

$\approx 0.265 = 26.5\%$.

(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. $*$ likehood.

we want to consider the satuation that,

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

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

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

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

and $0 \le \dfrac{\theta^2 \cdot \alpha}{\theta^2 + 2\theta\alpha - 2\theta - \alpha + 1} \le 1$, $0 \le \dfrac{\theta \cdot \alpha}{\theta \cdot \alpha + (1-\theta)(1-\alpha)} \le 1$

we can compute,

$0 < \alpha < \frac{1}{2}$, $\frac{1}{4}\left(3 - \sqrt{\frac{-7\alpha - 1}{\alpha - 1}}\right) < \theta \le 1 - \sqrt{-\frac{\alpha}{\alpha - 1}}$.

analyze this inequality, when α more close to 0 the lower boundge of θ would be higher, and close to 0.5 the upper boundge of θ would also be higher, and close to 1.

it means when the probilty of a disease $p(D=1)$ is lower, $p(R=1 \mid D=1)$ 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 $\theta^2 + 2\theta\alpha - 2\theta - \alpha + 1 > 0$, $2\theta\cdot\alpha - \theta - \alpha + 1 > 0$ we can get $\underline{0 < \alpha < 1 \text{ and } \frac{1}{2} < \theta < 1}$, which more accuracy and satisify anasise 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 you reasoning!

Solve: if we consider each camera identify travelers smuggling illegal subances is independant, 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 satified

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 $(X^TX + \lambda Id)\,V = \lambda' V$, where v is eigenvector of $(X^TX + \lambda Id)$, and λ' is eigenvalue of $(X^TX + \lambda Id)$.

$\therefore (X^TX + \lambda Id)\,V = \lambda' V$

$\Leftrightarrow X^TX\,V + \lambda V = \lambda' V$, where $Id\,V = V$.

$\Leftrightarrow X^TX\,V = (\lambda' - \lambda)\,V$

$\therefore$ V is eigenvector of X^TX and vice versa.

Thus $\forall V\,(X^TX\,V = \lambda' V \rightarrow (X^TX + \lambda Id)\,V = \lambda'' V)$. 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 X^TX, where $X \in \mathbb{R}^{m\times n}$ like part (a).

By the properties of symmetric matrix, X^TX is a symmetric matrix and X^TX is positive semi-definite.

we have part (a) $X^TX\,V = (\lambda' - \lambda)\,V$

$\therefore X^TX$ is positive semi-definite.

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

$\Rightarrow \lambda' \geq \lambda$

And we know in $(X^TX + \lambda id)$, $\lambda > 0$

$\therefore \lambda' > 0$

by part (a) $(X^TX + \lambda Id)\,V = \lambda' V \Rightarrow \lambda' > 0$. (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 eigenvalue of $(X^TX + \lambda Id)$ are strictly positive.

Thus, $(X^TX + \lambda Id)$ is positive definite matrix.

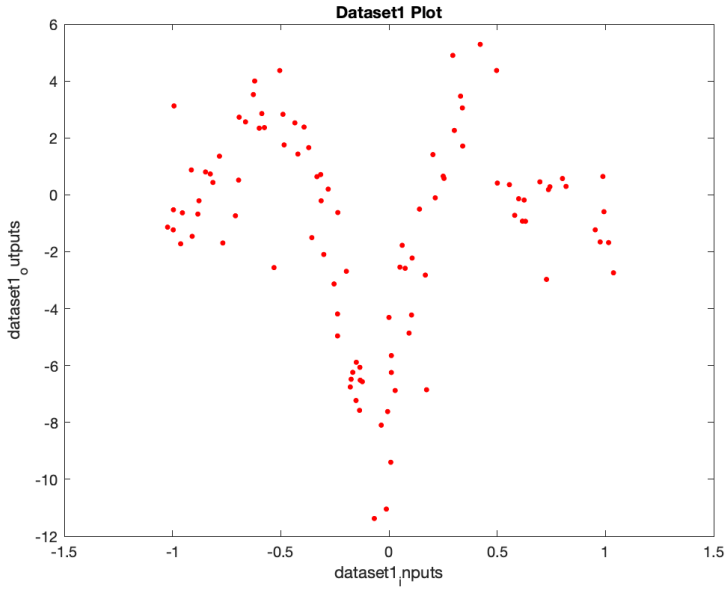
The number 0 is not an eigenvalue of $(X^TX + \lambda Id)$

$(X^TX + \lambda Id)x = 0$ has no non-trivial solution.

