Machine Learning
Assignment 2
CLO3 Exercise 19

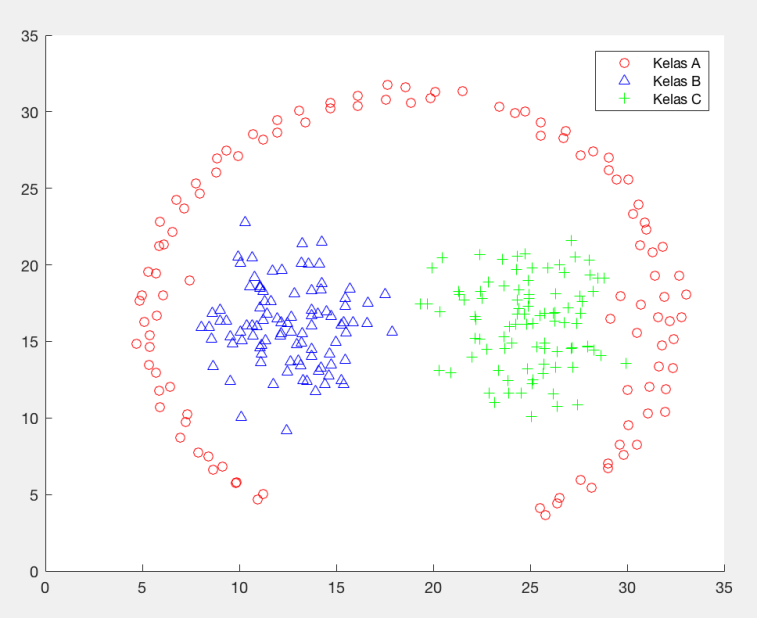
Ida Bagus Dwi Satria Kusuma
1301140297

November 3, 2017

1. In this exercise we will implement Probabilistic Neural Network (PNN) to classify data.

   (a) **(5 points)** Load the selected data set. Visualize all data points using scatter plot. Use different color or symbol for each class. Use attribute 1 as x -axis, attribute 2 as y -axis.

   Jawab: Visualisasi data menggunakan *scatter plot*, di mana simbol 'o' merah adalah data dengan kelas '1', dan simbol 'segitiga' biru adalah data dengan kelas '2', dan '+' hijau adalah data dengan kelas '3'.



(b) (10 points) Select randomly three data for test set (x, y), while remaining data as training set (x, y).

Dengan menggunakan fungsi *datasample* pada MATLAB, kita akan mendapatkan data untuk *test set* secara acak.

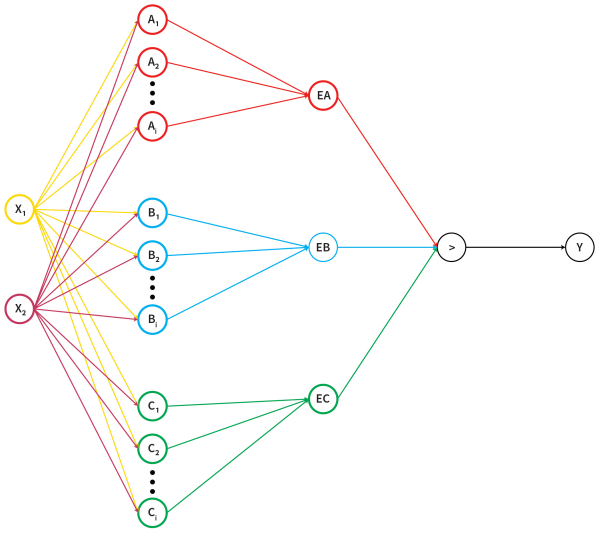
```
% membuat data untuk test secara acak
data_test = datasample(data,3);
```

(c) (15 points) Create a function that implements PNN. Inputs for the function are training set (x, y) and the attributes x of test set. The function outputs the predicted class y for test set.

