Nama : Adhyfa Fahmy Hidayat

NIM :1301154127

Kelas : IF 39-01

Tugas k-Means

Analisis Masalah

Permasalahan dalam tugas ini adalah mengklasifikasikan data atau *Clustering* menggunakan metode k-Means. Data yang diberikan adalah dataTrainset dan dataTestset. Data Train memiliki 2 atribut yang berjumlah 688 data. Sedangkan Data Test memiliki 2 atribut yang berjumlah 100 data. Baik Data Train dan Data Test tersebut harus di klasifikasi dengan metode k-Means.

Desain

K-Means merupakan algoritma yang bertujuan untuk membagi data menjadi beberapa kelompok. Pembelajaran ini termasuk ke dalam *unsupervised learning.* Inputannya adalah objek data dan banyak nya k yang diinginkan yang sekiranya paling optimum.

Algoritma untuk melakukan k-Means adalah sebagai berikut:

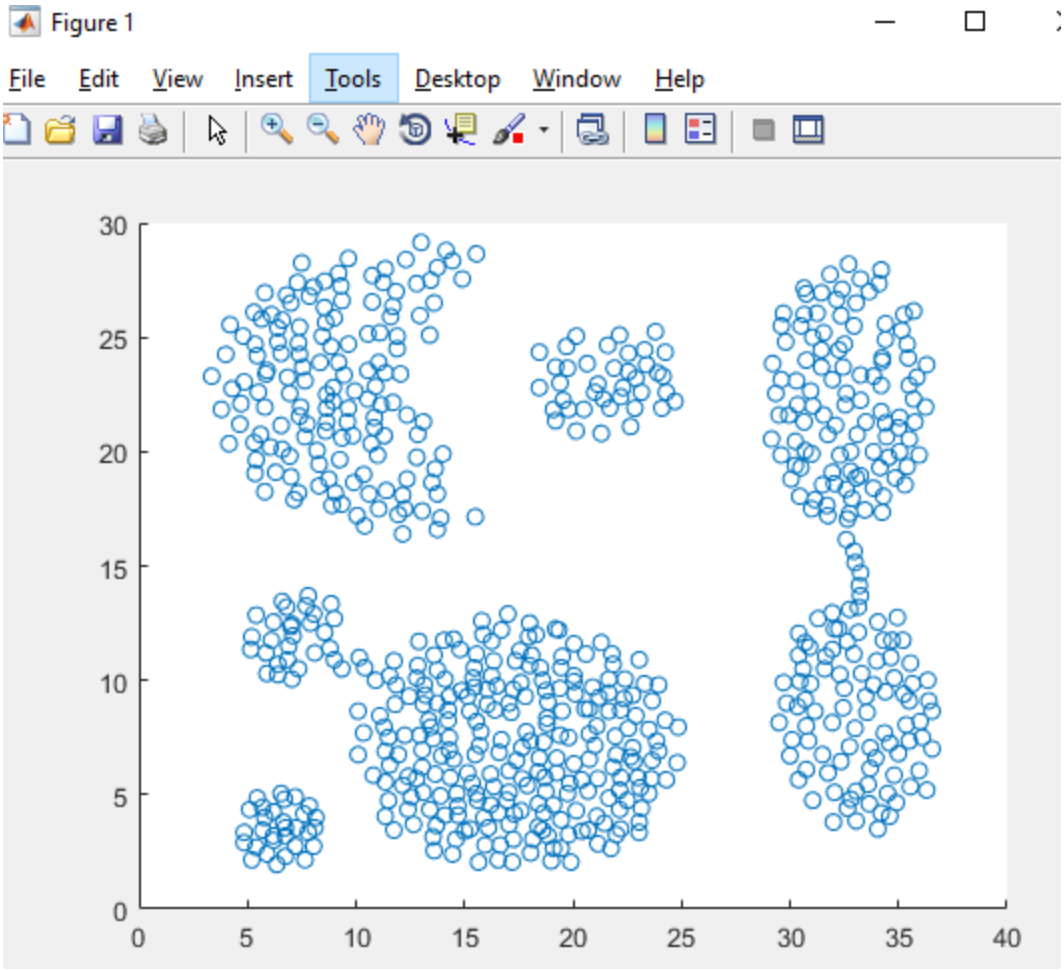
1. Pilih K buah titik *centroid* secara acak
2. Kelompokkan data sehingga terbentuk K buah *cluster* dengan titik *centroid* dari setiap *cluster* merupakan titik *centroid* yang telah dipilih sebelumnya
3. Perbaharui nilai titik *centroid*
4. Ulangi langkah 2 dan 3 sampai nilai dari titik *centroid* tidak lagi berubah