Thus, $(X^TX + \lambda Id)$ is invertible.

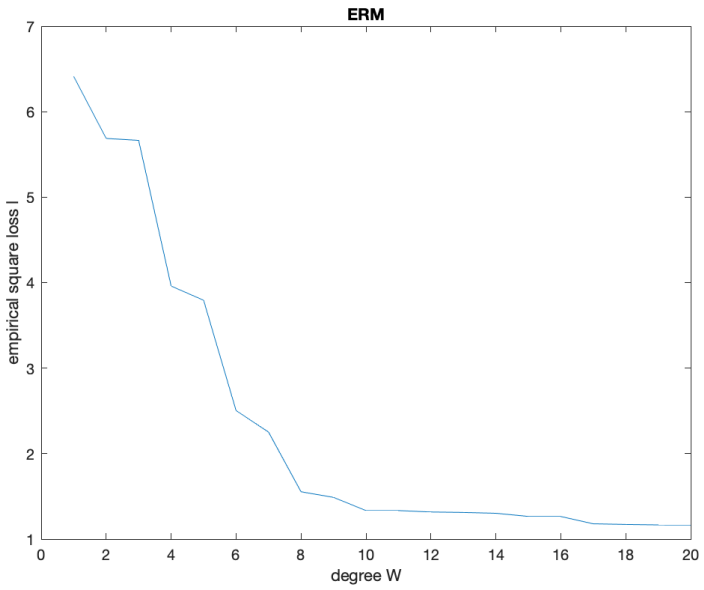
4. Liner Regression

Step 1 - load the data



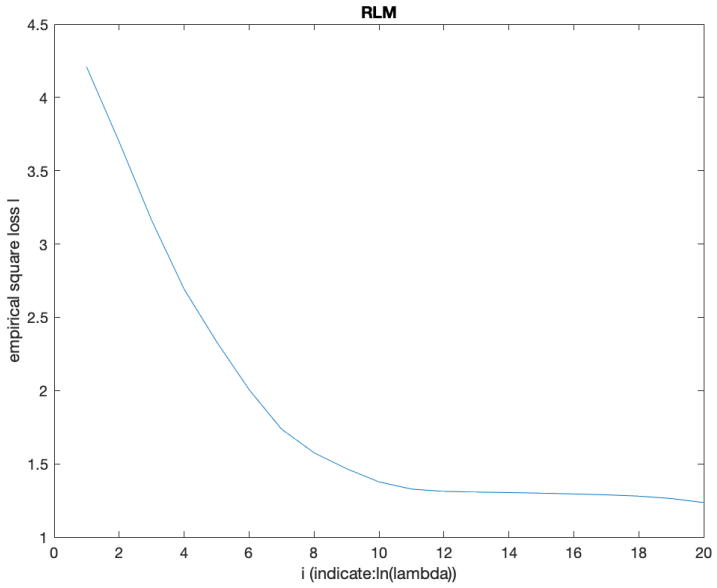
Step 2 - ERM

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



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

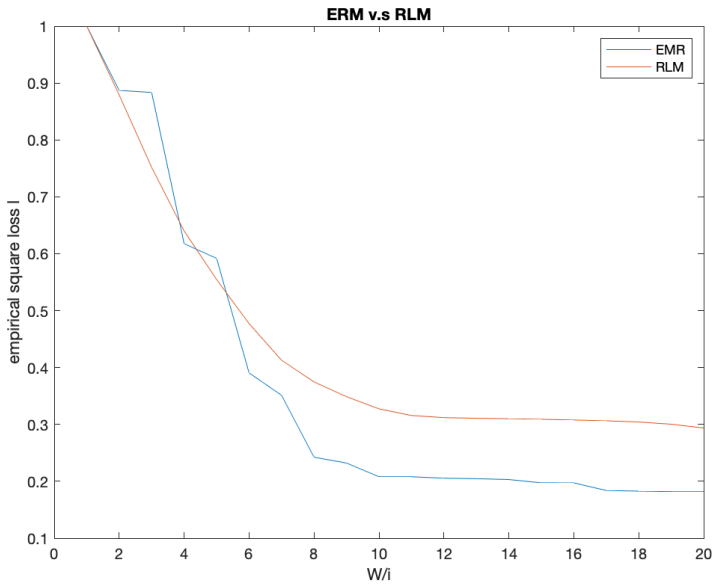
Step 3 - RLM



based on the plot, when degree $W = 20$ and the $ln(\lambda) = -1, -2, .., -20$, 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.

