實驗6

實驗 6.1

代碼：

```
function dataTag = kNN(train, trainTag, k, data)

% FUNCTION
%  dataTag = kNN(train, trainTag, k, data)
% Use k-Nearest-Neighbor classifying algorithm to classify data
%
% INPUT ARGUMENTS:
%  s1 is the number of features in the matrix
%  s2 is the number of Training Data provided
%  s3 is the number of Classifiying Data provided
%
%  train: s1*s2 matrix, the training data
%  trainTag: 1*s2 matrix, the training data tag naming the categories
%  k: the results should be splited into k categories
%  data: s1*s3 matrix, the data to be classified
% OUTPUT ARGUMENTS:
%  dataTag: 1*s3 matrix, the corresponding categories of data

s2 = size(train, 2);
s3 = size(data, 2);

% the returning dataTag
dataTag = zeros(1, s3);

% distanceList is the temporary saving for Euclidean Distance List
% between data(:, i) and every training data
distanceList = zeros(1, s2);

% realk is the real k, aka number of categories, used in the program
realk = min([k s2]);

for i = 1 : s3
    % compute distance between data(:, i) and every training data
    vdata = data(:, i)';
    for j = 1 : s2
        % compute Euclidean Distance
        %distanceList(1, j) = norm(train(:, j)' - data(:, i)');
        vsub = train(:, j)' - vdata;
        distanceList(1, j) = (vsub * vsub');
    end
    % sort the distance, and get the indexes
    [~, index] = sort(distanceList, 'descend');

    tagMap = containers.Map('KeyType', 'int64', 'ValueType', 'int64');
    for kk = 1 : realk
        tag = trainTag(1, index(1, kk));
        if tagMap.isKey(tag) == 0
            tagMap(tag) = 0;
        end

        tagMap(tag) = tagMap(tag) + 1;
    end

    % terrible matlab
    values = tagMap.values;
    values = [values{:}];
    [~, max_value_index] = max(values);
    keys = tagMap.keys;
    dataTag(1, i) = keys{max_value_index};
end
```

```
end
```

