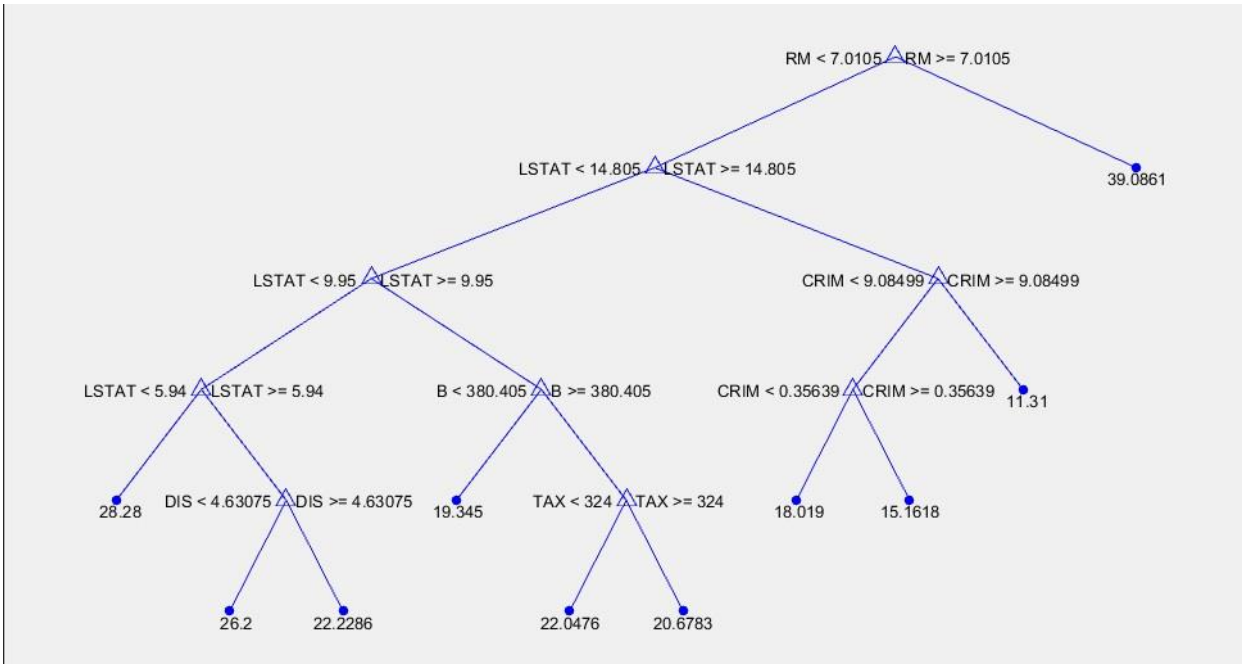
EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

