Question 2 – Source Code

```
M = csvread('student.csv');

%Response
y = M(:,1);
%Predictor
x = ones(length(y),2);
x(:,2) = M(:,2);

mean_of_data = mean(x);
std_of_data = std(x);

x = (x-mean(x))./std(x);
x(isnan(x)) = 1;
%Initialize beta
beta = rand(1,2);

gama = 0.001;
error = zeros(1,100);
prev_error = 0;

for ind = 1:100
    i = mod(ind,length(y)) + 1;
    fi = 1 - sigmf(x * transpose(beta),[1,0]);
    change = transpose(transpose(x) * (fi-y));
    error(ind) = calculate_mse_error((1-sigmf(x * transpose(beta),[1,0])), y);
    if abs(error(ind) - prev_error) < 0.001
        break
    end
    beta = beta + gama * change;
    prev_error = error(ind);
end

x_try = 0:800;
x_try = (x_try-mean(x(:,2))./std(x(:,2)));
x_try = horzcat(ones(801,1),transpose(x_try));
x_try(:,2) = (x_try(:,2) - mean_of_data) / std_of_data;

result = zeros(1,801);
result = (1-sigmf(x_try * transpose(beta),[1,0]));
figure;
plot(1:801,result);
hold on;
title("Pr(Y = 1 | X = x)");
ylabel("Probability");
xlabel("X");

function error = calculate_mse_error(y_pred, y)
    error = 0;
    for i = [1:length(y)]
        error = error + (y_pred(i) - y(i)) * (y_pred(i) - y(i));
    end
    error = error ./ length(y);
end
```

Question 3 – Source Code

```
%Preperation
train = load('PS2Q3train.mat');
test = load('PS2Q3test.mat');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%      OLS      %%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ols_train_error = zeros(1,10);
ols_test_error = zeros(1,10);

for p = [0:10]
    %Train
    x_train = create_polynomial(x_tr,p);
    beta = (inv(transpose(x_train)*x_train))*transpose(x_train)*y_tr;
    y_pred = x_train * beta;

    %Test
    x_test = create_polynomial(x_te,p);
    y_tested = x_test * beta;

    %Calculate errors
    ols_train_error(p+1) = calculate_error(y_pred,y_tr);
    ols_test_error(p+1) = calculate_error(y_tested,y_te);
end

optimal_p = find(ols_test_error == min(ols_test_error(:)));

figure;
plot(0:10,ols_train_error);
hold on;
plot(0:10,ols_test_error);
title("Test Error vs Train error with OLS");
legend("Train Error","Test Error");
ylabel("Error");
xlabel("p");
optimal_p = 5;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%      RIDGE      %%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ridge_train_error = zeros(1,101);
ridge_test_error = zeros(1,101);

%Built polynomials
x_train = create_polynomial(x_tr,optimal_p);
x_test = create_polynomial(x_te,optimal_p);

%Normalize input data, bias is equal to NAN in that case
x_train = (x_train-mean(x_train))./std(x_train);
x_train(isnan(x_train)) = 1;

x_test = (x_test-mean(x_test))./std(x_test);
```

