KLASIFIKASI DATA MULTIDIMENSI MENGGUNAKAN SUBTRACTIVE CLUSTERING DAN K-NEAREST NEIGHTBOR

(*Classification Multidimension Data Using Subtractive Clustering and K-Nearest Neightbor*)

Nur Wakhidah
Fakultas Teknologi Informasi dan Komunikasi Universitas Semarang

Abstract

The process of clustering in multi-dimensional data can be carried out using subtractive clustering algorithm. In the process of clustering to determine the centroid, influenced by the value of the parameters such influence range, squash, accept_ratio and reject_ratio. And in classifying data, there are several techniques that can be used, one of which is the method of k-Nearest Neightbor that classify data with the predicted value of the neighborhood as the new query instance.

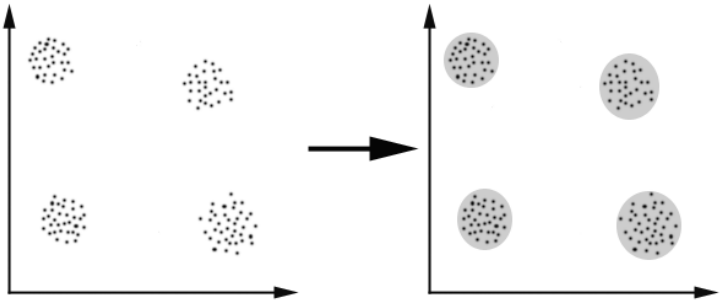
***Keyword** : classification, subtractive clustering, k-nearest neightbor*

1. PENDAHULUAN

Pattern Recognition merupakan sebuah disiplin ilmu yang bertujuan untuk mengklasifikasikan *object* berdasar image, berat atau parameter-parameter yang telah ditentukan ke dalam sejumlah kategori atau *class. Pattern recognition* meliputi berbagai aplikasi dan implementasi dalam kasus-kasus *real world*, misalnya *machine vision, character recognition (OCR) computer aided diagnosis, speech recognition, face recognition, biometrics, image database retrieval, data mining* dan *bioinformatics*.

Dalam *classification* terdapat 2 jenis algoritma yang pemilihnya tergantung pada kesediaan data awal, yaitu *supervised classification* dan *unsupervised classification*. Supervised adalah *pattern* yang mempunyai kelas yang telah diketahui dan digunakan untuk *training* (*cluster* klasifikasi yang sudah fix). Melakukan identifikasi suatu pola yang diamati sebagai pola yang sudah diketahui. Sedangkan *unsupervised* adalah sejumlah kelas tidak diketahui dan tidak terdapat *training pattern*. Memasukkan suatu pola yang diamati ke suatu kelas pola yang belum diketahui.

Clustering dapat dianggap yang paling penting dalam masalah *unsupervised learning,* karena setiap masalah semacam ini, ia berurusan dengan mencari *struktur* dalam kumpulan yang tidak diketahui datanya. Sehingga dapat didefinisikan bahwa *clustering* merupakan "proses mengatur objek menjadi anggota kelompok yang hampir sama dalam beberapa cara". Sebuah *cluster* merupakan kumpulan objek-objek yang "sama" di antara mereka dan "berbeda" pada objek dari cluster lainnya.



Gambar 1. Identifikasi Kelompok

Dengan memperhatikan gambar di atas, kita dengan mudah mengidentifikasikan 4 kelompok menjadi data yang dapat dibagi, yaitu kesamaan dengan kriteria jarak antara dua atau lebih benda dalam *cluster* yang sama jika mereka dekat dan sesuai dengan jarak yang diberikan. Hal ini disebut *distance-based clustering.* Lain halnya untuk jenis pengelompokan *konseptual clustering,* dimana dua atau lebih benda dalam *cluster* yang sama dengan mendefinisikan konsep secara umum untuk semua benda, dengan kata lain objek dikelompokkan menurut konsep deskriptif. Tujuan dari *clustering* adalah untuk mengklasifikasikan data, dengan cara menentukan pengelompokan dalam satu set data yang tidak diketahui.

2. PERMASALAHAN

Bagaimana melakukan pengelompokan dengan object data berupa numeric dengan jumlah data sebanyak 200 data dengan berdimensi 5 yang berasal dari 9289 data dengan 32 dimensi, kemudian setelah proses clustering terbentuk akan dilanjutkan proses classification dengan menguji data lain sejumlah 10 data dari data ke-201 sampai data ke-210 yang akan menjadi anggota dari clustering tersebut?