Athar Roshandelpoor
U46981822

## *Problem1*

**a)**



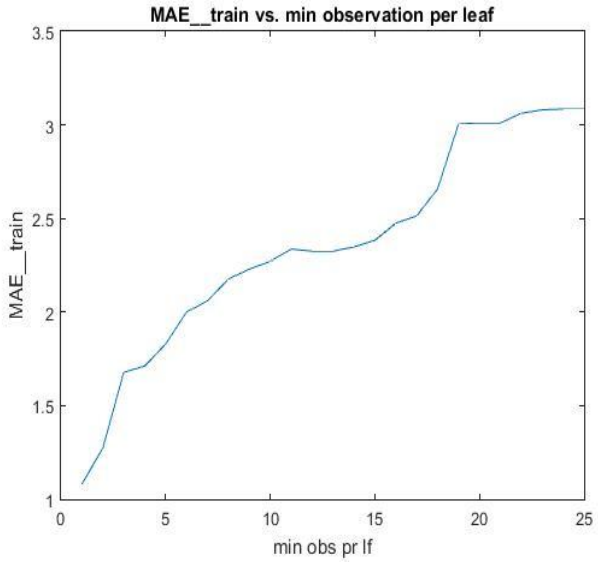

**b)** The estimated MEDV value for the test feature vector: 22.0476

EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

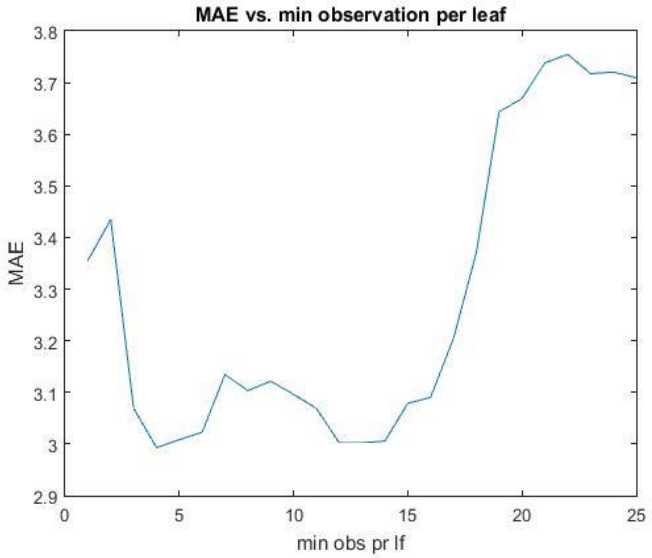
Athar Roshandelpoor
U46981822

**c)**

For train data set




For test data set




**For training data**: as we can see, by decreasing the number of min observation per leaf we get less value in mean absolute error. This makes sense because when we make leaf smaller and smaller, for training data we get the exact prediction. By increasing the min observation in each leaf we will have more points in each sub region so diversity increases, as a result, error increases.

**For test data:** as it is shown in the picture, when the number of min observation per leaf is very large, MAE is also large, which is logical. Because, it means that we did not partition or data enough to get the accurate answer for prediction.

However, when the number of min observation per leaf is very small, again we get a large error. The reason is that, when we divide our training data to very small sub region, actually, we don't learn anything we just memorizing data (overfitting problem). That reason make the error large when we have a small number for min observation per leaf.

EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

Athar Roshandelpoor
U46981822

## *Problem 2*

**a)** i) Yes it is unique. Because it has only one variant (x) and matrix sigma(x) is invertible. So the optimization problem has unique solution.

ii) $w_{OLS}$= 0.0221, $b_{OLS}$ = 2.9743 ,MSE= 0.2588 ,MAE= 0.3174

**b)** i)

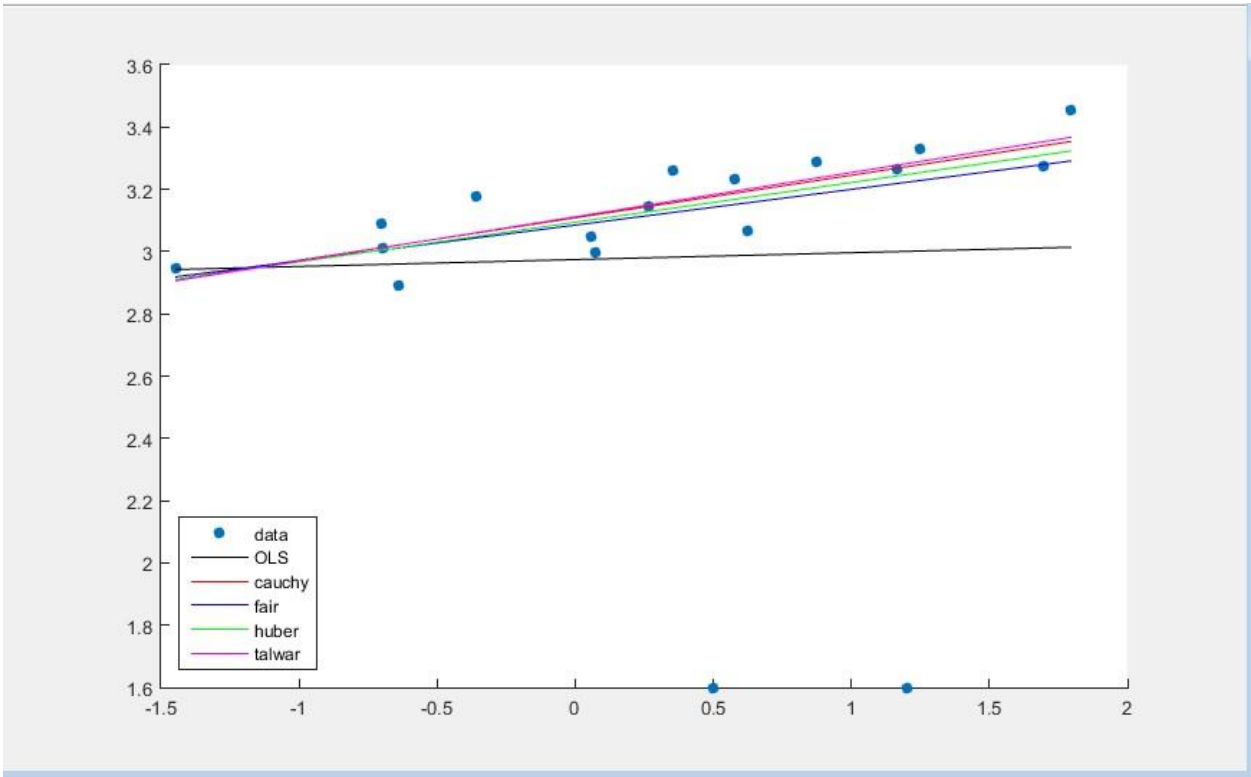
| Method | Mean Absolute Value | Mean Squared Error |
|---|---|---|
| OLS | 0.3174 | 0.2588 |
| Cauchy | 0.2427 | 0.2995 |
| fair | 0.2468 | 0.2861 |
| Huber | 0.2451 | 0.2922 |
| Talwar | 0.2435 | 0.3024 |

ii) $w_{huber}$= 0.1275 ,$b_{huber}$=3.0944

EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

Athar Roshandelpoor
U46981822

iii)




