2018

Tugas Program k-Means Clustering

Dosen Pengampu: Dr. Suyanto, S.T, M.Sc

Nama: Aziza Hayupratiwi

NIM: 1301150440

Kelas : IF 39-06

4/6/2018

1. Deskripsi Kasus dan Analisa Masalah

K-Means adalah metode klasterisasi atau penganalisaan yang melakukan proses pemodelan tanpa supervisi (unsupervised) dan merupakan salah satu metode yang melakukan pengelompokan data dengan sistem partisi. Metode k-means berusaha mengelompokkan data yang ada ke dalam beberapa kelompok, dimana data dalam satu kelompok mempunyai karakteristik yang sama satu sama lainnya dan mempunyai karakteristik yang berbeda dengan data yang ada di dalam kelompok yang lain. Dengan kata lain, metode ini berusaha untuk meminimalkan variasi antar data yang ada di dalam suatu cluster dan memaksimalkan variasi dengan data yang ada di cluster lainnya.

Data yang digunakan untuk data training ada sebanyak 688 data dengan dua atribut. Nantinya data tersebut akan di klaster agar mendapat nilai klasifikasinya berdasarnya nilai *Sum Square Error* (SSE) paling minimum. Hasil klaster dari data train akan diterapkan pada data test untuk mengetahui klasterisasi di data test.

2. Rancangan Metode

Untuk menyelesaikan masalah ini digunakan bahasa pemrograman matlab. Hal pertama yang perlu dilakukan adalah pada data train yang akan diklasterisasi, dipilih sejumlah *k* objek secara acak sebagai *centroid* awal. Pada program ini digunakan k=7 yang berasal dari metode Elbow yang mana pada titik siku penurunan nilai SSE terhadap jumlah klaster. Tahap kedua yaitu setiap objek yang bukan *centroid* dimasukkan ke klaster terdekat berdasarkan ukuran jarak tertentu. Ketiga, setiap *centroid* diperbarui berdasarkan rata-rata dari objek yang ada di dalam setiap klaster. Keempat, *looping* langkah kedua dan ketiga sampai semua *centroid* stabil atau konvergen, dalam artian semua *centroid* yang dihasilkan dalam iterasi saat ini sama dengan semua *centroid* yang dihasilkan pada iterasi sebelumnnya.

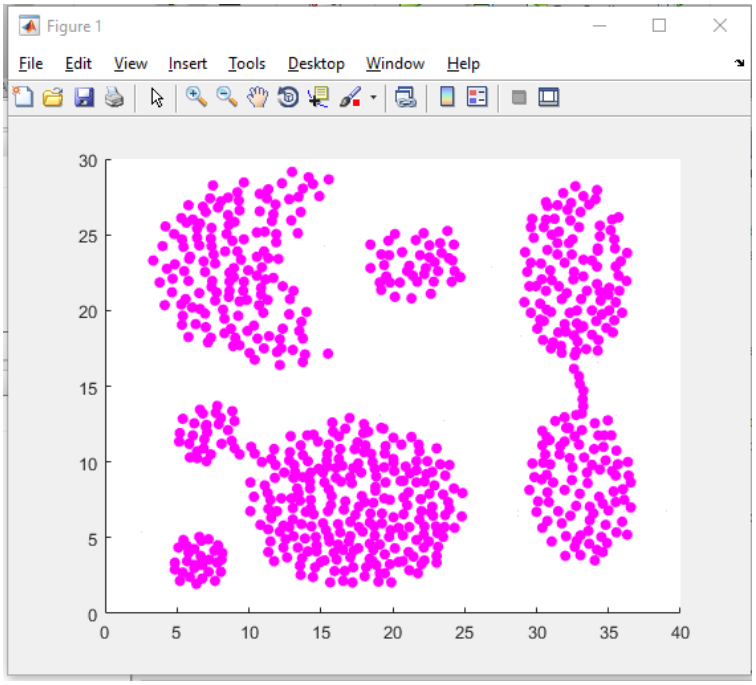
Setelah menemukan *centroid* yang stabil, maka akan disimpan dalam sebuah file bernama **fCentroid.txt** yang nantinya akan dipanggil ketika akan dilakukan pengujian pada data test. Lalu hitung jarak setiap data test ke *centroid* dan ambil nilai minimum atau nilai terdekatnya untuk menentukan klasternya.