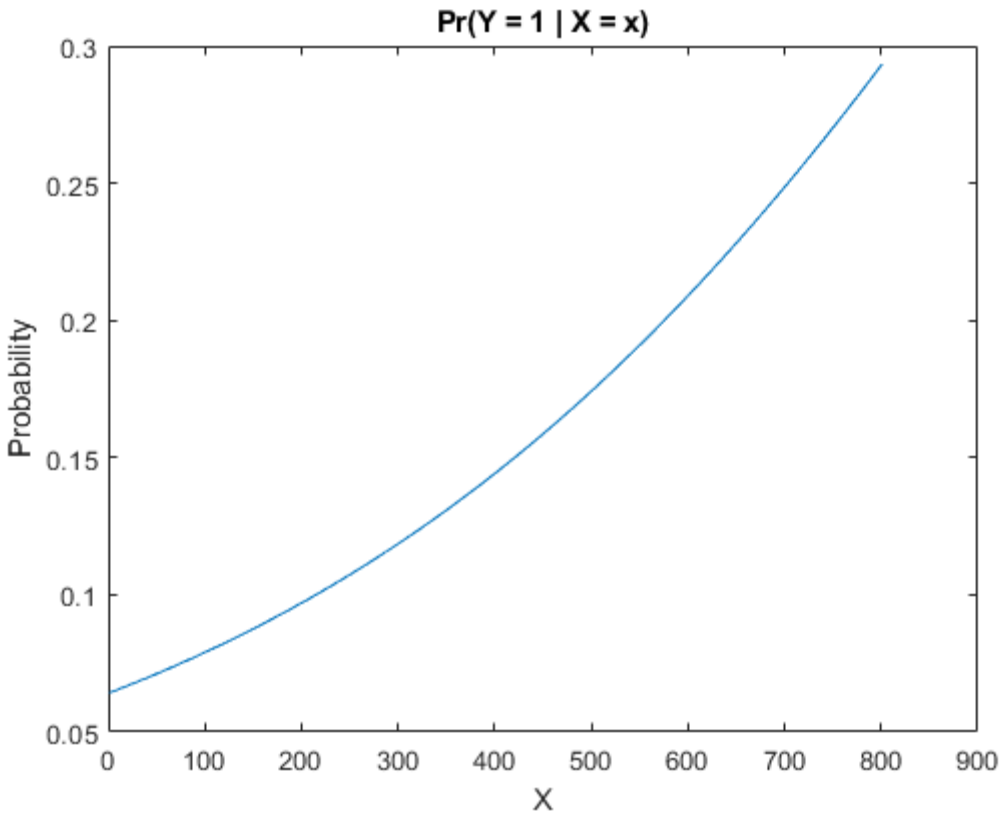
```
x_test(isnan(x_test)) = 1;

y_tr = (y_tr-mean(y_tr));
y_te = (y_te-mean(y_te));

for lambda = [1:100]
    %Train and Test
    beta = (inv(transpose(x_train)*x_train + lambda .*
eye(size(x_train,2))))*transpose(x_train)*y_tr;
    y_pred = x_train * beta;
    y_tested = x_test * beta;

    %Calculate errors
    ridge_train_error(lambda+1) = calculate_error(y_pred,y_tr);
    ridge_test_error(lambda+1) = calculate_error(y_tested,y_te);
end

ridge_train_error(1) = ols_train_error(optimal_p);
ridge_test_error(1) = ols_test_error(optimal_p);

figure;
plot(0:100,ridge_train_error);
hold on;
plot(0:100,ridge_test_error);
title("Test Error vs Train error with Ridge");
legend("Train Error","Test Error");
ylabel("Error");
xlabel("Lambda");

disp("Hello")
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%     LASSO     %%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
lasso_train_error = zeros(1,101);
lasso_test_error = zeros(1,101);

%Built polynomials
x_train = create_polynomial(x_tr,optimal_p);
x_test = create_polynomial(x_te,optimal_p);

%Normalize input data, bias is equal to NAN in that case
x_train = (x_train-mean(x_train))./std(x_train);
x_train(isnan(x_train)) = 1;

x_test = (x_test-mean(x_test))./std(x_test);
x_test(isnan(x_test)) = 1;

y_tr = (y_tr-mean(y_tr));
y_te = (y_te-mean(y_te));

for lambda = [1:100]
    %Train and Test
    beta = lasso(x_train,y_tr,'NumLambda',lambda);

    y_pred = x_train * beta;
```

```
    y_tested = x_test * beta;

    %Calculate errors
    lasso_train_error(lambda+1) = calculate_error(y_pred,y_tr);
    lasso_test_error(lambda+1) = calculate_error(y_tested,y_te);

end

lasso_train_error(1) = ols_train_error(optimal_p);
lasso_test_error(1) = ols_test_error(optimal_p);

figure;
plot(0:100,lasso_train_error);
hold on;
plot(0:100,lasso_test_error);
title("Test Error vs Train error with Lasso");
legend("Train Error","Test Error");
ylabel("Error");
xlabel("Lambda");
beta
disp("Hello")

function x_poly = create_polynomial(x, p)
    x_poly = ones(length(x),1);
    for i = 1:p
        x_p = x.^i;
        x_poly =  horzcat(x_poly,x_p);
    end
end

function error = calculate_error(y_pred, y)
    error = 0;
    for i = [1:length(y)]
        error = error + (y_pred(i) - y(i)) * (y_pred(i) - y(i));
    end
    error = error ./ length(y);
end
```

