Homework 3, CPSC 8420, Fall 2020

Sadeghi Tabas, Sadegh

October 29, 2020

Solution to Problem 1

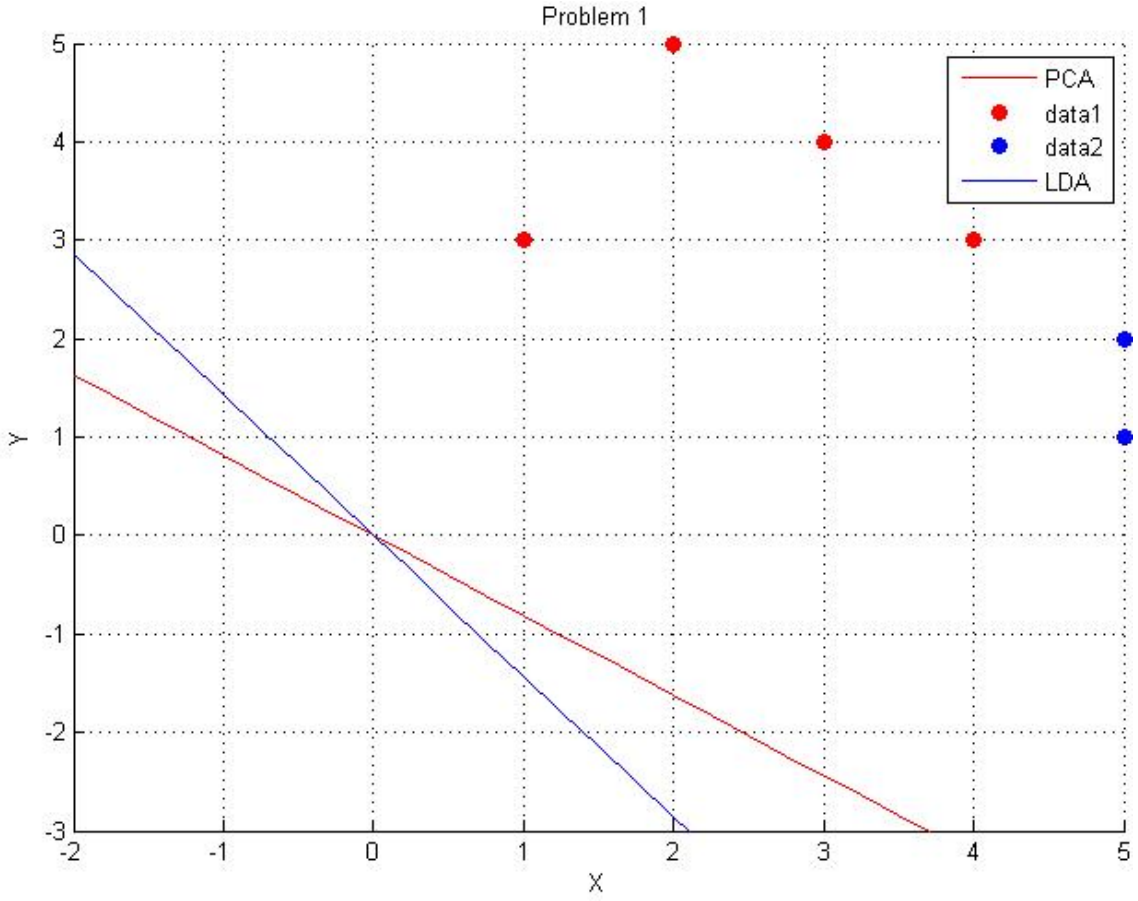

Figure 1: The PCA and LDA results (the code is attached)