Untuk mengimplementasikan PNN, kita menggunakan fungsi gaussian

$$y = f(x_1, x_2) = \sum_{i=1}^{n} e^{\frac{-\left((x_1 - x_{1i})^2 + (x_2 - x_{2i})^2\right)}{2\delta^2}}$$

Lalu, desain arsitektur PNN-nya



Code fungsi implementasi dari PNN :

```
function [ y ] = CLO19_pnn( data_training, x_cari )
%By: Ida Bagus Dwi Satria Kusuma / 1301140297
% Fungsi ini hanya digunakan untuk kasus pnn dengan dataset pathbased dari
% nomor 19 CLO 3 Tugas 2 matakuliah machine learning.

% menghitung jumlah atribut
jumlahAtribut = size(data_training,2)-1;

% memasukkan semua keterangan kelas ke dalam matriks data_kelas
data_kelas = data_training(:,jumlahAtribut+1);

% memasukkan semua atribut data training ke dalam matriks X.
dt = data_training(:,1:jumlahAtribut);

% mencari data dengan kelas tertentu
dt_A = dt(find(data_kelas==1),:);
dt_B = dt(find(data_kelas==2),:);
dt_C = dt(find(data_kelas==3),:);

% size_A  = size(dt_A,1);
% size_B = size(dt_B,1);
% size_C = size(dt_C,1);

% pnn dengan sigma = 0.2
sigma = 1;

x1 = x_cari(1,1);
x2 = x_cari(1,2);

% hasil sum kelas A
a_Ay = -(((x1 - dt_A(:,1)).^2+(x2 - dt_A(:,2)).^2)/(2*sigma^2));
% b_A = 1/sqrt(2*pi);
y_Ay = sum(exp(a_Ay));

% hasil sum kelas B
a_By = -(((x1 - dt_B(:,1)).^2+(x2 - dt_B(:,2)).^2)/(2*sigma^2));
% b_B = 1/sqrt(2*pi);
y_By = sum( exp(a_By));

% hasil sum kelas C
a_Cy = -(((x1 - dt_C(:,1)).^2+(x2 - dt_C(:,2)).^2)/(2*sigma^2));
% b_C = 1/sqrt(2*pi);
y_Cy = sum(exp(a_Cy));

y_Sy = [y_Ay y_By y_Cy];
```

```
% mencari nilai maksimum dari hasil sum kelas A, B, dan C
yy = max(y_Sy);

% meng-assign kelas ke dalam variabel y
if yy == y_Ay
    y = 1;
elseif yy == y_By
    y = 2;
else  %if yy == y_Cy
    y = 3;
end

end
```

(d) (5 points) Plot the decision boundary resulted from PNN classifier on the figure that has been created on 19(a). (Hints: First, generate data points using range of minimum and maximum value

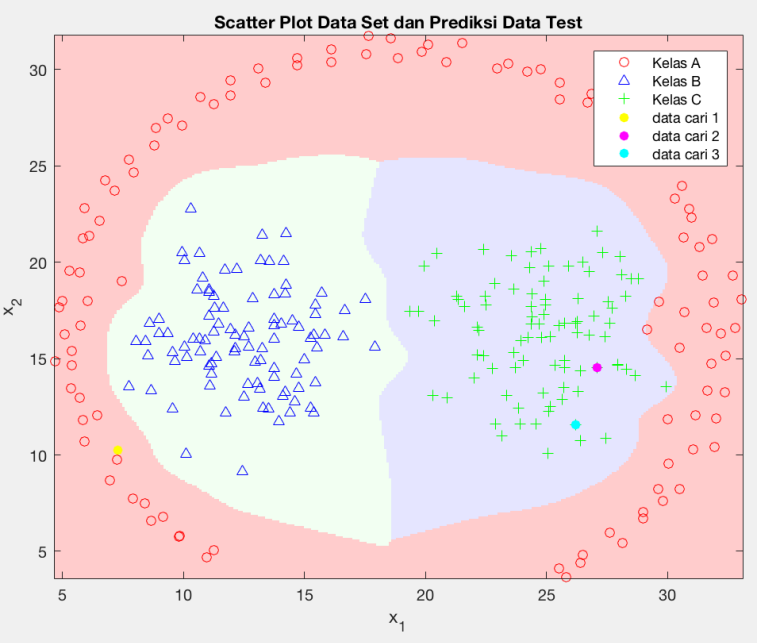
of each attribute, then classify each generated data points using the PNN classifier. Use attribute 1 and attribute 2 as x -axis and y -axis, respectively, of decision boundary location, while the predicted class label for giving coloring). One of online articles that you could learn on how to plot decision boundary using Octave/Matlab is here or go to the next url: http://www.peteryu.ca/tutorials/matlab/visua As alternative, you could use and modify a code program decbound2D that I sent to you. For python, Java, C++ and so on, you could search it on internet.