3. Simulasi Metode

Tahap pertama yaitu membuka file data train dalam bentuk txt dan menampilkan scatter plot persebaran data di program bernama **kMeansCluster.m.**

```
7       %menampilkan persebaran data train
8 -     dataTrain = readtable('TrainsetTugas2.txt'); %membuka data dan disimpan pada var dataTrain
9 -     dTrain = table2array(dataTrain); %konversi dataTrain dalam bentuk matriks agar dapat dilakukan scatter
10 -    x = dTrain(:,1);
11 -    y = dTrain(:,2);
12 -    figure;
13 -    scatter(x,y,'m','filled'); %scatter pesebaran data dengan berwarna magenta
```

Gambar 1 Open file Data Train



Gambar 2 Pesebaran Data Train

Bangkitkan nilai acak dari data train sebanyak k=7 yang berasal dari metode Elbow yang mana pada titik siku penurunan nilai SSE terhadap jumlah klaster. Lalu panggil fungsi kMeans.

```
15      %membangkitkan centroid acak
16 -    randd = randperm(size(dTrain,1)); %membangkitkan random permutasi dari dTrain
17 -    centroid = dTrain(randd(1:7),1:2); %nilai random dTrain disimpan sebagai centroid
18 -    [centroid, cluster] = kMeans(dTrain(:,1:2), centroid); %memanggil fungsi kMeans
```

Gambar 3 centroid random

Fungsi kMeans berisi metode klasterisasi k-means yang mana akan dilakukan iterasi sampai titik *centroid* tidak berubah dengan menghitung jarak setiap objek yang bukan *centroid* dimasukkan ke klaster terdekat berdasarkan ukuran jarak tertentu. Lalu dicari jarak paling minimum. Hitung nilai SSE dengan membagi jumlah tiap data train ke *centroid* menggunakan Euclidean Distance dengan jumlah masing-masing klasternya. Setelah itu nilai *centroid* akan diubah didalam iterasi.

kMeansCluster.m | kMeans.m | sse.m | uji_kMeans.m | function_kMeansUji.m

```
1    %Aziza Hayupratiwi - 1301150440
2
3    function [centroid,cluster] = kMeans(data, centroid)
4        %membuat matriks dataxl cluster yang berisi 0
5        %jika cluster ditaruh pada main menu, matriks yg dihasilkan akan lxdata
6        cluster = zeros(size(data,1),1);
7        while 1, %looping hingga tidak ada perubahan centroid
8            for i = 1:size(data,1) %dari i=1 sampai banyaknya data (688)
9                for j = 1:size(centroid,1) %dari j=1 sampai banyaknya centroid (7)
10                   %menghitung jarak dengan euclidean distance
11                   euclid(j) = sqrt((centroid(j,1)-data(i,1))^2 + (centroid(j,2)-data(i,2))^2);
12               end
13               %menyimpan nilai absolute dari min euclid.
14               %tanda ~ bekerja dgn nilai min euclid sehingga nilai cluster(i)
15               %diabaikan
16               [~, cluster(i)] = min(euclid);
17           end
18           %memperbarui centroid dengan nilai min SSE
19           tmp_cen = centroid;
20           centroid = sse(data, cluster, centroid); %memanggil fungsi sse dan disimpan di centroid
21           %jika tmp_cen = centroid maka iterasi dihentikan
22           if tmp_cen == centroid
23               break;
24           end
25       end
26   end
```