3. PEMBAHASAN

Pada pembahasan dalam proses *clustering* untuk data berdimensi banyak menggunakan *Subtractive Clustering*, sedangkan dalam proses *classification* menggunakan *k-Nearest Neightbor.* Dan untuk pengimplementasian algoritma S*ubtractive Clustering* dan *k-Nearest Neightbor* menggunakan pemrograman MATLAB.

3.1. SUBTRACTIVE CLUSTERING

Dalam mengelompokkan (*clustering process*), apabila jumlah cluster yang akan dibentuk belum diketahui, maka bisa digunakan *subtractive clustering*. *Subtractive clustering* didasarkan atas ukuran densitas (potensi) titik-titik data dalam suatu ruang (variabel).

Konsep dasar dari *subtractive clustering* adalah menentukan daerah-daerah dalam suatu variabel yang memiliki densitas tinggi terhadap titik-titik disekitarnya. Titik dengan jumlah tetangga terbanyak akan dipilih sebagai ***centroid***. Titik yang sudah terpilih sebagai *centroid* akan dikurangi densitasnya. Kemudian algoritma akan memilih titik lain yang memiliki tetangga terbanyak untuk dijadikan *centroid* yang lain. Hal ini akan dilakukan berulang-ulang hingga semua titik diuji. Apabila terdapat N buah data: u1, u2, ..., uN dan dengan menganggap bahwa data-data tersebut sudah dalam keadaan normal, maka densitas titik u_k (D_k) dapat dihitung sebagai:

$$D_k = \sum_{j=1}^{N} \exp\left(-\frac{\left\| u_k - u_j \right\|}{\left(r_a / 2 \right)^2} \right)$$

Setelah menghitung densitas tiat-tiap titik, maka titik dengan densitas tertinggi akan dipilih sebagai *centroid*. Misalkan u_{c1} adalah titik yang terpilih sebagai *centroid*, sedangkan D_{c1} adalah ukuran densitasnya. Selanjutnya densitas dari titik-titik di sekitarnya akan dikurangi menjadi:

$$D_k' = D_k - D_{c1} \times \exp\left(-\frac{\left\| u_k - u_j \right\|}{\left(r_b / 2 \right)^2} \right)$$

Nilai r_b menunjukkan suatu lingkungan yang mengakibatkan titik-titik berkurang ukuran densitasnya. Biasanya r_b bernilai lebih besar dibanding dengan r_a, r_b = *squash_factor*r_a (biasanya *squash_factor* = 1,5). Hal ini berarti bahwa titik-titik yang berada dekat dengan centroid u_{c1} akan mengalami pengurangan densitas besar-besaran, dan berakibat titik-titik tersebut akan sangat sulit untuk menjadi *centroid* berikutnya.

Pada implementasinya, bisa digunakan 2 pecahan sebagai faktor pembanding, yaitu ***accept ratio*** dan ***reject ratio***. Apabila hasil bagi antara potensi tertinggi suatu titik data dengan potensi tertinggi pertama kali yang diperoleh pada iterasi pertama lebih besar daripada *accept ratio*, maka titik data tersebut diterima sebagai *centroid* baru. Apabila hasil bagi antara potensi tertinggi suatu titik data dengan potensi tertinggi pertama kali yang

diperoleh pada iterasi pertama lebih kecil daripada *accept ratio* namun lebih besar daripada *reject ratio*, maka titik data tersebut baru akan diterima sebagai *centroid* baru hanya jika titik tersebut terletak pada jarak yang cukup jauh dengan *centroid* yang lainnya. Namun, apabila hasil bagi antara potensi tertinggi suatu titik data dengan potensi tertinggi pertama kali yang diperoleh pada iterasi pertama lebih kecil daripada *accept ratio* maupun *reject ratio*, maka titik data tersebut sudah tidak akan dipertimbangkan lagi untuk menjadi *centroid* baru (potensinya sama dengan nol).

ALGORITMA *SUBTRACTIVE CLUSTERING*:

1. Tetapkan nilai:

- ***r*** adalah jari-jari, merupakan vektor yang akan menentukan seberapa besar pengaruh *centroid* pada tiap-tiap dimensi data
- ***q*** adalah faktor pengali ke jari-jari yang akan menentukan kedekatan suatu *centroid* yang mana keberadaannya terhadap *centroid* yang lainnya akan dikurangi
- ***accept_ratio*** merupakan bilangan pecahan yang menunjukkan potensi terhadap *centroid* pertama, jika potensi lebih besar dari *accept_ratio*, maka keberadaan titik tersebut akan diterima sebagai *centroid* baru
- ***reject_ratio*** merupakan bilangan pecahan yang menunjukkan potensi terhadap *centroid* pertama, jika potensi lebih kecil dari *reject_ratio*, maka titik tersebut akan diabaikan untuk dipertimbangkan menjadi *centroid* baru selanjutnya