Dengan memodifikasi fungsi decbound2D menjadi

```
function decbound2D(X1,X2,classifier)

% X1 is vector contains all values of 1st attribute,
% X2 is vector contains all values of 2nd attribute,
% classifier is an abstract of odel of your classifier. You may extend it
% to more than one variable following your classifier design. For example,
% when you use Naive Bayes classifier then you may replace 'classifier'
% above by two variables, namely 'prior' and 'likelihood'

% Original author: Peter Yu, http://www.peteryu.ca/tutorials/matlab/visualize_decision_boundaries
% modified by milo.
data_train = [X1 X2 classifier];

% set up the domain over which you want to visualize the decision
% boundary
xrange = [min(X1) max(X1)];
yrange = [min(X2) max(X2)];

% interval how finely you want to visualize the decision boundary.
interval = 0.1;

% generate grid coordinates for the basis of decision boundary visualization.
[x, y] = meshgrid(xrange(1):interval:xrange(2), yrange(1):interval:yrange(2));

% size of decision boundary image for background of plot.
image_size = size(x);

% make (x,y) pairs as a new data point to be classified which 1st column as
% X1, 2nd column as X2
xy = [x(:) y(:)];


numxypairs = length(xy); % number of (x,y) pairs

% loop through each meshgrid points and get the predicted class
class_prediction = zeros(numxypairs,1);
for ii=1:numxypairs


    z=xy(ii,:);
    % classifiy each new data point xy, put here your code of classification
    % process using your classifier model. The input is each row of 'xy'.
    % Save the predicted class in 'class_prediction'.
    % for example:
    % when ii=1, then the input for your classifier is xy(i,:)
    % the output is saved on class_prediction(ii)
```

```
    % the output is saved on class_prediction(ii)
    class_prediction(ii) = CL019_pnn(data_train,z);

end

% reshape the idx (which contains the class label) into an image.
decisionmap = reshape(class_prediction, image_size);

figure;

%show the image
imagesc(xrange,yrange,decisionmap);
hold on;
set(gca,'ydir','normal');

% set RGB color for colormap for the classes:
cmap = [1 0.8 0.8;      % class 1 = light red
        0.95 1 0.95;    % class 2 = light green
        0.9 0.9 1];     % class 3 = light blue
colormap(cmap);

% label the axes.
xlabel('x_1');
ylabel('x_2');
```

Ketika fungsi tersebut di-plot pada figur 19(a) ditambah dengan mencoba fungsi PNN dengan data_test acak, maka didapatkan :




```
Data 1: 7.3000 10.2500 , y1 = 1.000000
Data 2: 27.100000 14.550000 , y2 = 3.000000
Data 3: 26.200000 11.550000 , y3 = 3.000000
```

(e) (5 points) How good is the classification result on test set? Give your opinion.

Berdasarkan hasil beberapa kali testing, *classifier PNN* selalu

berhasil mengklasifikasikan data dengan benar. Jadi menurut saya, *classifier* dengan PNN sangat bagus.

Referensi

[1] https://id.wikipedia.org/wiki/Regresi_Linier
[2] Introduction to Data Mining - Panning Tan, M. Steinbach
[3] https://en.wikipedia.org/wiki/Nonlinear_regression
[4] Regression book
[5] Regression slide
[6] http://www.nickgillian.com/wiki/pmwiki.php/GRT/MLP
[7] Machine Learning - Tom Mitchell
[8] https://medium.com/towards-data-science/activation-functions-and-its-types-which-is-better-a9a5310cc8f
[9] Slide ANN-MLP Machine Learning
[10] https://www.mathworks.com/help/optim/ug/quadprog.htmlinputarg_f
[11] http://www.robots.ox.ac.uk/ az/lectures/ml/ matlab2.pdf
[12] https://se.mathworks.com/matlabcentral/ answers/104248-implementation-support-vector-machine-nonlinear-case-with-quadprog- function-in-matlab