%MATLAB script for generation of plot figures 1 and 2. This uses k-NN

%algorithm.

filename = 'hw2\_dataProblem.txt'; %MODIFIED to remove line of dashes after the headings

delimiterIn = ' ';

L = [];

P = [];

D = [];

Lpoz = [];

Ppoz = [];

Lneg = [];

Pneg = [];

headerlinesIn = 1;

A = importdata(filename,delimiterIn,headerlinesIn);

L = A.data(:,1);

P = A.data(:,2);

D = A.data(:,3);

data1 = A.data;

distance = [];

pozneighbors = 0;

negneighbors = 0;

%first, we divide the test set into patients positive for the disease or

%negative

for i = 1:1:300

if A.data(i,3) == 1

Lpoz(end+1) = data1(i,1);

Ppoz(end+1) = data1(i,2);

else

Lneg(end+1) = data1(i,1);

Pneg(end+1) = data1(i,2);

end

end

%a plot for pos and negative

scatter(Lpoz,Ppoz,'red','+');

hold on

scatter(Lneg,Pneg,'blue');

hold off

xlabel('Sodium Levels (mEq/L)');

ylabel('Blood Pressure (mmHg)');

figure;

%new data point

L1 = 140.0;

P1 = 116.0;

D1 = 0; % find D1!

%Algorithm

k = 4;

for i = 1:1:300

distance(end+1) = sqrt( (data1(i,1)-L1)^2 + (data1(i,2)-P1)^2 ); %find distance for every data point in respect to the new point

end

data1(:,4) = distance; % append it to data

data1 = sortrows(data1,4); % sort the data according to the distance

for i = 1:1:k %tally ammount of positive and negative neighbors

if data1(i,3) == 1

pozneighbors = pozneighbors + 1;

else

negneighbors = negneighbors + 1;

end

end

% Make determinations about which class prevails over the other, then plot

% the new point black

if pozneighbors >= negneighbors

D1 = 1;

scatter(Lpoz,Ppoz,'red','+');

hold on

scatter(Lneg,Pneg,'blue');

hold on

scatter(L1,P1,'green', '+');

hold off

else

D1 = 0;

scatter(Lpoz,Ppoz,'red','+');

hold on

scatter(Lneg,Pneg,'blue');

hold on

scatter(L1,P1,'green', 'o');

hold off

end

title('Plot with new added point')

xlabel('Sodium Levels (mEq/L)');

ylabel('Blood Pressure (mmHg)');

%MATLAB script for generation of plot figures 3. This uses k-NN

%algorithm.

filename = 'hw2\_dataProblem.txt'; %MODIFIED to remove line of dashes after the headings

delimiterIn = ' ';

Lpoz = [];

Ppoz = [];

Lneg = [];

Pneg = [];

headerlinesIn = 1;

A = importdata(filename,delimiterIn,headerlinesIn);

data1 = A.data;

hitvector = [];

%Algorithm

for k = [1, 3, 5, 7, 9, 11]

data2 = [];

hits = 0;

for j = 1:1:300

D2 = 0;

pozneighbors = 0;

negneighbors = 0;

L1=data1(j,1);

P1=data1(j,2);

D1=data1(j,3);

data1(j,:) = [];

for i = 1:1:299

data1(i,4) = sqrt( (data1(i,1)-L1)^2 + (data1(i,2)-P1)^2 ); %find distance for every data point in respect to the new point, and append to data

end

data1 = sortrows(data1,4); % sort the data according to the distance

for i = 1:1:k %tally ammount of positive and negative neighbors

if data1(i,3) == 1

pozneighbors = pozneighbors + 1;

else

negneighbors = negneighbors + 1;

end

end

% Make determinations about which class prevails over the other, then plot

% the new point black

if pozneighbors >= negneighbors

D2 = 1;

else

D2 = 0;

end

if D1 == D2

hits = hits+1;

end

% put new data in another matrix

data2(end+1,:) = [L1; P1; D2];

% put old data back in the old matrix

data1(:,4) = [];

data1(end+1,:) = [L1; P1; D1];

end

%divide test set into positive and negatives

for i = 1:1:300

if data2(i,3) == 1

Lpoz(end+1) = data2(i,1);

Ppoz(end+1) = data2(i,2);

else

Lneg(end+1) = data2(i,1);

Pneg(end+1) = data2(i,2);

end

end

%a plot for pos and negative

hitvector(end+1) = hits;

end

plot([1,3,5,7,9,11], hitvector);

xlabel('k (even numbers counting to 11)');

ylabel('no. of hits');

title('number of hits vs. k');

%MATLAB script for generation of plot figures 4. This uses the neighborhood algorithm.

filename = 'hw2\_dataProblem.txt'; %MODIFIED to remove line of dashes after the headings

delimiterIn = ' ';

Lpoz = [];

Ppoz = [];

Lneg = [];

Pneg = [];

headerlinesIn = 1;

A = importdata(filename,delimiterIn,headerlinesIn);

data1 = A.data;

hitvector = [];

%Algorithm

for R = 1:0.1:20

data2 = [];

hits = 0;

for j = 1:1:300

D2 = 0;

pozneighbors = 0;

negneighbors = 0;

L1=data1(j,1);

P1=data1(j,2);

D1=data1(j,3);

data1(j,:) = [];

for i = 1:1:299

data1(i,4) = sqrt( (data1(i,1)-L1)^2 + (data1(i,2)-P1)^2 ); %find distance for every data point in respect to the new point, and append to data

end

data1 = sortrows(data1,4); % sort the data according to the distance

for i = 1:1:299 %tally ammount of positive and negative neighbors

if (data1(i,3) == 1) && (data1(i,4) <= R)

pozneighbors = pozneighbors + 1;

elseif (data1(i,3) == 0) && (data1(i,4) <= R)

negneighbors = negneighbors + 1;

end

end

% Make determinations about which class prevails over the other, then plot

% the new point black

if pozneighbors >= negneighbors

D2 = 1;

else

D2 = 0;

end

if D1 == D2

hits = hits+1;

end

% put new data in another matrix

data2(end+1,:) = [L1; P1; D2];

% put old data back in the old matrix

data1(:,4) = [];

data1(end+1,:) = [L1; P1; D1];

end

%divide test set into positive and negatives

for i = 1:1:300

if data2(i,3) == 1

Lpoz(end+1) = data2(i,1);

Ppoz(end+1) = data2(i,2);

else

Lneg(end+1) = data2(i,1);

Pneg(end+1) = data2(i,2);

end

end

%a plot for pos and negative

hitvector(end+1) = hits;

end

plot(1:0.1:20, hitvector);

xlabel('Radius (distance)');

ylabel('no. of hits');

title('number of hits vs. radius R');