ECE 805 – Machine Learning

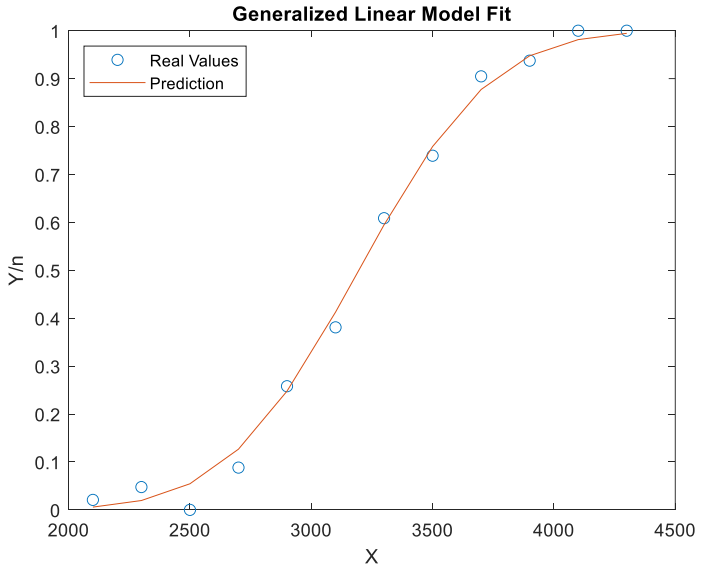
Tutorial 2, Friday, March 06, 2020

Examples:

1. Fit a probit regression model for y on x. Each y(i) is the number of successes in n(i) trials.

x	2100	2300	2500	2700	2900	3100	3300	3500	3700	3900	4100	4300
n	48	42	31	34	31	21	23	23	21	16	17	21
y	1	2	0	3	8	8	14	17	19	15	17	21

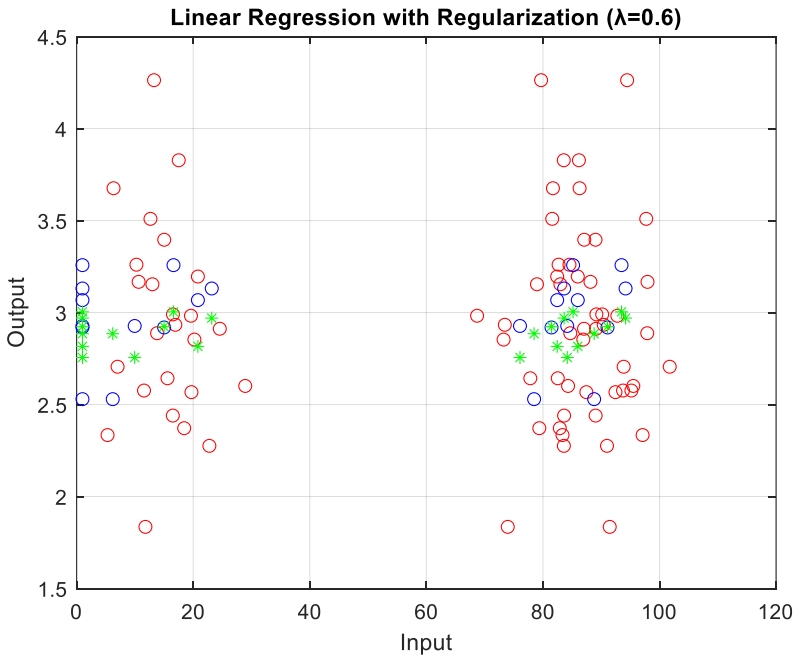
```
%% Example 1
clc; clear all;
close all;

x = [2100 2300 2500 2700 2900 3100 3300 3500 3700 3900 4100 4300]';
% input
n = [48 42 31 34 31 21 23 23 21 16 17 21]'; % trials
y = [1 2 0 3 8 8 14 17 19 15 17 21]';% number of successes in each
trial (y/n)

b = glmfit(x, [y n], 'binomial', 'link', 'probit'); %Fit a
generalized linear model
yfit = glmval(b, x, 'probit', 'size', n); % Predict values for a
generalized linear model
plot(x, y./n, 'o', x, yfit./n, '-') % plot the prediction
xlabel('X')
ylabel('Y/n')
title('Generalized Linear Model Fit')
legend('Real Values','Prediction','Location','northwest')
error=sum((yfit-y).^2) % error
```