2. Normalisasi:

- $Xmin_j$ = min[x_{ij}|j =1,2,…, m];
- $Xmax_j$ = max[x_{ij}|j =1,2,…, m];
- Hitung:

$$x_{ij} = \frac{x_{ij} - X\min_j}{X\max_i - X\min_j} \quad ; \quad i = 1,2,...,n; \quad j = 1,2,...,m$$

3. Tentukan potensi awal:

- i = 1
- Kerjakan hingga i=n,
  - $T_{ij} = x_{ij}$; j=1,2,…,m
  - $D_{kj} = (x_{kj} - T_{ij})/r_j$; j=1,2,…,m; k=1,1,…,n
  - Hitung:

$$P_i = \sum_{j=1}^{m} e^{-\left[4\sum_{k=1}^{n}\left(D_{kj}\right)^2\right]}$$

4. Cari titik dengan potensi tertinggi:

- M = max[Pi|i=1,2,…,n];
- h = i, sedemikian hingga Pi = M;

5. Tentukan *centroid* dan kurangi potensinya terhadap titik-titik disekitarnya.

- M = max[Pi|i=1,2,…,n];
- Cn = 0 (jumlah *cluster*);
- Z = M;
- Hitung:

$$Rasio = \frac{\|Z\|}{\|M\|}$$

- Key= 1 (ada titik yang bisa dipertimbangkan sebagai *centroid* baru)
- Kerjakan jika (Key $\neq$ 0) atau (tidak ada elemen Z = 0)
  - Jika Rasio > *accept_ratio*, maka Key=1;
  - Jika *reject_ratio* < Rasio $\in$ *accept_ratio*, maka kerjakan
    - MD = -1;
    - Kerjakan untuk i=1 sampai i=C:
      - $G_j = V_j - C_{ij}$; j=1,2,…,m
      - Hitung:

$$Sd = \sum_{j=1}^{m}\left(G_j\right)^2$$

      - Jika (MD < 0) atau (Sd < MD), maka MD = Sd;
      - Jika (Rasio + $\sqrt{Sd}$) $\geq$ 1, maka (Key = 1), artinya titik data diterima sebagai *centroid*
      - Jika (Rasio + $\sqrt{Sd}$) < 1, maka (Key = 2), artinya titik data tidak akan dipertimbangkan kembali sebagai *centroid*
    - Jika Key=1, kerjakan:
      - Cn = Cn+1;

i=1,2,…, n

i=1,2,…, n

- CC = Z;
- Kurangi potensi dari titik-titik di dekat *clentroid*:

▪ Hitung:

$$S_{ij} = \frac{V_{hj} - x_{ij}}{r_j \times q}; \quad j = 1,2,...,m; \quad i = 1,2,...,n$$

▪ Hitung:

$$D_i = M \times e^{-\left[4\sum_{k=1}^{n}\left(D_{kj}\right)^2\right]}; \quad i = 1,2,...,n$$

▪ P = P – D;
▪ Jika Pi ∈ 0, maka Pi = 0;
▪ Z = max[Pi|i=1,2,…,n];
▪ h = i, sedemikian hingga Pi = Z;

- Jika Key=2, maka Ph = 0;

6. Kembalikan *centroid* dari bentuk ternormalisasi ke bentuk semula.

$$C_{ij} = C_{ij} \times \left(X\max_j - X\min_j\right) + X\min_j$$

7. Hitung nilai sigma *cluster*.

$$S_j = r_j \times \frac{\left(X\max_j - X\min_j\right)}{\sqrt{8}}$$

3.2. k-NEAREST NEIGHTBOR

k-Nearest Neightbor (KNN) adalah suatu metode yang menggunakan algoritma *supervised* dimana hasil dari *query instance* yang baru diklasifikasikan berdasarkan mayoritas dari kategori pada KNN. Tujuan dari algoritma ini adalah mengklasifikasi objek baru berdasakan atribut dan *training sample*. *Classifier* tidak menggunakan model apapun untuk dicocokkan dan hanya berdasarkan pada memori. Diberikan titik *query*, akan ditemukan sejumlah K objek atau (titik *training*) yang paling dekat dengan titik *query*. Klasifikasi menggunakan *voting* terbanyak di antara klasifikasi dari K objek. Algoritma KNN menggunakan klasifikasi ketetanggaan sebagai nilai prediksi dari query *instance* yang baru.