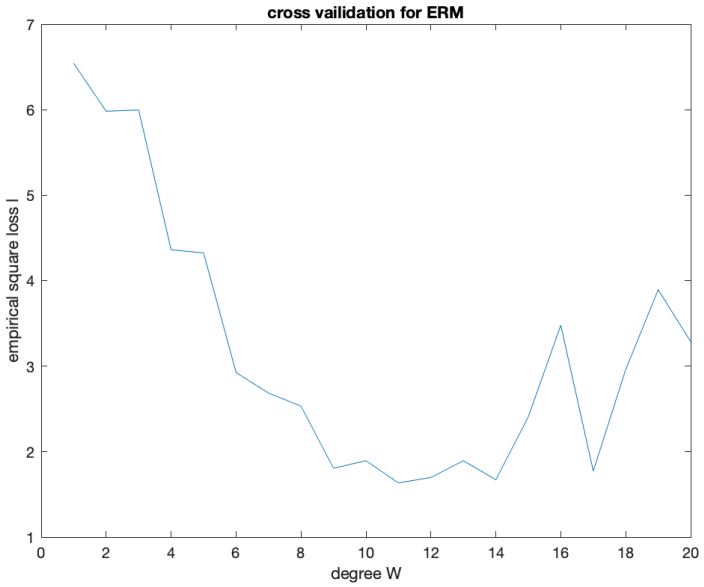
compare ERM and RLM

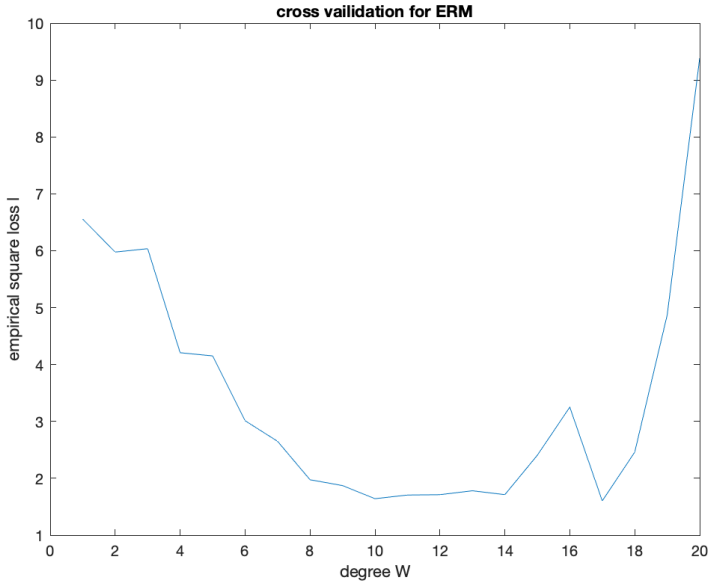
Note: we nomalize the loss into $(0, 1)$, for comparing ERM and RLM



When degree $W = 20$, $ln(\lambda) = -1, -2, .., -20$ 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.