Gambar 4 fungsi kMeans

kMeansCluster.m | kMeans.m | sse.m | uji_kMeans.m | function_kMeansUji.m

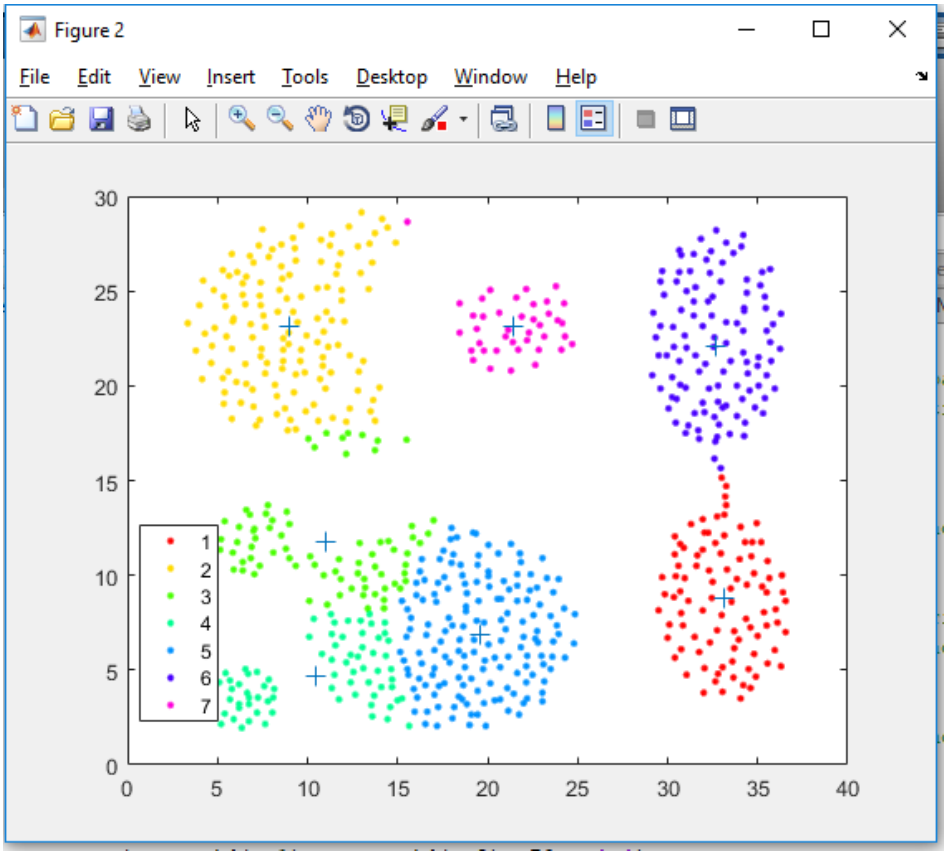
```
1    %Aziza Hayupratiwi - 1301150440
2
3    function [centroid] = sse(data, cluster, centroid)
4    %Sum of Squared Errors (SSE) untuk mencari centroid terbaru
5        for i=1:size(centroid,1) %dari i=1 sampai banyaknya centroid
6            dCentro = data(cluster==i,:); %centroid tiap klaster
7            centroid(i,:) = sum(dCentro)/size(dCentro,1); %jumlah jarak centroid ke data/banyaknya jumlah klaster
8        end
9    end
```

Gambar 5 function_kMeansUji

Kemudian akan menampilkan scatter data train beserta titik *centroid* nya yang ditandai dengan '+'.

```
20 -    figure;
21 -    gscatter(dTrain(:,1), dTrain(:,2), cluster);
22 -    hold on; %menahan scatter data train agar centroid dapat terlihat secara bersamaan
23 -    scatter(centroid(:,1), centroid(:,2), 70 , '+');
24 -    hold off;
```

Gambar 6 Scatter Centroid



Gambar 7 Scatter Centroid

Setelah dapat titik *centroid* terbaik, simpan ke dalam file **fCentroid.txt** yang nantinya akan dipanggil pada file pengujian uji **uji_kMeans.m**.