實驗結果：

```
homework6_1

pr_err =

    0.8596
```

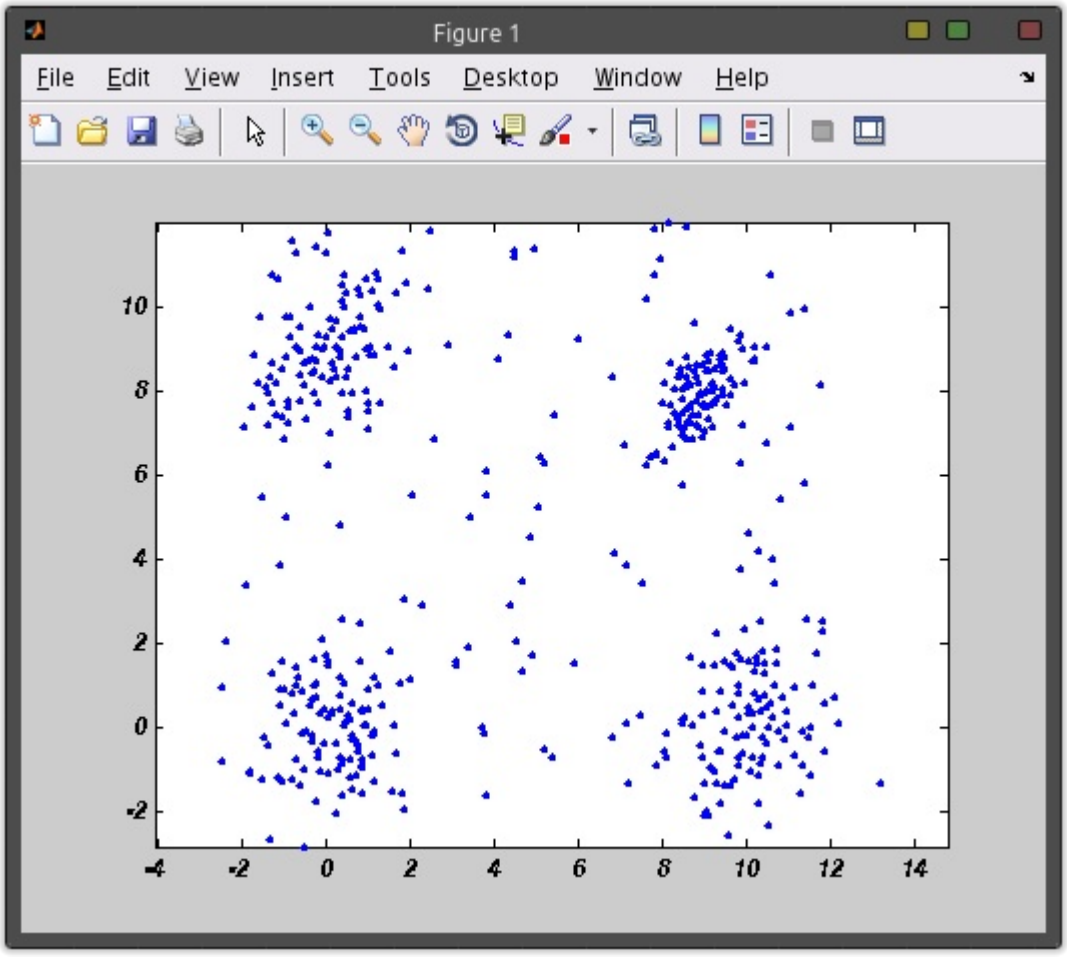
實驗 6.2

代碼：

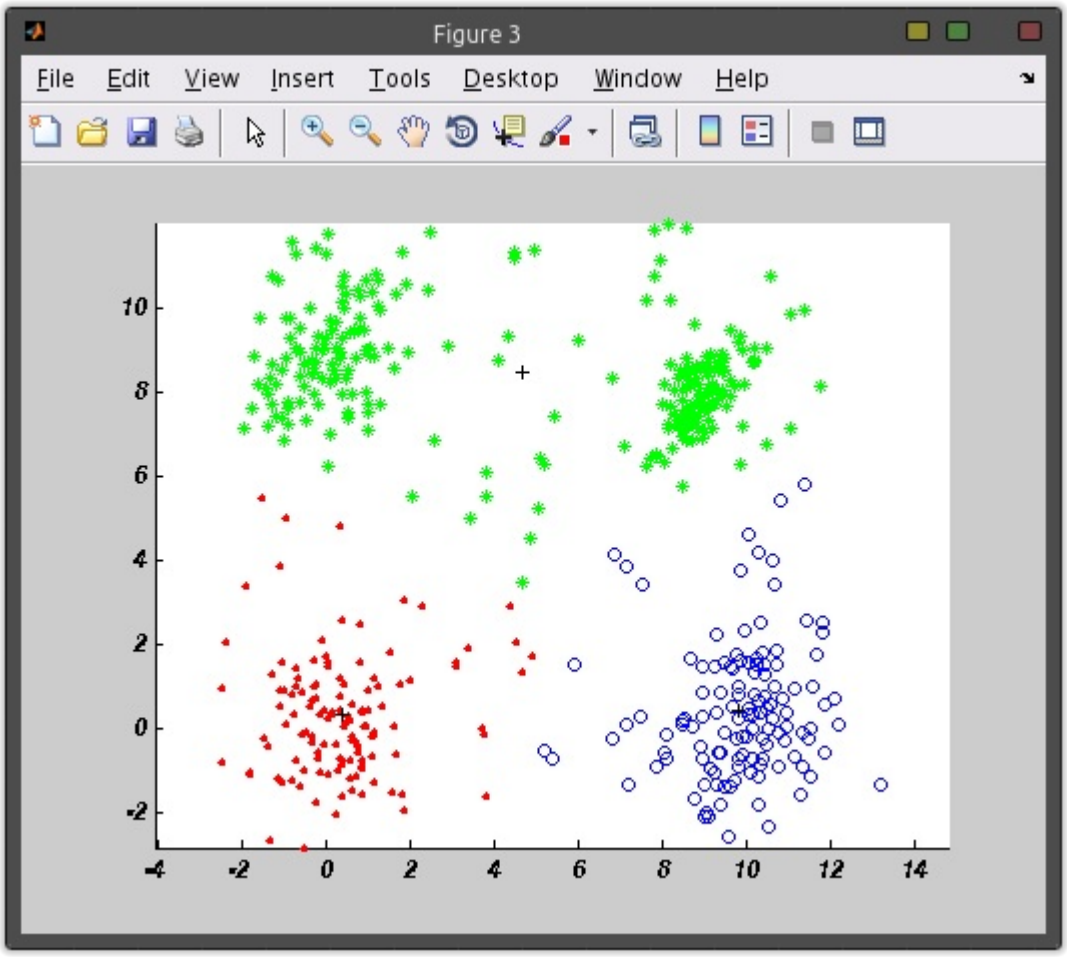
```
function [center, label] = k_means(X, center_init)
% FUNCTION
%  [center, label] = k_means(X, center_init)
%  This is k-means clustering algorithm
% INPUT ARGUMENTS:
%  s1 is the number of features in the input data matrix
%  s2 is the number of data in data matrix
%  s3 is the number of categories to be seperated, aka the 'k'
%
%  X: s1*s2 matrix, input data matrix
%  center_init: s1*s3 matrix, initial center guess matrix
% OUTPUT ARGUMENTS:
%  center: s1*s2 matrix, category center of corresponding data in X
%  label: 1*s2 vector, label of corresponding data in X

s1 = size(X, 1);
s2 = size(X, 2);
s3 = size(center_init, 2);

center = zeros(s1, s2);
label = zeros(1, s2);

center_iter = center_init;

% change_flag indicates whether center and label are changed in the loop
change_flag = 1;

while change_flag == 1
    change_flag = 0;

    % for every data
    for i = 1 : s2
        min_dist = inf;
        min_index = 0;

        % for every center
        for j = 1 : s3
            % calculate the distance between every data and every center
            distance = norm(center_iter(:, j) - X(:, i));
            if distance < min_dist
                min_dist = distance;
                min_index = j;
            end
        end

        % if it is changed
        if label(1, i) ~= min_index
            change_flag = 1;
            label(1, i) = min_index;
        end
```

```
    end

    % for every category, aka every center
    for i = 1 : s3
        points_in_category = find(label == i);
        center_iter(:, i) = mean(X(:, points_in_category)')';
    end
end

for i = 1 : s2
    center(:, i) = center_iter(:, label(1, i));
end

end
```