Solution to Problem 2

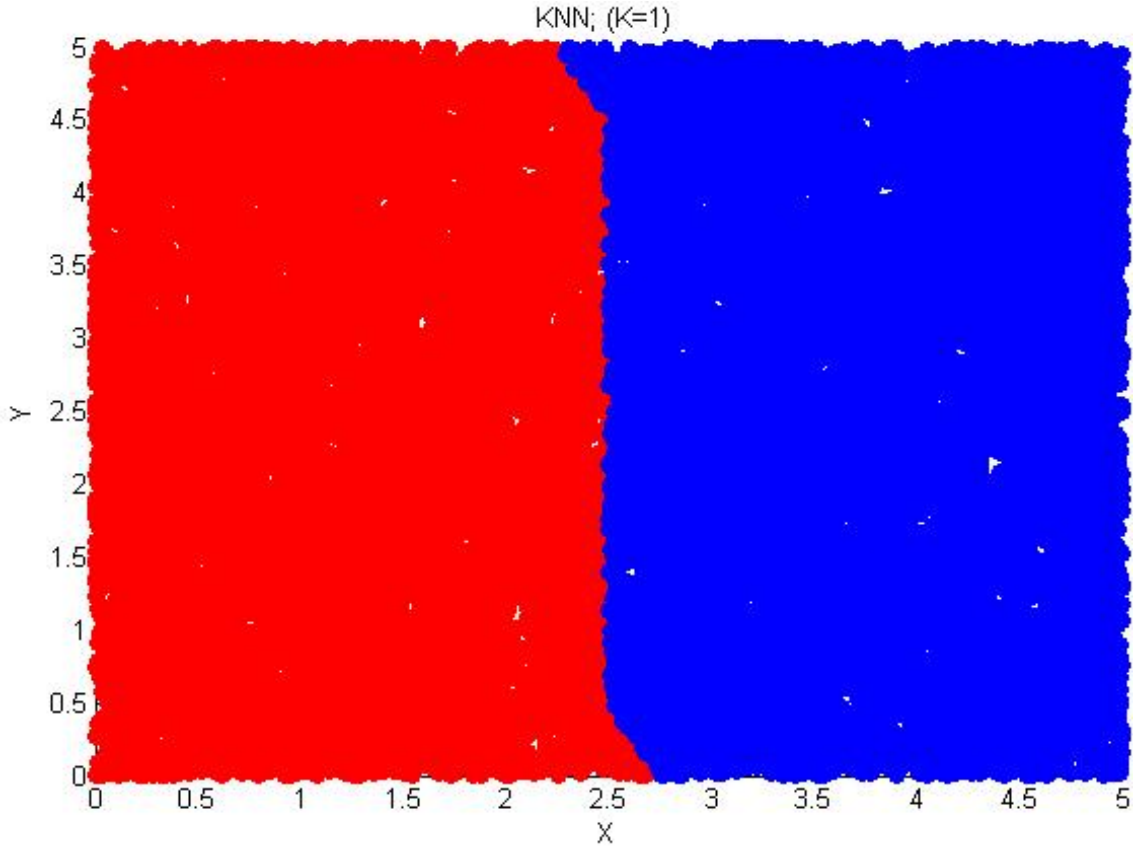

Figure 2: KNN for k=1

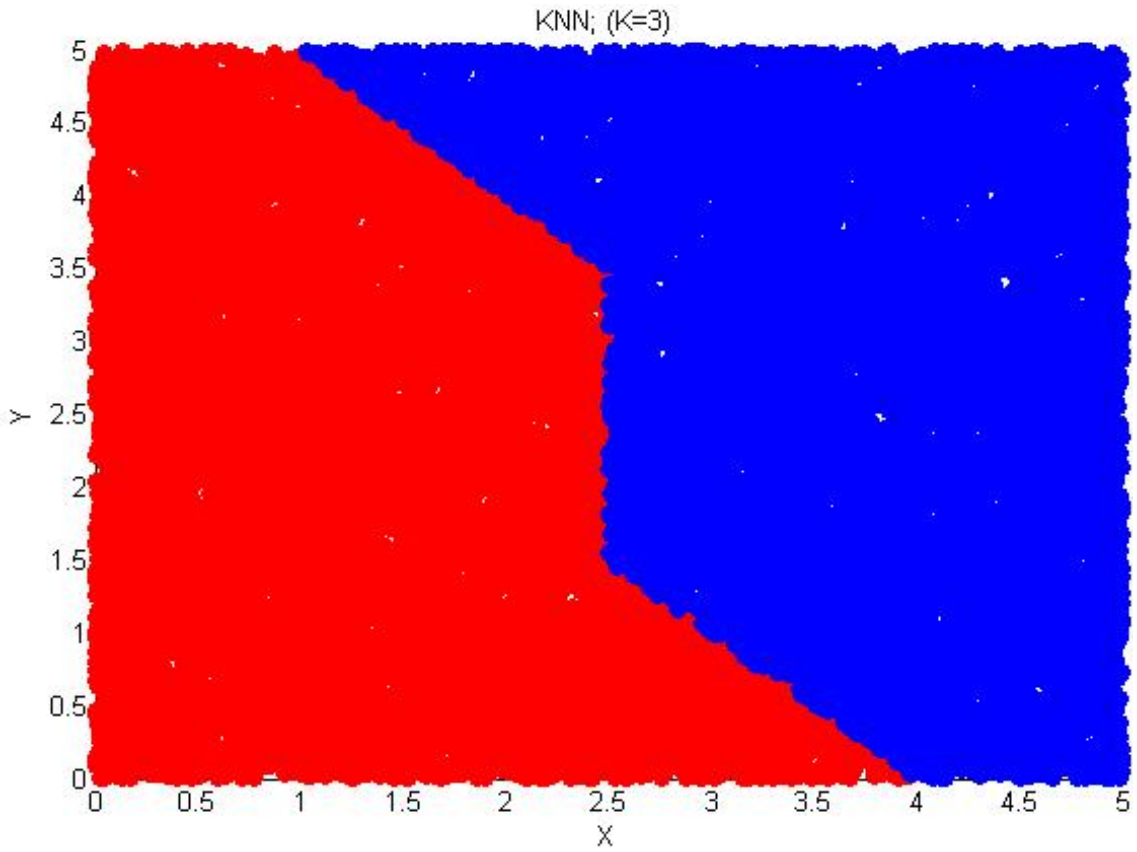

Figure 3: KNN for k=3

Solution to Problem 3

3-1- Assuming $\xi < 0$, the constraint $y_i(\omega^T x_i + b) \geq 1 - \xi_i$ will satisfy if $\xi_i = 0$ therefore the objective function will be lower, so it is not the optimal solution, thus the optimal value of objective function will be same whether this constraint is exist or not, so we can drop it.

3-2-

$$L = \frac{1}{2}\omega^T\omega + \frac{C}{2}\sum_{i=1}^{m}\xi_i^2 - \sum_{i=1}^{m}\alpha_i(y_i(\omega^T x_i + b) - 1 + \xi_i); \alpha_i \geq 0 \tag{1}$$

3-3-

$$\frac{\delta L}{\delta \omega} = 0 = \omega - \sum_{i=1}^{m}\alpha_i y_i x_i \hookrightarrow \omega = \sum_{i=1}^{m}\alpha_i y_i x_i \tag{2}$$

$$\frac{\delta L}{\delta b} = 0 = -\sum_{i=1}^{m}\alpha_i y_i \hookrightarrow \sum_{i=1}^{m}\alpha_i y_i = 0 \tag{3}$$

$$\frac{\delta L}{\delta \xi} = 0 = C\xi_i - \alpha_i \hookrightarrow \xi_i = \frac{\alpha_i}{C} \tag{4}$$