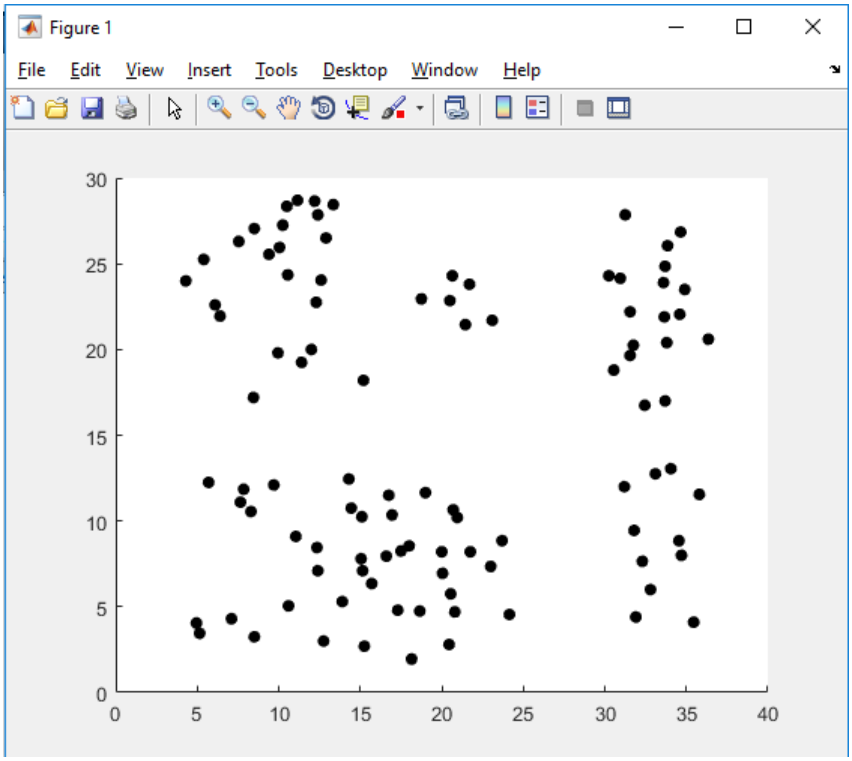
```
26      %menyimpan hasil centroid terbaik
27 -    dlmwrite('fCentroid.txt',centroid)
```

Gambar 8 Save File Centroid

Untuk data tes dijalankan di file **uji_kMeans.m**. File tersebut memanggil data test **TestsetTugas2.txt** dan menampilkan pesebaran data test dalam scatter plot.

```
%menampilkan persebaran data test
dataTest = readtable('TestsetTugas2.txt'); %membuka data dan disimpan pada var dataTest
dTest = table2array(dataTest); %konversi dataTest dalam bentuk matriks agar dapat dilakukan scatter
x = dTest(:,1);
y = dTest(:,2);
scatter(x,y,'k','filled'); %scatter pesebaran data dengan berwarna hitam
```

Gambar 9 Open File Data Test



Gambar 10 Pesebaran Data Test

Buka file centroid **fCentroid.txt** lalu panggil fungsi function_kMeansUji. Dalam function_kMeansUji tidak ada perubahan *centroid*, karena *centroid* terbaik dihasilkan dari data train. File *centroid* dibuka kemudian hitung jarang setiap data test ke *centroid* dengan Euclidean Distance, kemudian ambil jarak terdekat atau nilai minimum untuk menentukan klaster dari data test. Hasil klasterisasi data test akan disimpan dalam file **klasterDataTest.csv**.

```
dataCentroid = readtable('fCentroid.txt'); %membuka data centroid dan disimpan pada var dataCentroid
centroid = table2array(dataCentroid); %konversi dataCentroid dalam bentuk matriks agar dapat dilakukan scatter

[centroid, cluster] = function_kMeansUji(dTest(:,1:2), centroid); %memanggil fungsi kMeansUji

csvwrite('klasterDataTest.csv',cluster); %menyimpan hasil klaster pada file
```

Gambar 11 uji_kMeans

4. Kesimpulan

Metode *k-means clustering* dapat membuktikan bahwa adanya keterkaitan antar data yang hasilnya menunjukan bahwa adanya kemiripan data yang sejenis. Namun, *k-means clustering* hanya dapat digunakan pada data yang bernilai numerik, padahal nyatanya banyak data yang tidak bernilai numerik.

Saat membangkitkan titik *centroid*, *k-means clustering* mempunyai posibilitas tinggi untuk menemukan titik yang tepat untuk metode klaster. Namun karena *centroid* awal bernilai acak dan selalu berubah saat dijalankan, akan sangat sensitif terhadap hasil klasterisasi. Misalnya apabila titik *centroid* berada jauh dari titik pusat, maka hasil klasterisasi akan sangat tidak tepat.

Selain itu, permasalahan pada *k-means clustering* adalah model *clustering* yang berbeda-beda, pemilihan data train sebagai data acuan, kegagalan untuk *converge,* pendeteksian *outliers*, bentuk setiap *cluster*, dan *overlapping*.