Algoritma metode KNN sangatlah sederhana, bekerja dengan berdasarkan pada jarak terpendek dari *query instance* ke *training sample* untuk menentukan KNN nya. Setelah mengumpulkan KNN, kemudian diambil mayoritas dari KNN untuk dijadikan prediksi dari *query instance*. Data untuk algoritma KNN terdiri dari beberapa atribut *multi-variate* Xi yang akan digunakan untuk mengklasifikasikan Y. Data dari KNN dapat dalam skala ukuran apapun, dari ordinal ke nominal.

KNN memiliki beberapa kelebihan yaitu bahwa dia tangguh terhadap training data yang *noisy* dan efektif apabila *training data*-nya besar. Sedangkan kelemahan dari KNN adalah KNN perlu menentukan nilai dari parameter K (jumlah dari tetangga terdekat), pembelajaran berdasarkan jarak tidak jelas mengenai jenis jarak apa yang harus digunakan dan atribut mana yang harus digunakan untuk mendapatkan hasil yang terbaik, dan biaya komputasi cukup tinggi karena diperlukan perhitungan jarak dari tiap *query instance* pada keseluruhan *training sample.*

3.3. PEMROGRAMAN MATLAB

MATLAB (*Matrix Laboratory*) adalah sebuah program untuk analisis dan komputasi numerik, merupakan suatu bahasa pemrograman matematika lanjutan yang dibentuk dengan dasar pemikiran menggunakan sifat dan bentuk matriks.

MATLAB merupakan software yang dikembangkan oleh Mathworks, Inc. dan merupakan software yang paling efisien untuk perhitungan numerik berbasis matriks. Dengan demikian jika di dalam perhitungan kita dapat memformulasikan masalah ke dalam format matriks, maka MATLAB merupakan software terbaik untuk penyelesaian numeriknya.

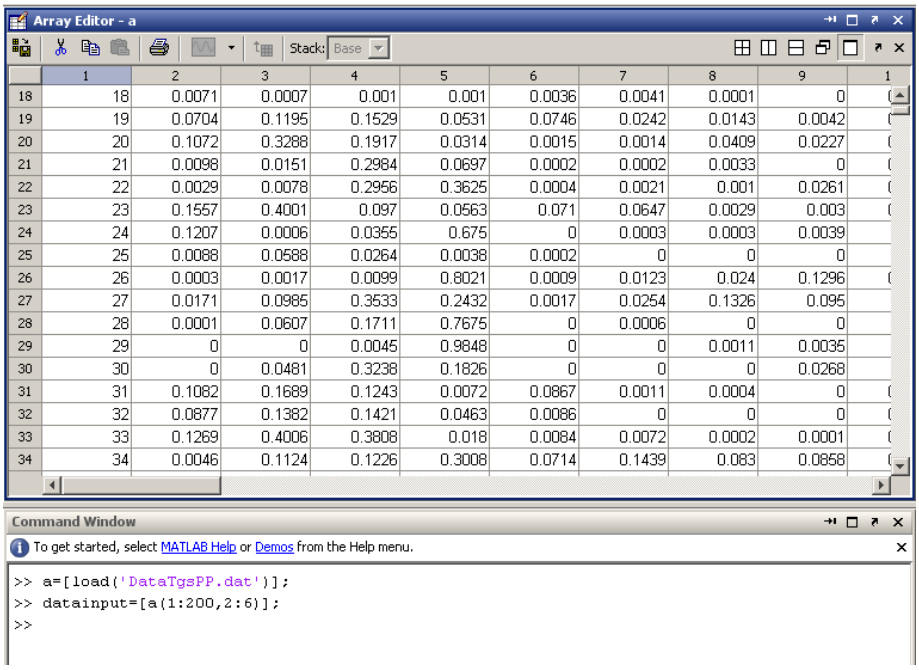
MATLAB yang merupakan bahasa pemrograman tingkat tinggi berbasis pada matriks sering digunakan untuk teknik komputasi numerik, digunakan untuk menyelesaikan masalah-masalah yang melibatkan operasi matematika elemen,

matrik, optimasi, aproksimasi, dan lain-lain. MATLAB banyak digunakan pada :

- Matematika dan Komputasi
- Pengembangan dan Algorithma
- Pemrograman modelig, simulasi dan pembuatan prototipe
- Analisa data, eksplorasi dan visualisasi
- Analisa numerik dan statistik
- Pengembangan aplikasi teknik