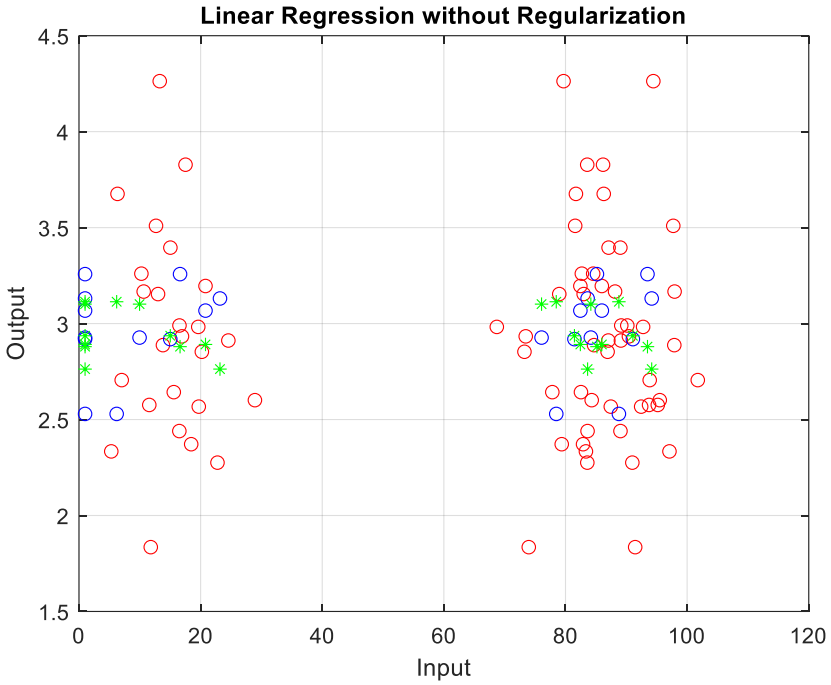


2. Linear Regression without and with Regularization
   - Linear regression without Regularization – Least Squares Solution: $\theta^* = (\Phi\Phi^T)^{-1}\Phi Y$
   - Linear regression with Regularization: $\theta^* = (\lambda I + \Phi\Phi^T)^{-1}\Phi Y$, λ – regularization coefficient
   - Input is the input samples, the target is the output, the test is a small number of samples used for the testing/prediction and the realt is the real value of the test samples.

```
%% Example 2
clear all; clc; close all;
load input
load target
load test
load realt

Fi=[ones(25,1) input'];

target=target';

Wml1=inv(Fi'*Fi)*(Fi'*target); % Linear Regression
Wml2=inv(0.1*eye(4)+Fi'*Fi)*(Fi'*target); % Linear Regression with
regularization coefficient = 0.1
Wml3=inv(0.6*eye(4)+Fi'*Fi)*(Fi'*target);% Linear Regression with
regularization coefficient = 0.6

test=[ones(6,1) test'];
pred3=test*Wml3; % prediction 3 (with regularization coefficient =
0.6 )
%% Prediction 1 plot
pred1=test*Wml1; % prediction 1 (without regularization)

figure(1)
plot(input, target, 'ro', test, realt, 'bo',test, pred1,'g*')
hold on
grid on
xlabel('Input')
ylabel('Output')
title('Linear Regression without Regularization')
%legend('Training samples','','','Testing
samples','','','Prediction 1')
er1=sum((realt-pred1').^2) % error 1 (without regularization)

%% Prediction 2 plot
pred2=test*Wml2; % prediction 2 (with regularization coefficient =
0.1 )
figure(2)
plot(input, target, 'ro', test, realt, 'bo',test, pred2,'g*')
%legend('Training samples', 'Testing samples', 'Prediction 2')
grid on
xlabel('Input')
ylabel('Output')
title('Linear Regression with Regularization (λ=0.1)')
er2=sum((realt-pred2').^2) % error 2 (with regularization
coefficient = 0.1 )
```

```
%% Prediction 3 plot
pred3=test*Wml3; % prediction 2 (with regularization coefficient =
0.1 )
figure(3)
plot(input, target, 'ro', test, realt, 'bo', test, pred3,'g*')
%legend('Training samples', 'Testing samples', 'Prediction 3')
hold on
grid on
xlabel('Input')
ylabel('Output')
title('Linear Regression with Regularization (λ=0.6)')
er3=sum((realt-pred3').^2) % error 3 (with regularization
coefficient = 0.6 )
```