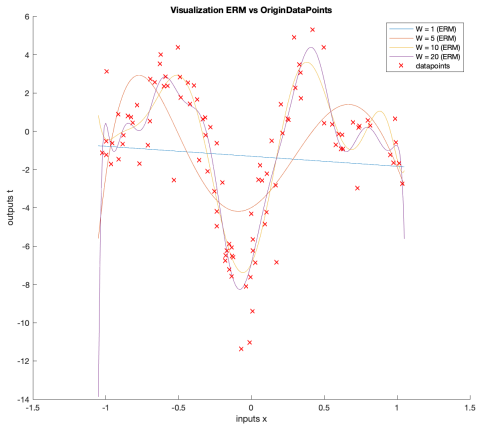
Step 4 - cross validation

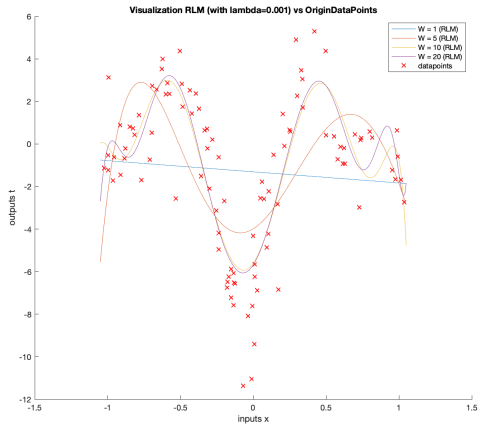




We can see when we use cross validation for ERM, $W = 20$ is not lowest loss yet, because of overfitting degree larger than 10. Although each time the curves have some difference, the tendency would be same. Thus, $W = 10 \sim 12$ is suitable after consider the overfitting.

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 $\{1, 2, .., 20\}$ to smaller range such as $\{5, 6, .., 9\}$, 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`

```
% STEP 1 load data
[x, t] = load_data(["dataset1_inputs.txt","dataset1_outputs.txt"]);
fprintf("step1: Finish load data and plot, please press enter to
continue!\n")
pause
clc;close all;

% STEP 2 minimizer of the empirical risk (ERM). compute the empirical
square
% loss (Loss) on the data and plot it as a function of degree W
loss_erm = zeros(20,1);
% compute weight with ERM and loss, which is with each degree W
for d = 1:20
```

```
    w = erm_w(x, t, d);
    loss_erm(d) = q_loss(w, x, t);
end
% Normalization for loss
loss_erm = loss_erm/max(loss_erm);

% plot the loss_erm graph with degree W
plot(loss_erm);
title('ERM');
ylabel('empirical square loss l');
xlabel('degree W');
fprintf("step2: Finish compute and plot empirical square loss on the data, please press enter to continue!\n")
pause;
clc;close all;

% STEP 3 minimizer of the regularized risk (RLM). compute regularized least
% squares regression on the data and plot the empirical loss as a function
% of i. compare ERM and RLM
loss_rlm = zeros(20,1);
% compute weight with RLM and loss, which is with each degree W
for i = 1:20
    w = rlm_w(x, t, 20, -i);
    loss_rlm(i) = q_loss(w, x, t);
end
% Normalization for loss
loss_rlm = loss_rlm/max(loss_rlm);
% plot the loss_rlm graph with degree W
plot(loss_rlm);
title('RLM');
ylabel('empirical square loss l');
xlabel('i (indicate:ln(lambda))');
fprintf("step3.1: Finish compute and plot empirical square loss on the data, please press enter to continue!\n")
pause;
plot(loss_erm);
hold on
plot(loss_rlm);
legend('EMR', 'RLM');
title('ERM v.s RLM');
ylabel('empirical square loss l');
xlabel('W/i');
fprintf("step3.2: Finish compare ERM and RLM, please press enter to continue!\n")
pause;
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";
```

```
% plot the data along with the ERM learned models
figure;
subplot(1,2,1)
hold on;
for d = degrees
    w_erm_vis = erm_w(x, t, d);
    plot(interval,func(w_erm_vis,interval));
end
plot(x,t,'rx');
title('Visualization ERM vs OriginDataPoints');
ylabel('outputs t');
xlabel('inputs x');
legend(label_erm');
subplot(1,2,2)
hold on
% plot the data along with the RLM learned models
for d = [1 5 10 20]
    w_rlm_vis = rlm_w(x, t, d, log(0.001));
    plot(interval,func(w_rlm_vis,interval));
end
% plot origin dataset
plot(x,t,'rx');
title('Visualization RLM (with lambda=0.001) vs OriginDataPoints');
ylabel('outputs t');
xlabel('inputs x');
legend(label_rlm');
fprintf("step5: Finish visualization with ERM and RLM, please press enter to continue!\n")
pause;
clc;close all;
```