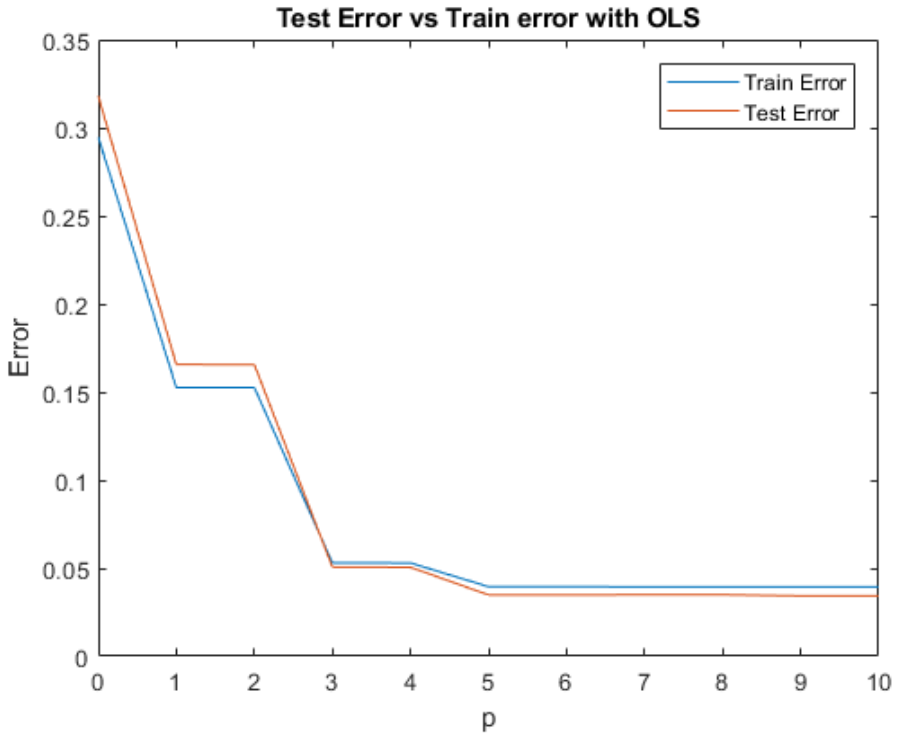
Question 4 – Source Code

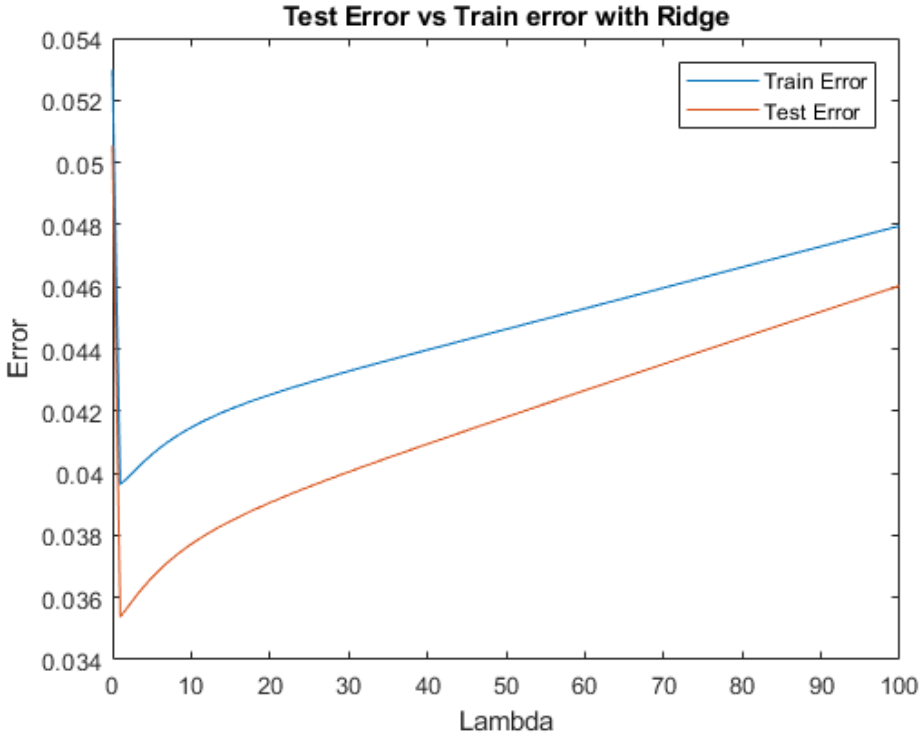
```
test = load('PS2Q4.mat');

%Response
y = y_tr;

%Predictor
x = ones(length(y),2);
x(:,2) = x_tr;

x = (x-mean(x))./std(x);
x(isnan(x)) = 1;

%Initialize beta
beta = rand(1,2);

gama = 0.1;

error = zeros(1,100);

for i = 1:1
    gradient = 0;

    for j = [1:1000]
        gradient = gradient - y(i) * x(i) + exp(x(i) * beta)* x(i);
    end
    gradient = gradient ./1000;

    beta = beta - gama * gradient;

    y_pred = zeros(length(y),1);
    for k = [1:1000]
        y_pred(k) = pdf('Poisson',exp(x(k,:) * transpose(beta)),k);
    end