Langkah awal sebelum masuk ke dalam algoritma k means adalah import data dan memvisualisasikannya:

```
s file can be opened as a Live Script. For more information, see Creating Live Scripts.
    %% Import Data
    dataTrain = csvread('Tugas kMeans\TrainsetTugas2.csv');
    X = dataTrain(:,1:2);
    % Visualisasi data dengan scatter

    scatter(X(:,1),X(:,2));
```

Lalu outputannya adalah sebagai berikut:



Setelah itu membuat centroid secara acak sebanyak k.

```
%% Buat centroid random
[numRow , numCol] = size(X);
k = 7; %
startCentroids = zeros(k,2);
startCentroids = X(randperm(numRow, k), :);
```

Masukkan fungsi k-Means

```
%% Fungsi kMeans
% Definisi semua variabel yang dibutuhkan
[ numRow numCol ]    = size(X);
[ numK numkCol ]     = size(startCentroids);
change               = true;
i                    = 0;
tempCentroids        = startCentroids;
result               = X;
finalCentroids       = startCentroids;
listSSE              = [];
```

```
% Lakukan perulangan hingga centroid awal dan centroid akhir tidak berubah
    while (change )
        % Masukkan nilai centroid yang telah diupdate kedalam variabel
        % tempCentroids
        tempCentroids = finalCentroids;
        % Lakukan looping
        for i = 1:numRow
            % Set array kosong untuk menampung koordinat jarak minimum
            whoMin = [];
            % Lakukan looping untuk menghitung jarak tiap baris data
            % terhadap centroid awal
            for j = 1:numK
                sub = X(i,:) - finalCentroids(j,:);
                % Menghitung jarak menggunakan euclidean distance
                euclidean = sqrt(sum(sub.^2));
                whoMin = [whoMin; euclidean];
            end
            [~ , idx ] = min(whoMin);
            result(i,3) = idx;
        end
        % Lakukan perulangan untuk mengupdate centroid baru
        for i = 1:numK
            condition = result(:, 3) == i;
            finalCentroids(i,:) = mean(result(condition,1:2));
        end
```

Terdapat parameter input X yaitu data training dan startCentroids yang merupakan nilai acak dari centroids pada cluster yang ada. Cluster tersebut merupakan banyaknya k.

Setelah itu cari SSE nya untuk mencari centroid akhir yang paling optimum. Yaitu dengan cara menghitung menggunakan rumus Euclidean dari centroid terhadap call itu sendiri.

```
%% Hitung nilai sse untuk mendapatkan nilai K yang optimum
  % Assign array awal
  listSSE = [];
  % Lakukan perulangan dari centroid yang paling optimum
  for i = 1:length(finalCentroids)
    % Cari label yang sama pada array hasil (nb: label yang dimaksud adalah
    % cluster)
    condition = result(:, 3) == i;
    dataCondition = result(condition,1:2);

    % Lakukan perulangan untuk setiap setiap kluster (pake rumus yang ada
    % di slide buat ngitung ssenya)
    for j=1:length(dataCondition)
        sub = dataCondition(j,:) - finalCentroids(i,:);
        euclidean = sqrt(sum(sub.^2));
    end
    % hasilnya masukkan ke dalam variable listSSE
    listSSE = [listSSE ; euclidean];

  end
  % total semua nilai sse pada variabel sse
  resultSSE = sum(listSSE);

        % Lakukan pengecekan untuk nilai centroid yang telah diupdate
        % terhadap centroid awal. Jika nilai centroid akhir dan centroid
        % awal tidak berubah maka perulangan dihentikan dan program
        % diakhiri
        if ((tempCentroids == finalCentroids))
            change = false;
        end
    end
```