As it is shown in the picture, we have two outliers ([0.5, 1.6] ,[1.2, 1.6]). These outliers have some effects on OLS and pull the OLS line down. Because OLS is not robust enough compare to robust linear regression line. On the other hand, robust fit is more robust and as a result, it follows the trend of the majority of the data points. In the sense of robustness is in the order: talwar> Cauchy> Huber> fair

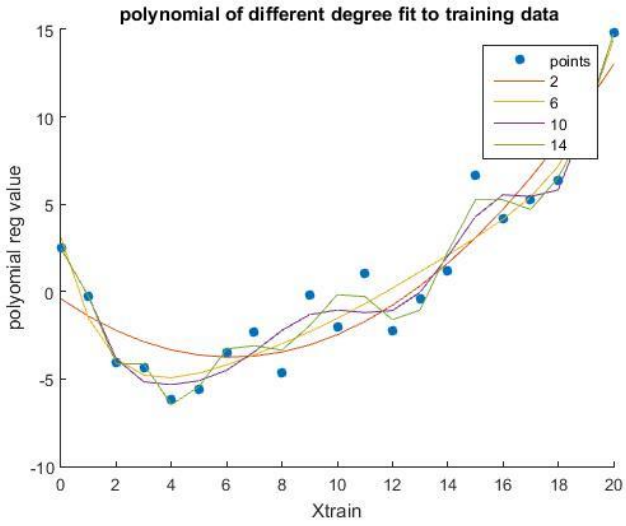
EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

Athar Roshandelpoor
U46981822

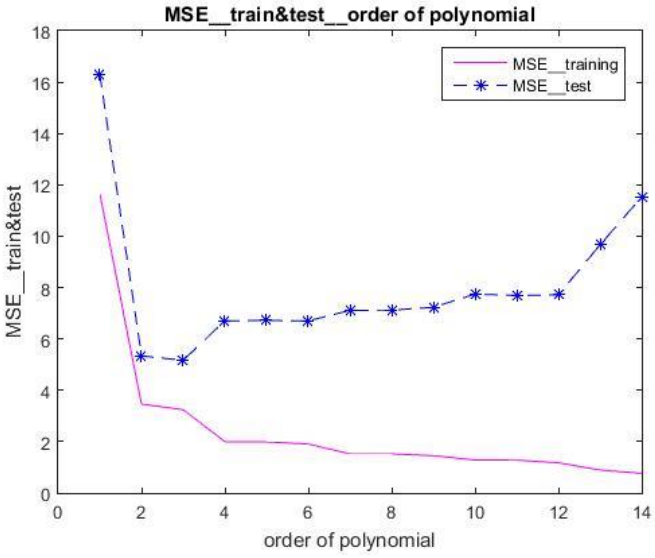
## *Problem3*

**a)**

i)




ii)



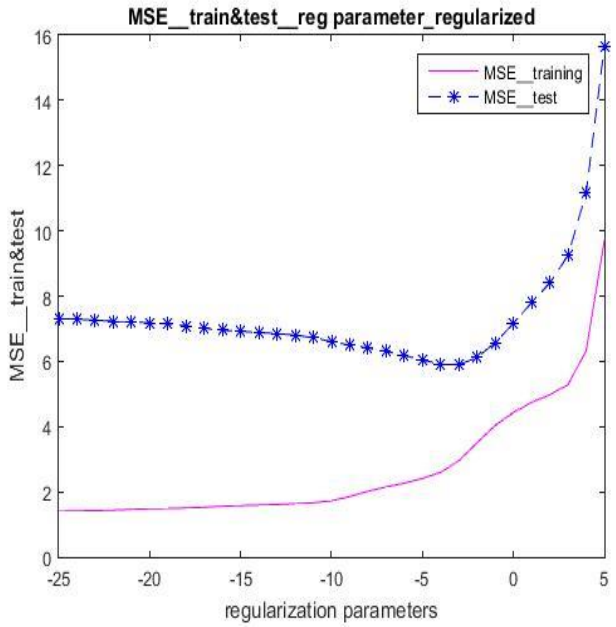

At first by increasing degree of the polynomial, MSE both for training and test data will increases. For **training data** by increasing the degree we pushes the line through the training data so always MSE decreases by increasing the degree od polynomial. However, for **test data**, at some point MSE starts increasing again, this is because of the problem of overfitting. We only have data so by increasing degree of polynomial we pushes the polynomial to the training data, in fact, we are not learning anything we just memorize the training data, so the accuracy will decrease.

EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

Athar Roshandelpoor
U46981822

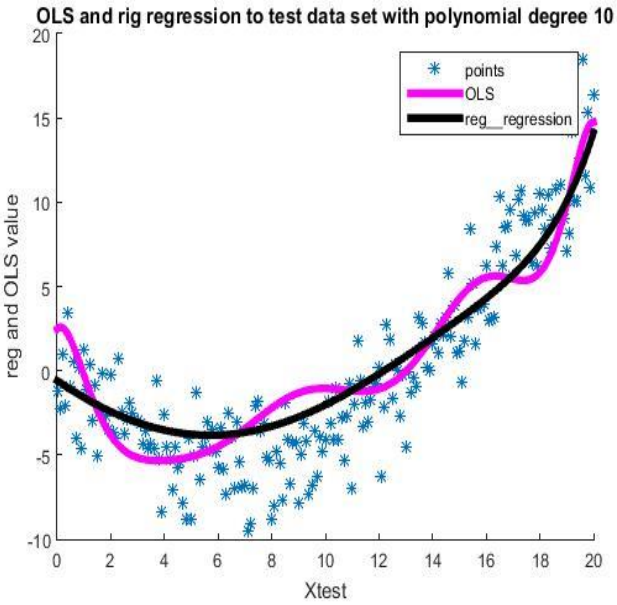
**b)**

i)




We can see that as the regularization parameter gets larger, MSE for training data increases and MSE for test data first decreases and then increases. For training data, since the regularization parameter forces to fit a lower degree polynomial to data, by increasing the parameter we do to go through all data so MSE will increase. For test data we have lower MSE when we find a good enough model for our data (at this point we are kind of avoiding the overfitting problem). However, by increasing the parameter to a very large number we miss a lot of data from training data so again MSE will increases.

ii)

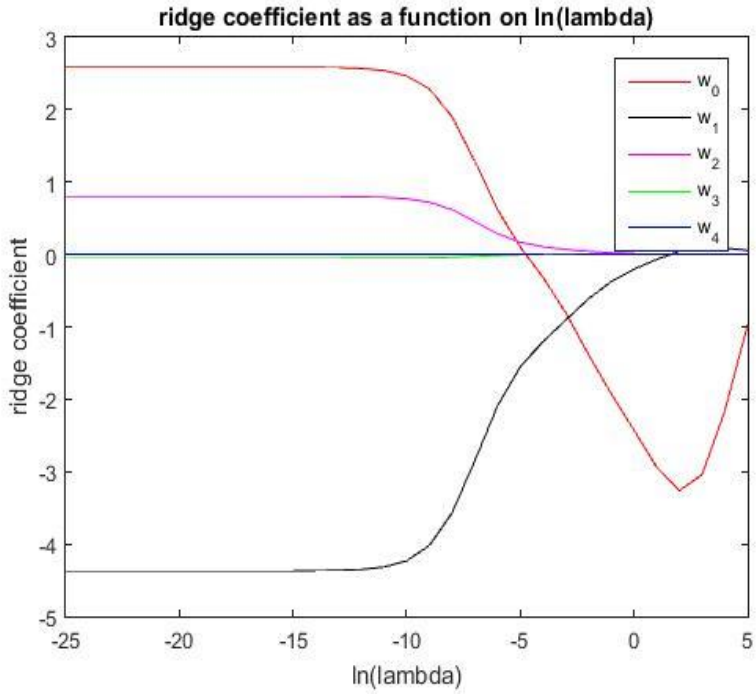



As it is shown, ridge-regression gives a smoother fit compare to OLS. Although both are in degree 10 but the coefficient of rig-regression fit for high degree terms is very small and near zero. So they don't look like in the same degree. This is reasonable since rig- regression imposes a penalty for overfitting.

EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

Athar Roshandelpoor
U46981822

c)




By increasing the lambda w coefficient are going to zero because of the penalty term which is become larger (l2 norm). So the minimum for our objective function achieves when W are zero. So by increasing the lambda all W go to zero. On the other hand, regularization parameter forces to fit a lower degree polynomial to data, as a result, here $W_3$ and $W_4$ are almost zero from the beginning by adding the regularization.