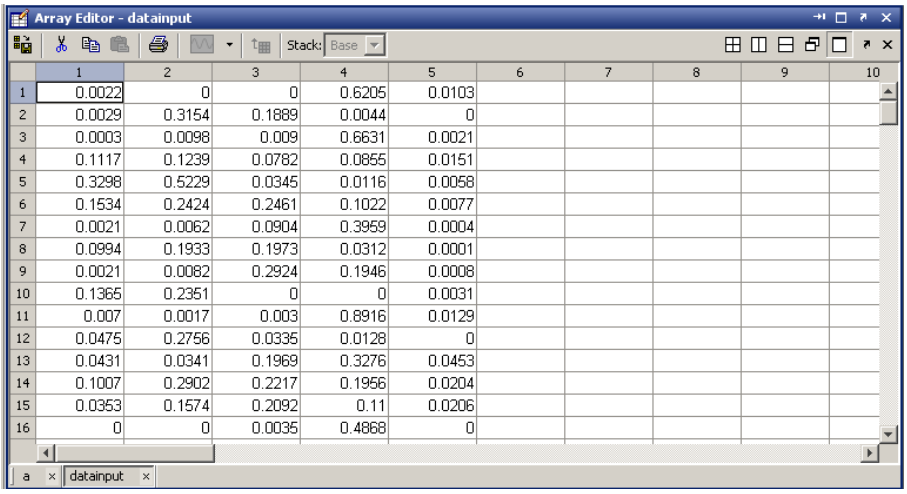
4. IMPLEMENTASI PROGRAM

Dalam implementasi program untuk menyelesaikan permasalahan diatas menggunakan pemrograman MATLAB dengan algoritma *Subtractive Clustering* dan teknik *k-NN Classify*. Dari sumber data (object) yang diperoleh berupa data numeric sebanyak 9289 baris dan 32 dimensi (matrik 9289x32) yang tersimpan pada file X.doc, maka untuk mengklasifikasikan data numeric pada 200 baris dengan 5 dimensi (matrik 200x5) perlu dilakukan pemotongan data, namun untuk melakukan pemotongan pada pemrograman MATLAB perlu perubahan ekstensi file terlebih dahulu dari X.doc ke X.dat melalui aplikasi Notepad. Untuk cara pemotongan data adalah file X.dat di-load dan disimpan dalam sebuah variable (yaitu a) dengan perintah a=[load('X.dat')];, kemudian dilakukan pemotongan data menjadi data sejumlah matrik 200x5 yang tersimpan dalam variable datainput dengan perintah datainput = [a(1:200,2:6)];, untuk lebih jelasnya dapat dilihat pada gambar berikut:



	1	2	3	4	5	6	7	8	9	1
18	18	0.0071	0.0007	0.001	0.001	0.0036	0.0041	0.0001	0	
19	19	0.0704	0.1195	0.1529	0.0531	0.0746	0.0242	0.0143	0.0042	
20	20	0.1072	0.3288	0.1917	0.0314	0.0015	0.0014	0.0409	0.0227	
21	21	0.0098	0.0151	0.2984	0.0697	0.0002	0.0002	0.0033	0	
22	22	0.0029	0.0078	0.2956	0.3625	0.0004	0.0021	0.001	0.0261	
23	23	0.1557	0.4001	0.097	0.0563	0.071	0.0647	0.0029	0.003	
24	24	0.1207	0.0006	0.0355	0.675	0	0.0003	0.0003	0.0039	
25	25	0.0088	0.0588	0.0264	0.0038	0.0002	0	0	0	
26	26	0.0003	0.0017	0.0099	0.8021	0.0009	0.0123	0.024	0.1296	
27	27	0.0171	0.0985	0.3533	0.2432	0.0017	0.0254	0.1326	0.095	
28	28	0.0001	0.0607	0.1711	0.7675	0	0.0006	0	0	
29	29	0	0	0.0045	0.9848	0	0	0.0011	0.0035	
30	30	0	0.0481	0.3238	0.1826	0	0	0	0.0268	
31	31	0.1082	0.1689	0.1243	0.0072	0.0867	0.0011	0.0004	0	
32	32	0.0877	0.1382	0.1421	0.0463	0.0086	0	0	0	
33	33	0.1269	0.4006	0.3808	0.018	0.0084	0.0072	0.0002	0.0001	
34	34	0.0046	0.1124	0.1226	0.3008	0.0714	0.1439	0.083	0.0858	