3-4-

$$W(\alpha) : MinL(\omega, \beta, \xi, \alpha) \tag{5}$$

$$= \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}(\alpha_i y_i x_i)^T(\alpha_j y_j x_j) + \frac{1}{2}\sum_{i=1}^{m}\frac{\alpha_i}{\xi_i}\xi_i^2 - \sum_{i=1}^{m}\alpha_i(y_i((\sum_{j=1}^{m}\alpha_j y_j x_j)^T x_i + b) - 1 + \xi_i) \tag{6}$$

$$= -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j (x_i)^T x_j + \frac{1}{2}\sum_{i=1}^{m}\alpha_i\xi_i - b\sum_{i=1}^{m}\alpha_i y_i + \sum_{i=1}^{m}\alpha_i + \sum_{i=1}^{m}\alpha_i\xi_i \tag{7}$$

$$= \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j x_i^T x_j - \frac{1}{2}\sum_{i=1}^{m}\alpha_i\xi_i \tag{8}$$

$$= \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j x_i^T x_j - \frac{1}{2}\sum_{i=1}^{m}\frac{\alpha_i^2}{C} \tag{9}$$

Dual form:

$$Max_\alpha \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j x_i^T x_j - \frac{1}{2}\sum_{i=1}^{m}\frac{\alpha_i^2}{C} \tag{10}$$