⇨ As the λ increases, the error is decreasing.

3. Linear Regression with a basis function

We have the following values and we want to calculate the linear regression function: $y = w_0 + w_1\varphi(x)$ where $\varphi(x) = \frac{1}{x}$.

x	5	4.3	2	3.3	8	1.5	3.4	7.2	4.3	6.7	9.8	1.2
y	-0.30	-0.19	-0.11	-0.18	-0.20	0.03	0.15	0.31	-0.21	0.23	0.26	0.12

a) Calculate the w_0 and w_1 from the given data below and by using the regularization coefficient to be equal to λ=0.4 .

b) Repeat the above bye using different values for λ = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1] and calculate the error for each value of λ. Which value of λ gives the smaller value of error?

```
%% Example 3 - Linear Regression
clc;
clear all;
close all;

x=[5 4.3 2 3.3 8 1.5 3.4 7.2 4.3 6.7 9.8 1.2]'; % input
t=[-0.30 -0.19 -0.11 -0.18 -0.20 0.03 0.15 0.31 -0.21 0.23 0.26
0.12]'; % target output
f=(1./x); %Basis Function f(x)=1/x
%% Part A
fprintf('\nPart A:\n')
Fi=[ones(12,1) f];
%Regularization coefficient l=0.4
Wml=inv(0.4*eye(2)+Fi'*Fi)*(Fi'*t);
h=Wml(1)+(Wml(2)*f);
error=sum((h-t).^2)
%% Part B
fprintf('\nPart B:\n')
 x=[5 4.3 2 3.3 8 1.5]';
 t=[-0.30 -0.19 -0.11 -0.18 -0.20 0.03]'; % target output
% % input
 f=(1./x);
 Fi=[ones(6,1) f];
%
% Regularization coefficient l
l=[0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
for i=1:length(l)
    W(i,:)=inv(l(i)*eye(2)+Fi'*Fi)*(Fi'*t);
    %disp(W(i,:));
    %fprintf('\n');
end

clear i error;
for i=1:length(l)
    error(i)=0;
    for k=1:length(x)
```

```
        h(i,k)=W(i,1)+(W(i,2)*f(k));
        error(i)=error(i)+((h(i,k)-t(k)).^2);
    end
end

%disp(h)
disp(error)
```

4. Radial Basis Function Network

```
%% Example 4 - RBFN
clear all; clc;
%close all;
X = -1:.1:1; %input
T = [-.9602 -.5770 -.0729  .3771  .6405  .6600  .4609 ...
      .1336 -.2013 -.4344 -.5000 -.3930 -.1647  .0988 ...
      .3072  .3960  .3449  .1816 -.0312 -.2189 -.3201];
% Target
figure()
plot(X,T,'+'); %plot input over output-target
title('Training Vectors');
xlabel('Input Vector X');
ylabel('Target Vector T');

eg = 0.00002; % sum-squared error goal
sc = 0.2;     % spread constant
net = newrb(X,T,eg,sc);
%net=newrbe(X,T,sc);

X = -1:.01:1;
Y = net(X); % Prediction


hold on;
plot(X,Y); % plot input over prediction
hold off;
legend({'Target','Output'})
%%
%Calculate the cost function
cost_function=0;
for i=1:length(T)
    cost_function=cost_function+(T(i)-Y(i))^2;
end
cost_function=cost_function/2;

fprintf('\nCost Function: %.4f \n',cost_function);
```