    error(i) = calculate_mse_error(y_pred, y);
end

figure;
plot(1:100,error);
hold on;
title("Pr(Y = 1 | X = x)");
ylabel("Probability");
xlabel("X");

function error = calculate_mse_error(y_pred, y)
    error = 0;
    for i = [1:length(y)]
        error = error + (y_pred(i) - y(i)) * (y_pred(i) - y(i));
    end
    error = error ./ length(y);
end
```

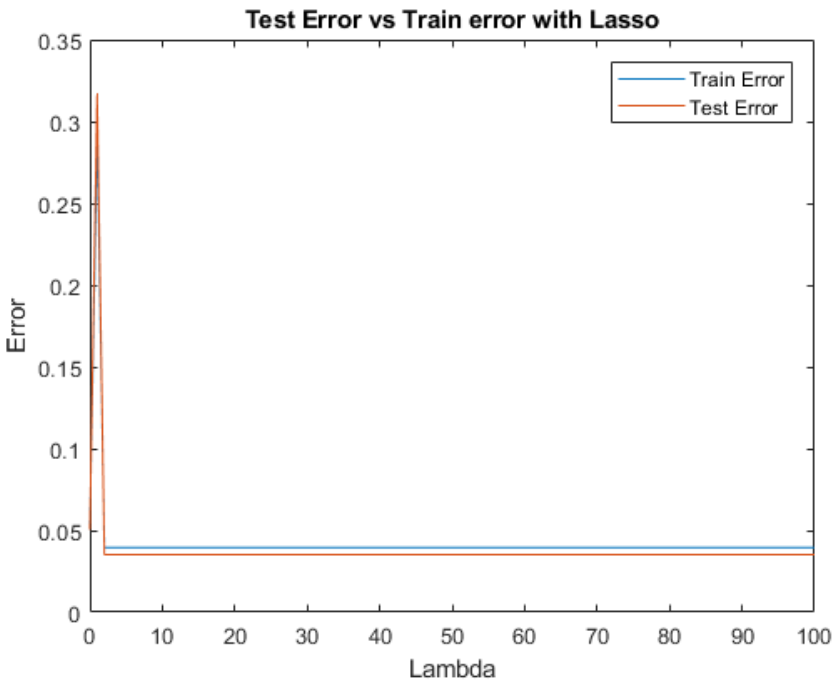
Question 2 – Plot



Question 3 – Plot






Test Error vs Train error with Lasso
Train Error
Test Error
Error
Lambda
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
10
20
30
40
50
60
70
80
90
100

EEE – 485

Statistical Learning and Data Analytics

Problem Set #2

Kerem Ayöz

21501569