Subject to:

$$\alpha_i \geq 0; \sum_{i=1}^{m}\alpha_i y_i = 0 \tag{11}$$

3-5- When C decreases ($C \hookrightarrow 0$) the solution will be determined by the regularization term (the first term), so that the bias goes high and the variance goes low; but when C increases ($C \hookrightarrow \infty$) it takes away all the regularization bias but also lose its variance reduction.

```
% Code for the Problem 1
clc
clear
close all

%% Problem 1 Part 1:
data=[1 3;2 5;3 4;4 3;5 2;5 1];             % add the data
%scatter(data(:,1),data(:,2),'fill','r');   % scatter plot
axis([-2 5 -3 5]);
title('Problem 1')
xlabel('X')
ylabel('Y')

hold on

%% Problem 1 Part 2:
mu = mean(data);         % Calculate average of the dataset for each column
mu = repmat(mu,length(data),1)

newData = (data-mu);     % self centering the dataset
[U,S,V]=svd(newData'*newData);  % Singular Value Decomposition
% Y1 = U(1,1)* newData(1,1) + U(2,1)* newData(1,2);
% Y6 = U(1,1)* newData(6,1) + U(2,1)* newData(6,2);

% plot the first principal component
x = -2:0.25:5;      % Defines the domain as [-2,5]

% the slope is U(2,1)/U(1,1)based on the derived solution in our class
m=U(2,1)/U(1,1);      % Define Slope for the 1st principal component
x1 = 0;               % Origin x
y1 = 0;               % Origin y
y = m*(x - x1) + y1;  % the line equation
h = plot(x,y, 'r');   % Plot the 1st PCA line with red color
grid on
hold on

%% Problem 1 Part 3:
data0 = data(1:4,:);  % Assign the first class
data1 = data(5:6,:);  % Assign the second class
scatter(data0(:,1),data0(:,2),'fill','r');   % scatter plot of first class
scatter(data1(:,1),data1(:,2),'fill','b');   % scatter plot of second class
data0 = data0';
data1 = data1';
mu0 = mean(data0,2);  % Mean value of first class
mu1 = mean(data1,2);  % Mean value of second class

% Calculation of Sw based on what we derived in our class
Sw0=0;
Sw1=0;
for i=1:4
    Sw0 = Sw0 + (data0(:,i)-mu0)*(data0(:,i)-mu0)';     % calc Sw part 1
end
for i=1:2
    Sw1 = Sw1 + (data1(:,i)-mu1)*(data1(:,i)-mu1)';     % calc Sw part 2
end
```

```
Sw=Sw0+Sw1;           % calculation of Sw (summation of the above two parts)
W=inv(Sw)*(mu0-mu1); % calculation of w based on the solution we drived in class

% Plot the LDA line
x = -2:0.25:5; % Defines the domain as [-2,5]
% the slope is W(2,1)/W(1,1)
m  = W(2,1)/W(1,1);  % Define Slope for the LDA
x1 = 0;              % Origin x
y1 = 0;              % Origin y
y = m*(x - x1) + y1; % the line equation
h = plot(x,y, 'b');  % Plot the LDA line with blue color
legend('PCA','data1','data2','LDA');    % legend
```

```
% Code for the Question 2
clc
clear
close all

data = [1 1;2 2;2 3;3 2;3 3;4 4];  % Define the training dataset
labels = [1 1 1 2 2 2]';           % Define classes
k = 3;                             % Define k
test = 5*rand(20000,2);            % Generate random data to test the model

% Calculation of Euclidean Distances between each testing data point
% and the training data
for i=1:length(test)
    for j=1:length(data)
        ED(i,j)=sqrt(sum((test(i,:)-data(j,:)).^2));
    end
    [ED(i,:) index(i,:)]=sort(ED(i,:));  % save ED as  well as indices
end

% Find the k nearest neigbors for each data point of the testing data

nn=index(:,1:k);

for i=1:length(nn)
    nnLabel=labels(nn(i,:)');       % Labels of neareast neigbors
    UnnLabel = unique(nnLabel);     % Remove the repeated labels
    maxCount=0;
    maxLabel=0;
    for j=1:length(UnnLabel)
        n=length(find(nnLabel==UnnLabel(j))); % find the the number of similar ...
            labels
        if n>maxCount
            maxLabel=UnnLabel(j);             % assgin and update the major ...
                labales to maxLabel
            maxCount=n;                       % assign and update the max count ...
                of the label
        end
    end
    test(i,3)= maxLabel;            % assign the selected label to the test point
end

```

```
i=1;
j=1;
for z=1:length(test)
    if test(z,3)==1;
        red(i,:) =test(z,:);        % save the label one in red class
        i=i+1;
    else
        blue(j,:) = test(z,:);      % save the label two in blue class
        j=j+1;
    end
end

scatter(red(:,1),red(:,2), 'fill','r')  % scatter plot of the test points for ...
    class 1
hold on
scatter(blue(:,1),blue(:,2), 'fill','b')% scatter plot of the test points for ...
    class 2
title(['KNN; (K=',num2str(k),') '])
xlabel('X')
ylabel('Y')
```