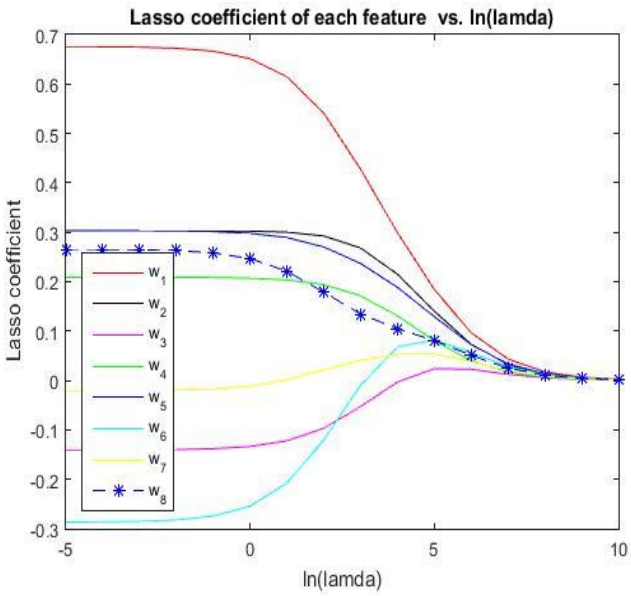
Besides, $W_0$ does not converge to zero because it is not in the penalty term. $W_0$ should converge to mean of the y values.

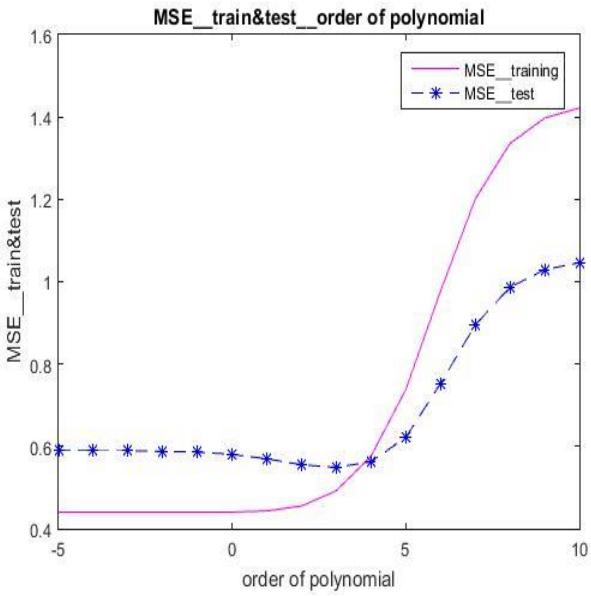
EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

Athar Roshandelpoor
U46981822

## *Problem4*

a)



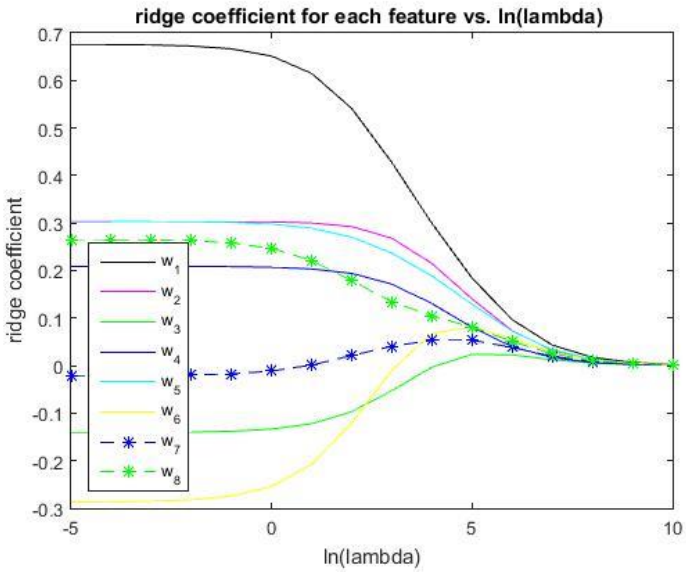

b) As it is shown in the picture, Lasso coefficients converge to zero when lambda increases. This is because when we have a very large penalty term in our objective function the first term (which has w and it is influenced by data) is negligible and the penalty term is dominant. Since the penalty term in always nonnegative the minimum value for that is zero so we get all the w zero.

We can see that the first two W are converging last. Which means that the 2 most meaningful terms (dominant features) are lcavol - log (cancer volume), lweight -log (prostate weight). We can thus discard other predictors and only use lcavol and lweight to fit a regression model to predict lpsa (log (prostate specific anti-gen)). It means that for lpsa, among 8 features these two are the most dominant ones.

Also for Lasso coefficients, from beginning we have some almost zero coefficients and some coefficients which are not close to zero. It is a sparse matrix, which means that some of W s are zero or almost zero and coefficients do not behave in same way. But eventually, by increasing lambda, all of them go to zero.