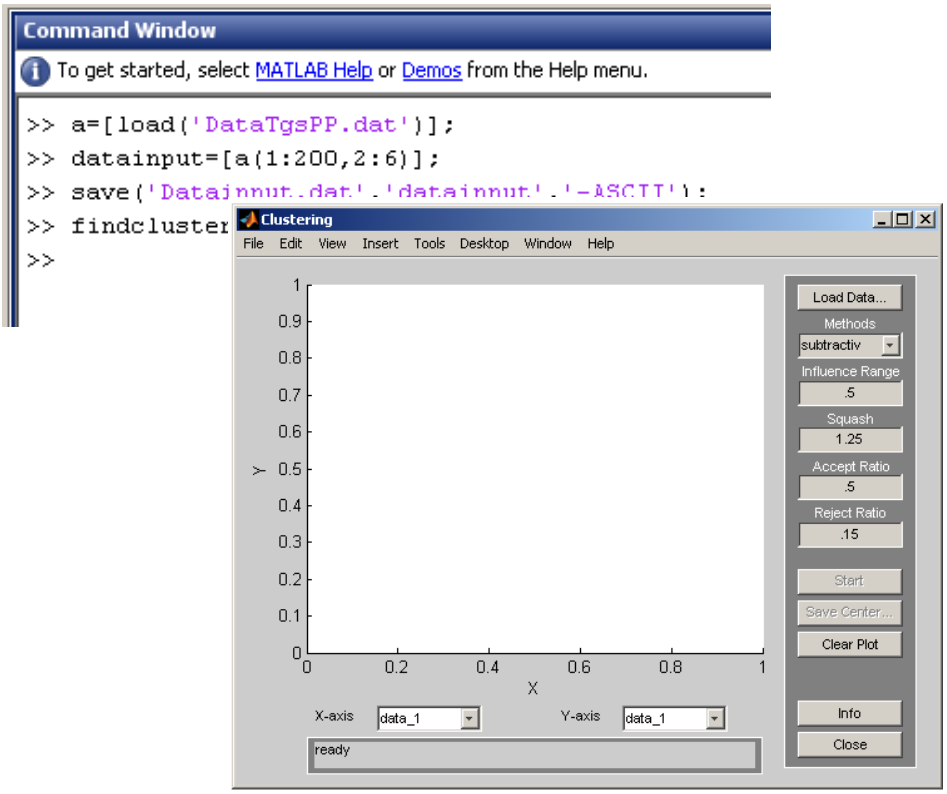
Gambar 2. Pemotongan Data



	1	2	3	4	5	6	7	8	9	10
1	0.0022	0	0	0.6205	0.0103					
2	0.0029	0.3154	0.1889	0.0044	0					
3	0.0003	0.0098	0.009	0.6631	0.0021					
4	0.1117	0.1239	0.0782	0.0855	0.0151					
5	0.3298	0.5229	0.0345	0.0116	0.0058					
6	0.1534	0.2424	0.2461	0.1022	0.0077					
7	0.0021	0.0062	0.0904	0.3959	0.0004					
8	0.0994	0.1933	0.1973	0.0312	0.0001					
9	0.0021	0.0082	0.2924	0.1946	0.0008					
10	0.1365	0.2351	0	0	0.0031					
11	0.007	0.0017	0.003	0.8916	0.0129					
12	0.0475	0.2756	0.0335	0.0128	0					
13	0.0431	0.0341	0.1969	0.3276	0.0453					
14	0.1007	0.2902	0.2217	0.1956	0.0204					
15	0.0353	0.1574	0.2092	0.11	0.0206					
16	0	0	0.0035	0.4868	0					

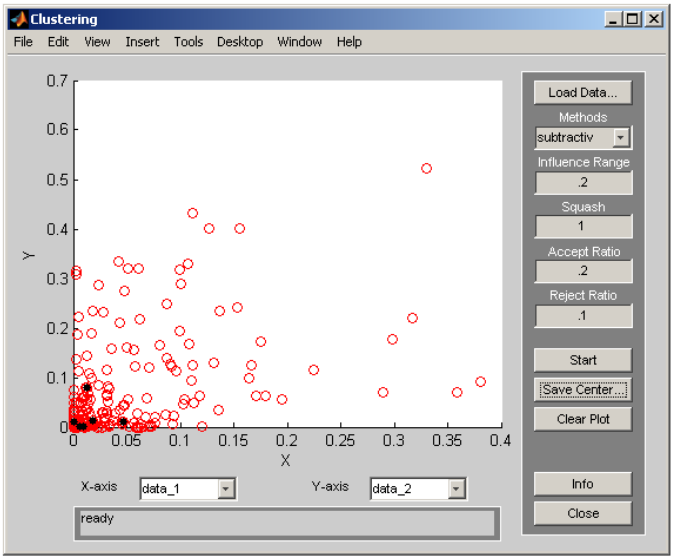
Gambar 3. Hasil Pemotongan Data

Setelah sumber data telah terpotong, data yang diperoleh disimpan pada Datainput.dat dengan perintah save('Datainput.dat','datainput','-ASCII');. kemudian untuk proses clustering pada data input dilakukan pemanggilan fungsi findcluster sehingga muncul jendela *clustering*, seperti gambar berikut.