`functions()`

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

```
function w = erm_w(x, t, d)
N = size(x, 1);
% design matrix X of the data,
% where N is size of the data and d is degree of poly
% |x1^0 x1^1 ... x1^d|
% |x2^0 X2^1 ... X2^d|
% |...               | = X
% |...               |
% |xN^0 xN^1 ... xN^d|
X = zeros(N,d);
for r = 1:N
    for c = 1:d
        X(r, c) = x(r)^c;
    end
end
% first column would be constant
X = [ones(N,1), X];
% vector w that solves the unregularized least squares linear regression
problem
% ERM solution w = (X'*X)^-1 * X' * t from slide,
% where X is design matrix of the data
% w = (X' * X)^-1 * X' * t;
w = pinv(X' * X) * X' * t;
```

```
function w = rlm_w(x, t, d, ln_lambda)
N = size(x, 1);
% d = 2;
% ln_lambda = -2
lambda = exp(ln_lambda);
% design matrix X of the data,
% where N is size of the data and d is degree of poly
% |x1^0 x1^1 ... x1^d|
% |x2^0 X2^1 ... X2^d|
% |...               | = X
% |...               |
% |xN^0 xN^1 ... xN^d|
X = zeros(N,d);
for r = 1:N
    for c = 1:d
        X(r, c) = x(r)^c;
    end
end
% first column would be constant
X = [ones(N,1), X];
Id = eye(size(X' * X));
```

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

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

```
% yw(xi) = (Xw)i = w0x0+w1x1+..+wdxd
function y = func(w,x)
w_inverse = zeros(1,size(w,1));
for i = 1:size(w)
    w_inverse(i) = w(size(w,1)-i+1);
end
y = polyval(w_inverse,x);
```

```
function avg_loss = cross_vailidation_erm(rank_data,degree,fold)
chunck = size(rank_data,1)/fold; % the number of times of testing
tot_loss = 0; % init total loss
for i = 1:fold
    n=1;
    testing = zeros(chunck, 2);
    % load testing set
    for j = 1+(i-1)*chunck : i*chunck
        testing(n,:) = rank_data(j,:);
        n=n+1;
    end
    % load remaining rank_data for training set
    training = rank_data(~ismember(rank_data,testing,'rows'),:);
    % training our model
    w = erm_w(training(:,1), training(:,2), degree);
    % compute the total loss
    tot_loss = tot_loss + q_loss(w, testing(:,1), testing(:,2));
end
avg_loss = tot_loss/fold;
```

Step 6 - bonus

main.m

```
% STEP 6 bonus
x_b = load("dataset2_inputs.txt");
t_b = load("dataset2_outputs.txt");
interval = -1:0.01:1.25;
[opt_w, min_loss] = train_6(x_b,t_b);
for i = 1:5
    [w,l] = train_6(x_b,t_b);
    if l < min_loss
        min_loss = l;
        opt_w = w;
    end
end
opt_w
plot(interval,func(opt_w,interval));
hold on
plot(x_b,t_b,'rx');
title('Visualization Model vs OriginDataPoints');
ylabel('outputs t');
xlabel('inputs x');
save('opt_w_s6.txt','opt_w');
fprintf("step6: Finish training model, please press enter to continue!\n")
pause;
clc;close all;
```

function()

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

```
        n=1;
        testing = zeros(chunck, 2);
        % load testing set
        for j = 1+(i-1)*chunck : i*chunck
            testing(n,:) = rank_data(j,:);
            n=n+1;
        end
        % load remaining rank_data for training set
        training = rank_data(~ismember(rank_data,testing,'rows'),:);

          % training our model
          % w(:,i) = erm_w(training(:,1), training(:,2), degree);
          % compute the total loss
          % loss(:,i) = q_loss(w(:,i), testing(:,1), testing(:,2));

        n = 1;
        loss_rlm = zeros(1,20);
        w_rlm = zeros(degree+1,20);
        for ln_lambda = 1:20
            % training our model
            w_rlm(:,n) = rlm_w(training(:,1), training(:,2), degree, -
ln_lambda);
            % compute the total loss
            loss_rlm(:,n) = q_loss(w_rlm(:,n), testing(:,1), testing(:,2));
            n=n+1;
        end
        if min(loss_rlm) < min_loss
            min_loss = min(loss_rlm);
            index = loss_rlm==min(loss_rlm);
            opt_w = w_rlm(:,index);
            %flag="RLM";
        end
    end

%     if min(loss) < min_loss
%         min_loss = min(loss);
%         index = loss==min(loss);
%         opt_w = w(:,index);
%         %flag="EMR";
%     end
end
```