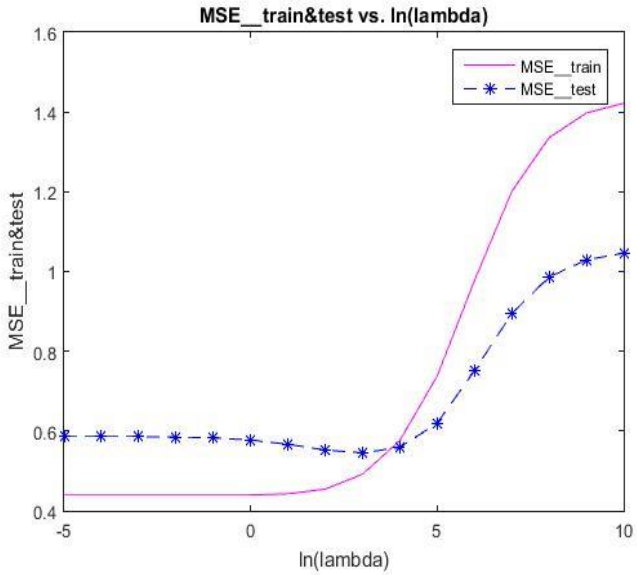
Athar Roshandelpoor
U46981822

c)




As we can see by increasing the lambda, ridge coefficient are again converges to zero. But here is much slower than Lasso coefficients. As we can see here not even exactly converging to zero (although some number really close to zero). Here all the coefficient are converging to zero approximately at same time and it is not like lasso that some w converges later than others. This is because ridge regression imposes a l2 norm penalty on objective function while lasso regression imposes a l1 norm penalty. The comparison shows that lasso regression does yield sparse estimates of the parameters.

d)




Same argument as 3-b prat i.

EC503: Learning From Data
Matlab Assignment 3
April 14, 2016

Athar Roshandelpoor
U46981822

## *Code*

**Athar_matlab3_prob1**

```
%%
%training tree
clear
clc

load('housing_data.mat')
t = classregtree(Xtrain, ytrain, 'method','regression ', 'names',...
    {'CRIM','ZN','INDUS','CHAS','NOX','RM','AGE','DIS','RAD','TAX','PTRATIO','B','LSTAT'},...
    'prune','off','minleaf',20);
view(t)

%%
%testing
inst=[5,18,2.31,1,0.5440,2,64,3.7,1,300,15,390,10];
prediction = eval(t, inst)

%%
for i=1:25
    tt = classregtree(Xtrain, ytrain, 'method','regression ', 'names',...
        {'CRIM','ZN','INDUS','CHAS','NOX','RM','AGE','DIS','RAD','TAX','PTRATIO','B','LSTAT'},...
        'prune','off','minleaf',i);


    pred_test=eval(tt,Xtest);
    MAE(i)=(1/253)*sum(abs(pred_test-ytest));

    pred_train=eval(tt,Xtrain);
    MAE_train(i)=(1/length(ytrain))*sum(abs(pred_train-ytrain));

end
figure(1)
plot(1:25,MAE)
title('MAE vs. min observation per leaf')
xlabel('min obs pr lf')
ylabel('MAE')
figure(2)
plot(1:25,MAE_train)
title('MAE__train vs. min observation per leaf')
xlabel('min obs pr lf')
ylabel('MAE__train')
```

**athar_matlab3_prob2**

```
%%
clear
clc
load('linear_data.mat');
Mux=mean(xData);
Muy=mean(yData);
sigmax=var(xData);
sigmaxy=(1/18)*(xData-Mux*ones(18,1))'*(yData-Muy*ones(18,1));
w=sigmaxy/sigmax;
b=Muy-w*Mux;
MSE=0;
MAE=0;
for i=1:18
   MSE=MSE+(abs( yData(i)-(w*xData(i)+b)))^2;
   MAE=MAE+abs( yData(i)-(w*xData(i)+b));
end
MSE=MSE/18
```

