# Question 1:-

**Assignment 6**

## **Aim:** Implement Fuzzy c-means clustering algorithm. Use IRIS data to evaluate performance of the algorithm.

Compare your results with that of the in-built function.

**Code:-**

clc; clear all;

load iris.dat

setosa = iris((iris(:,5)==1),:); % data for setosa versicolor = iris((iris(:,5)==2),:); % data for versicolor virginica = iris((iris(:,5)==3),:); % data for virginica

obsv\_n = size(iris, 1); % total number of observations

Characteristics = {'sepal length','sepal width','petal length','petal width'};

pairs = [1 2; 1 3; 1 4; 2 3; 2 4; 3 4];

h = figure; for j = 1:6

x = pairs(j, 1);

y = pairs(j, 2);

subplot(2,3,j);

plot([setosa(:,x) versicolor(:,x) virginica(:,x)],...

[setosa(:,y) versicolor(:,y) virginica(:,y)], '.'); xlabel(Characteristics{x},'FontSize',10); ylabel(Characteristics{y},'FontSize',10);

end

cluster\_n = 3; % Number of clusters

expo = 2.0; % Exponent for U

max\_iter = 100; % Max. iteration min\_impro = 1e-6;

% initialize fuzzy partition

U = initfcm(cluster\_n, obsv\_n);

if ishghandle(h) figure(h);

else

for j = 1:6,

x = pairs(j, 1);

y = pairs(j, 2);

subplot(2,3,j);

plot([setosa(:,x) versicolor(:,x) virginica(:,x)],...

[setosa(:,y) versicolor(:,y) virginica(:,y)], '.'); xlabel(Characteristics{x},'FontSize',10); ylabel(Characteristics{y},'FontSize',10);

end

end

% iteration

for i = 1:max\_iter,

[U, center, obj] = stepfcm(iris, U, cluster\_n, expo); fprintf('Iteration count = %d, obj. fcn = %f\n', i, obj);

if i>1 && (abs(obj - lastobj) < min\_impro) for j = 1:6,

subplot(2,3,j);

for k = 1:cluster\_n,

text(center(k, pairs(j,1)), center(k,pairs(j,2)), int2str(k), 'FontWeight', 'bold');

end

end

break; elseif i==1

for j = 1:6,

subplot(2,3,j);

for k = 1:cluster\_n,

text(center(k, pairs(j,1)), center(k,pairs(j,2)), int2str(k), 'color', [0.5 0.5 0.5]);

end

end

end

lastobj = obj;

end

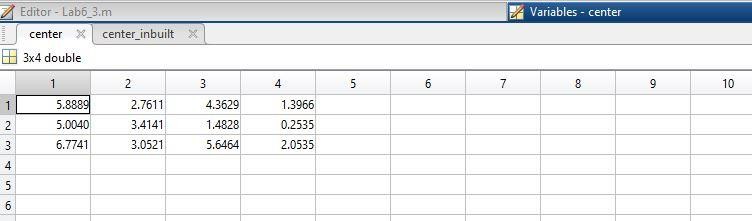
center=center(:,1:4); center=center/10;

%FCM by inbuilt method data = load('iris.csv');

data=data(:,1:4);% load some sample data n\_clusters\_inbuilt = 3; % number of clusters

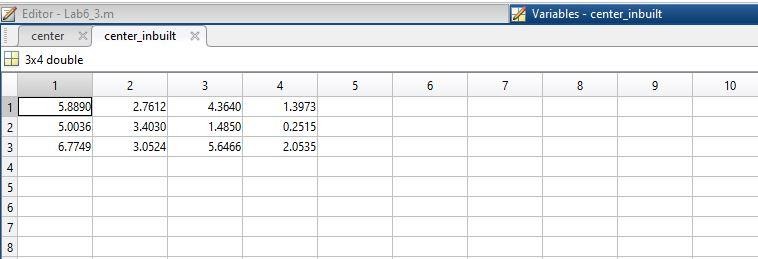
[center\_inbuilt,U\_inbuilt,obj\_fcn\_inbuilt] = fcm(data, n\_clusters\_inbuilt);

# Output:-



**Fig 1. Centroid values without In-built Function**

**Fig 2. Centroid values with In-built Function**



**Conclusion:**

## Centroid values computed by Fig. 1 and Fig. 2 is almost same. In program, center matrix is for centroid values without using in-built function and center\_inbuilt matrix is for centroid values using In-built function.