Gambar 4. Pemanggilan Fungsi Findcluster

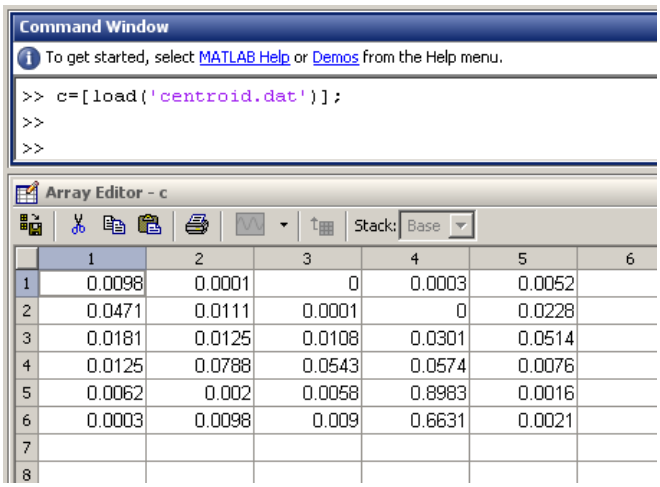
Kemudian lakukan pengoperasian pada fungsi ini, dimana pada Methods telah disediakan algoritma *Subtractive Clustering*. Untuk mengelompokkan data input sejumlah matrik 200x5 (Datainput.dat) dapat diambil dari button Load Data, kemudian diberikan nilai pada influence range sebesar 0.2, squash sebesar 1, accept ratio sebesar 0.2 dan reject ratio sebesar 0.1. Untuk mengetahui hasil dari clustering dapat diperoleh dari button Start. Hasil dari *clustering* adalah berupa *centroid* yang dapat disimpan melalui button Save Center dengan nama Centroid.dat. Sehingga pengelompokan data pada data sejumlah matrik 200x5 telah diperoleh.



Gambar 5. *Centroid*

Dari *centroid* yang telah diperoleh, yaitu 6 cluster, dapat dilakukan klasifikasi untuk data test sejumlah matrik 10x5. Data test diperoleh dengan pemotongan data dari sumber data (variable a) mulai data ke 201 – 210 berdimensi 5 dengan cara yang sama pada pemotongan data sebelumnya, dan tersimpan pada variable datatest dengan perintah datatest=[a(201:210,2:6)];.

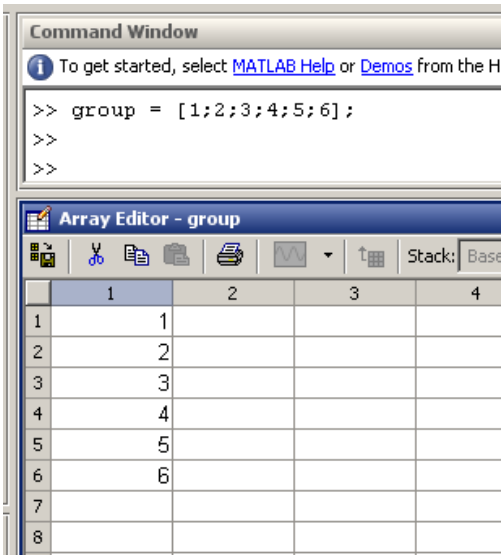
Dalam mengklasifikasi 10 data tersebut dapat digunakan teknik k-Nearest Neightbor dengan cara *centroid* yang telah tersimpan di-load dengan perintah c=[load('centroid.dat')];,



	1	2	3	4	5	6
1	0.0098	0.0001	0	0.0003	0.0052	
2	0.0471	0.0111	0.0001	0	0.0228	
3	0.0181	0.0125	0.0108	0.0301	0.0514	
4	0.0125	0.0788	0.0543	0.0574	0.0076	
5	0.0062	0.002	0.0058	0.8983	0.0016	
6	0.0003	0.0098	0.009	0.6631	0.0021	
7						
8						