Athar Roshandelpoor
U46981822

```
MAE=MAE/18

%%
a1=robustfit(xData,yData,'cauchy');%cauchy
a2=robustfit(xData,yData,'fair');%fair
a3=robustfit(xData,yData,'huber');%huber
a4=robustfit(xData,yData,'talwar');%talwar


%intial zero
MSE1=0;
MAE1=0;
MSE2=0;
MAE2=0;
MSE3=0;
MAE3=0;
MSE4=0;
MAE4=0;
for i=1:18
   MSE1=MSE1+(abs( yData(i)-(a1(1)+xData(i)*a1(2))))^2;
   MAE1=MAE1+abs( yData(i)-(a1(1)+xData(i)*a1(2)));

    MSE2=MSE2+(abs( yData(i)-(a2(1)+xData(i)*a2(2))))^2;
   MAE2=MAE2+abs( yData(i)-(a2(1)+xData(i)*a2(2)));

    MSE3=MSE3+(abs( yData(i)-(a3(1)+xData(i)*a3(2))))^2;
   MAE3=MAE3+abs( yData(i)-(a3(1)+xData(i)*a3(2)));

    MSE4=MSE4+(abs( yData(i)-(a4(1)+xData(i)*a4(2))))^2;
   MAE4=MAE4+abs( yData(i)-(a4(1)+xData(i)*a4(2)));

end

MSE1=MSE1/18
MAE1=MAE1/18
MSE2=MSE2/18
MAE2=MAE2/18
MSE3=MSE3/18
MAE3=MAE3/18
MSE4=MSE4/18
MAE4=MAE4/18

scatter(xData,yData,'filled')
hold on
plot(xData,w*xData+b,'k')
plot(xData,a1(2).*xData+a1(1),'r')
plot(xData,a2(2).*xData+a2(1),'b')
plot(xData,a3(2).*xData+a3(1),'g')
plot(xData,a4(2).*xData+a4(1),'m')
legend('data','OLS','cauchy','fair','huber','talwar','Location','southwest')
```

**athar_matlab3_prob3**

```
%% Part A
clear
clc
load('quad_data.mat')
figure(1)
scatter(xtrain,ytrain,'filled')
title(' polynomial of different degree fit to training data')
xlabel('Xtrain')
ylabel('polyomial reg value')
hold on

for d=1:14
```

Athar Roshandelpoor
U46981822

```
power=repmat([1:d],21,1);
xtrain_rev=repmat(xtrain,1,d);
xtrain_rev=xtrain_rev.^power;

b = ridge(ytrain,xtrain_rev,0,0);
%testing data
power=repmat([1:d],201,1);
xtest_rev=repmat(xtest,1,d);
xtest_rev=xtest_rev.^power;

hold on
if d==2%||6||10||14
plot(xtrain,xtrain_rev*b(2:d+1)+b(1))
else if d==6
plot(xtrain,xtrain_rev*b(2:d+1)+b(1))
else if d==10
plot(xtrain,xtrain_rev*b(2:d+1)+b(1))
else if d==14
plot(xtrain,xtrain_rev*b(2:d+1)+b(1))
    end
    end
    end
end
MSE_training(d)=(1/21)*sum((xtrain_rev*b(2:d+1)+b(1)-ytrain).^2);
MSE_test(d)=(1/201)*sum((xtest_rev*b(2:d+1)+b(1)-ytest).^2);
end

legend('points','2','6','10','14')
hold off
figure(2)
plot(1:14,MSE_training,'m',1:14,MSE_test,'--*b')
title('MSE__train&test__order of polynomial')
xlabel('order of polynomial')
ylabel('MSE__train&test')
legend('MSE__training','MSE__test')



%% Part B
clear all
clc
load('quad_data.mat')
d=10;
figure(3)


power=repmat([1:d],21,1);
xtrain_rev=repmat(xtrain,1,d);
xtrain_rev=xtrain_rev.^power;
%testing data
power=repmat([1:d],201,1);
xtest_rev=repmat(xtest,1,d);
xtest_rev=xtest_rev.^power;

for i=-25:5
   lambda=exp(i);
   b = ridge(ytrain,xtrain_rev,lambda,0);
   MSE_training(i+26)=(1/21)*sum((xtrain_rev*b(2:d+1)+b(1)-ytrain).^2);
   MSE_test(i+26)=(1/201)*sum((xtest_rev*b(2:d+1)+b(1)-ytest).^2);

end
plot(-25:5,MSE_training,'m',-25:5,MSE_test,'--*b')
title('MSE__train&test__reg parameter_regularized')
xlabel('regularization parameters')
ylabel('MSE__train&test')
legend('MSE__training','MSE__test')
% find the lamda with smallest testing MSE
[~,I]=min(MSE_test);
lambda=exp(I-26);
```