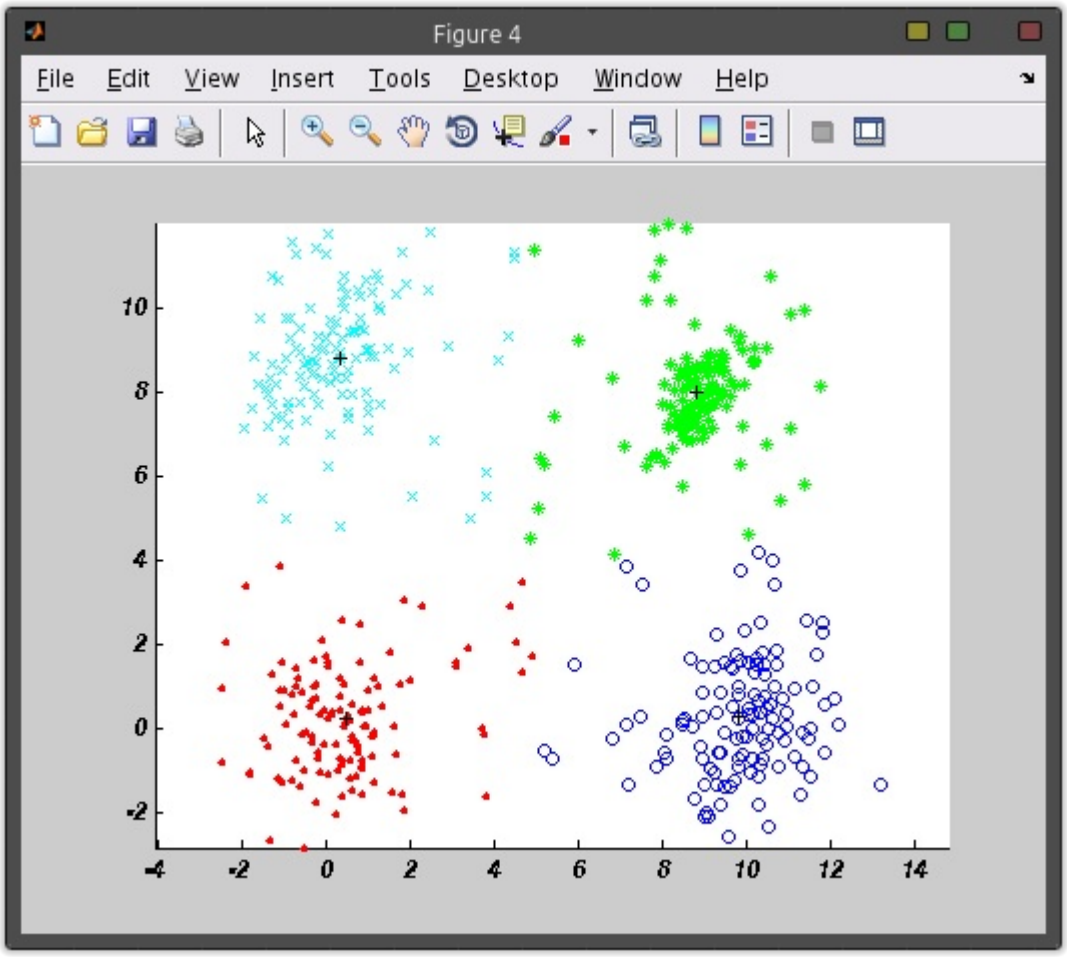
實驗結果：

- 未分類圖:
  - 

- 3類圖:

- 
Figure 3
File Edit View Insert Tools Desktop Window Help
10
8
6
4
2
0
-2
-4 -2 0 2 4 6 8 10 12 14

- 4類圖:

- 
Figure 4
File Edit View Insert Tools Desktop Window Help
10
8
6
4
2
0
-2
-4 -2 0 2 4 6 8 10 12 14

- 5類圖:

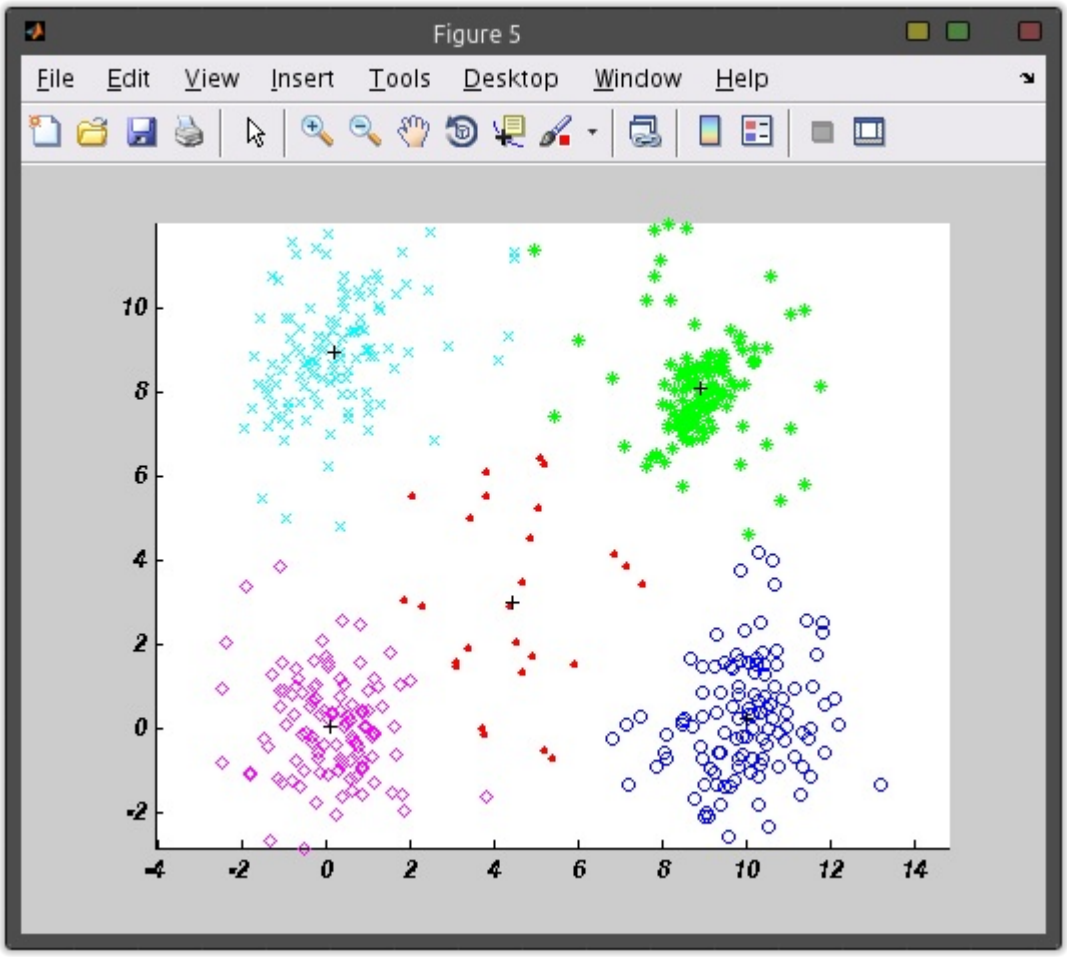


Figure 5
File Edit View Insert Tools Desktop Window Help
10
8
6
4
2
0
-2
-4 -2 0 2 4 6 8 10 12 14