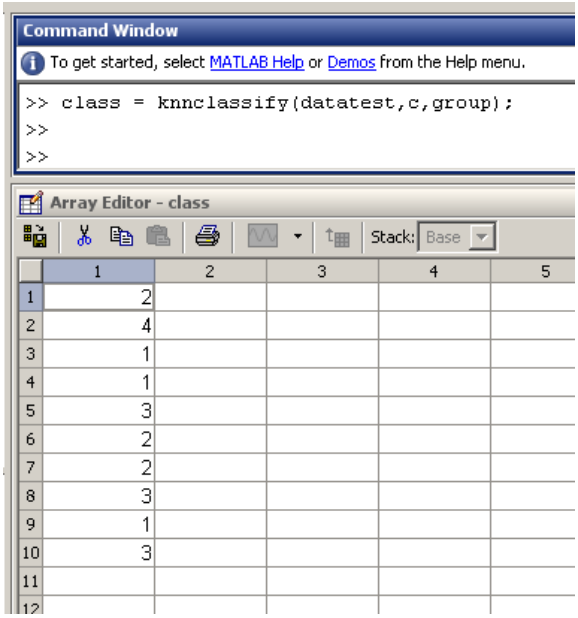
Gambar 6. Data *Centroid*

Kemudian dilakukan pelabelan untuk 6 *centroid* tersebut dengan perintah group = [1;2;3;4;5;6];



Gambar 7. Pelabelan (Group)

Untuk mengklasifikasikan 10 data tersebut digunakan perintah class = knnclassify(datatest,c,group); sehingga diperoleh hasil pengklasifikasian 10 data tersebut yang tersimpan pada variable class. Hasilnya adalah [2;4;1;1;3;2;2;3;1;3]. Untuk lebih jelasnya dapat dilihat pada gambar berikut.



Gambar 8. Klasifikasi Data Test

5. KESIMPULAN

Dari impelementasi diatas, dapat ditarik kesimpulan sebagai berikut:

1. Proses pengelompokan data (*clustering*) pada data multi dimensi dapat dilakukan dengan menggunakan algoritma *Subtractive Clustering*.
2. Dalam proses pengelompokan data (*clustering*) untuk menentukan pusat *cluster (centroid*), dipengaruhi oleh besarnya nilai parameter yang

diantaranya *influence range, squash, accept_ratio* dan *reject_ratio.*

3. Kelemahan algoritma *Subtractive Clustering* adalah tidak bisa melihat anggota setiap dimensi dari clustering yang terbentuk secara visual hanya dapat melihat pusat cluster (*centroid*) saja.
4. Dalam mengklasifikasikan data terdapat beberapa teknik yang dapat digunakan, salah satunya adalah metode *k-Nearest Neightbor* yang mengklasifikasikan data dengan ketetanggaan sebagai nilai prediksi dari *query instance* yang baru.
5. Kelebihan KNN adalah ketangguhannya terhadap *training* data yang *noisy* dan efektif kalo nilai *training* datanya besar. Kelemahannya adalah KNN perlu menentukan nilai dari parameter K (jumlah dari tetangga terdekat), pembelajaran berdasarkan jarak tidak jelas mengenai jenis jarak apa yang harus digunakan dan atribut mana yang harus digunakan untuk mendapatkan hasil yang terbaik, dan biaya komputasi cukup tinggi karena diperlukan perhitungan jarak dari tiap *query instance* pada keseluruhan *training sample.*
6. MATLAB yang merupakan bahasa pemrograman tingkat tinggi berbasis pada matriks sering digunakan untuk teknik komputasi numerik, digunakan untuk menyelesaikan masalah-masalah yang melibatkan operasi matematika elemen, matrik, optimasi, aproksimasi, dan lain-lain. Serta tersedianya toolbox pada MATLAB yang dapat membantu untuk mengimplementasikan kedua teknik tersebut, yaitu *Subtractive Clustering* dan *k-Nearest Neightbor*.

DAFTAR PUSTAKA

E.S. Gopi, "*Algorithm Collections for Digital Signal Processing Applications Using Matlab*", Spinger: National Institute of Technology, Tiruchi, India,

http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/

home.dei.polimi.it/matteucc/Clustering/tutorial_html (clusteringa)

yudiagusta.wordpress.com/clustering/

http://en.wikipedia.org/wiki/knn_algorithm

Riza Ramadan, Jurnal : Penerapan Pohon Untuk Klasifikasi Dokumen Teks Berbahasa Inggris

Arhami M, Desiani A, 2005, Pemrograman MATLAB, Andi, Yogyakarta.