Athar Roshandelpoor
U46981822

```
b_reg = ridge(ytrain,xtrain_rev,lambda,0);
b =ridge(ytrain,xtrain_rev,0,0);

figure(4)
scatter(xtest,ytest,'*')
hold on
plot(xtest,xtest_rev*b(2:d+1)+b(1),'m',xtest,xtest_rev*b_reg(2:d+1)+b_reg(1),'k','LineWidth',4)
title('OLS and rig regression to test data set with polynomial degree 10')
xlabel('Xtest')
ylabel('reg and OLS value')
legend('points','OLS','reg__regression')
%% Part C

clear all
clc
load('quad_data.mat')
d=4;
figure(5)


power=repmat([1:d],21,1);
xtrain_rev=repmat(xtrain,1,d);
xtrain_rev=xtrain_rev.^power;

for i=-25:5
   lambda=exp(i);
   b(:,i+26) = ridge(ytrain,xtrain_rev,lambda,0);

end

plot(-25:5, b(1,:),'r',-25:5,b(2,:),'k',-25:5,b(3,:),'m',-25:5,b(4,:),'g',-25:5,b(5,:),'b')
title('ridge coefficient as a function on ln(lambda)')
xlabel('ln(lambda)')
ylabel('ridge coefficient')
legend('w_0','w_1','w_2','w_3','w_4','w_5')
```

**athar_matlab3_prob4**

```
%% Part A
clear
clc
load('prostateStnd.mat')

   %centering Data

  Xtrain_cent=Xtrain-repmat(mean(Xtrain),67,1);
  ytrain_cent=ytrain-repmat(mean(ytrain),67,1);
  Xtest_cent=Xtest-repmat(mean(Xtest),30,1);
  ytest_cent=ytest-repmat(mean(ytest),30,1);
  a=2*sum((Xtrain_cent).^2);
  w=[];

  isconvergent = 0;
  %
for i=-5:10
   lambda=exp(i);
   W = ridge(ytrain,Xtrain,lambda,0);
   w(:,1)=W(2:9);
   w0(i+6)=W(1);
   t=2;
      %w(:,2)= w(:,1)+ones(8,1);
   while(~isconvergent)
      for j=1:8
          c(j)=2*(Xtrain_cent(:,j))'*(ytrain_cent- Xtrain_cent*w(:,t-1)+w(j,t-
1)*Xtrain_cent(:,j));
          w(j,t)= wthresh(c(j)/a(j),'s',lambda/a(j));
      end
```

Athar Roshandelpoor
U46981822

```
      isconvergent = ( norm(w(:,t)-w(:,t-1))<0.0001);
      disp(norm(w(:,t)-w(:,t-1)))
      t=t+1;
   end
   W_plot(:,i+6)=w(:,t-1);
   MSE_train(i+6)=(1/length(ytrain))*sum(((Xtrain_cent* w(:,t-1))+w0(1,i+6)-ytrain).^2);
   MSE_test(i+6)=(1/length(ytest))*sum(((Xtest_cent* w(:,t-1))+w0(1,i+6)-ytest).^2);

end
figure(3)

plot(-5:10, W_plot(1,:),'r',-5:10,W_plot(2,:),'k',-5:10,W_plot(3,:),'m',-
5:10,W_plot(4,:),'g',...,
    -5:10,W_plot(5,:),'b',-5:10,W_plot(6,:),'c',-5:10,W_plot(7,:),'Y',-5:10,W_plot(8,:),'--*b')

title('Lasso coefficient of each feature  vs. ln(lamda)')
xlabel('ln(lamda)')
ylabel('Lasso coefficient')
legend('w_1','w_2','w_3','w_4','w_5','w_6','w_7','w_8','Location','southwest')


figure(4)
plot(-5:10,MSE_train,'m',-5:10,MSE_test,'--*b')
title('MSE__train&test__order of polynomial')
xlabel('order of polynomial')
ylabel('MSE__train&test')
legend('MSE__training','MSE__test')



%% Part C
clear
clc
 load('prostateStnd.mat')
for i=-5:10
   lambda=exp(i);
   b(:,i+6) = ridge(ytrain,Xtrain,lambda,0);
   MSE_train(i+6)=(1/length(ytrain))*sum(((Xtrain* b(2:9,i+6))+b(1,i+6)-ytrain).^2);
   MSE_test(i+6)=(1/length(ytest))*sum(((Xtest* b(2:9,i+6))+b(1,i+6)-ytest).^2);

end
figure(1)

plot(-5:10,b(2,:),'k',-5:10,b(3,:),'m',-5:10,b(4,:),'g',...,
    -5:10,b(5,:),'b',-5:10,b(6,:),'c',-5:10,b(7,:),'Y',-5:10,b(8,:),'--*b',-5:10,b(9,:),'--*g')
legend('w_1','w_2','w_3','w_4','w_5','w_6','w_7','w_8','Location','southwest')


title('ridge coefficient for each feature vs. ln(lambda) ')
xlabel('ln(lambda)')
ylabel('ridge coefficient')

figure(2)
plot(-5:10,MSE_train,'m',-5:10,MSE_test,'--*b')
title('MSE__train&test vs. ln(lambda)')
xlabel('ln(lambda)')
ylabel('MSE__train&test')
legend('MSE__train','MSE__test')
```