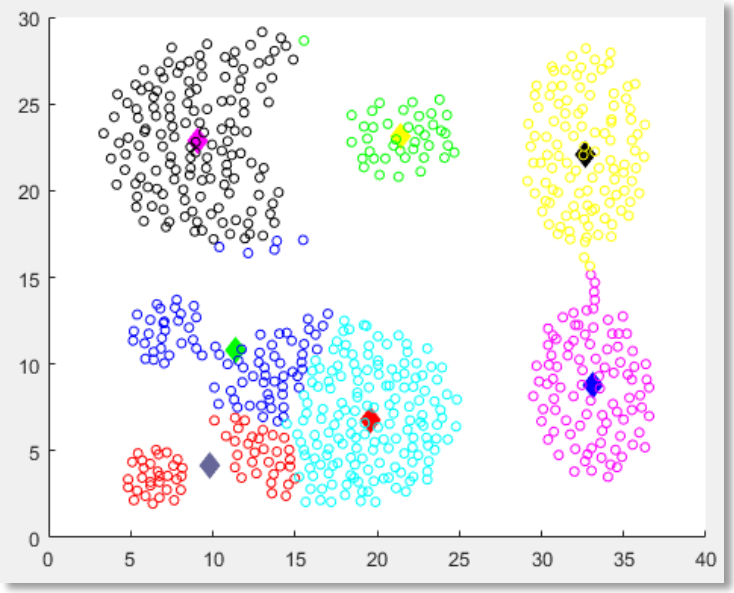
i

Lalu tinggal memvisualisasikan hasil akhirnya saja.

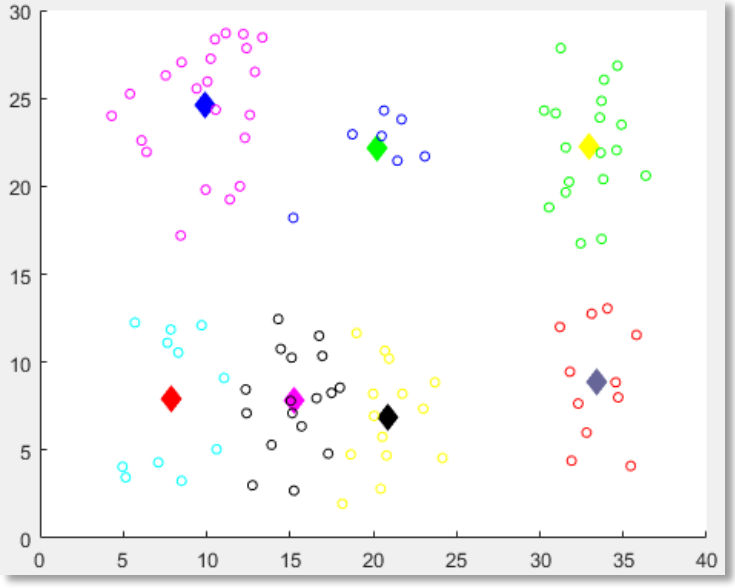
Evaluasi

Hasil dari program diatas yaitu didapatnya nilai k yang optimal untuk digunakan dalam menklasifikasi atau *clustering* data test. Dalam tugas ini saya membuat 2 script yaitu kMeansTestset.m dan kMeansTrainset.m. Data train set terdiri dari 688 data, data testset terdiri dari 100 data. Perbedaan lain dalam kMeansTestset dan kMeansTrainset adalah jika Trainset k nya ditentukan secara acak. Sedangkan pada Testset, k nya ditentukan dengan nilai mean atau rata-rata yang diperoleh dari k-Means pada Trainset.

Output:



Hasil k-Means DataTrain



Hasil k-means DataTest