Metode Peng*cluster*an Berbasis Densitas Menggunakan Algoritma DBSCAN

Methods of Density-Based Clustering Algorithm using DBSCAN

[1]Nur Arsih, [2]Nusar Hajarisman, [3]Sutawanir Darwis

[1,2,3]Prodi Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Bandung, Jl. Tamansari No.1 Bandung 40116

email: [1]nur_a65@yahoo.co.id

Abstract. Cluster analysis (clustering) is one of the statistical technique used to classify a range of data with high similarity in comparison to one another, but they are different from the objects in other groups. Clustering generally classified into hierarchical and non-hierarchical algorithm later evolved into many of them DBSCAN. DBSCAN is one algorithm which classifies the object based on the density of the input parameters Eps and MinPts. In this paper DBSCAN method will be compare with the k-means. The data used is secondary data of customers that having a credit facility. The results show that with Eps = 0.0128 and MinPts = 5 debtor data is divided into two clusters, that is good credit cluster and bad credit cluster with a run time faster than classical k-means algorithm.

Keywords: Cluster Analysis, DBSCAN, Density, Eps, K-means, Min Pts, Noise.

Abstrak. Analisis cluster (clustering)merupakan salah satu teknik statistika yang digunakan untuk mengelompokkan suatu gugus data dengan kemiripan yang tinggi dibandingkan satu sama lain, tetapi mereka berbeda dengan objek di lain kelompok. Umumnya clustering diklasifikasi menjadi hierarki dan non hierarki yang kemudian berkembang menjadi banyak algoritma diantaranya DBSCAN. DBSCAN merupakan salah satu algoritma yang mengelompokkan objek berdasarkan densitas dengan parameter input Eps dan MinPts. Dalam skripsi ini metode DBSCAN akan dibandingkan dengan k-means. Adapun data yang digunakan adalah data sekunder nasabah yang memiliki fasilitas kredit. Hasilnya menunjukkan bahwa dengan Eps=0,0128dan MinPts=5 data debitur terbagi menjadi dua cluster yaitu cluster kredit baik dan buruk dengan run time lebih cepat dibanding algoritma klasik k-means.

Kata Kunci: Analisis *Cluster*, DBSCAN, Densitas, *Eps, K-means, Min Pts, Noise.*

A. Pendahuluan

Dalam lingkup statistik multivariat salah satu tindakan yang bisa diambil dalam hal mengelompokkan suatu gugus data adalah dengan menggunakan metode *clustering*. Metodeini bertujuan mempelajari kemiripan sampel yang berbeda dengan menggunakan prinsip kerja bahwa suatu gugus objek dikelompokkanke dalam *cluster-cluster* sehingga objek dalam sebuah *cluster* tersebut memiliki kemiripan yang tinggi dibandingkan satu sama lain, tetapi mereka berbeda atau tidak mirip dengan objek di*cluster* yang lain (Yan dan Deng, 2015).

Dalam praktiknya terutama di dunia kerja, pemanfaatan *clustering* tidak terbatas dalam lingkup statistika saja. Secara luas metode ini bisa diterapkan dalam kasus pengenalan pola, data mining, pembelajaran mesin, aplikasi riset pasar ilmu ekonomi, dan aplikasi web yang meliputi klasifikasi dokumen dan *weblog cluster*. Oleh karena itu pengaplikasian metode *clustering* ini terus dikembangkan.

Contoh *real* dari pemanfaatan teknik *clustering* misalnyadalam bidang perbankan adalah dimana ketika kreditur (Bank) menerima permohonan pinjaman dari debitur (nasabah), maka Bank harus mempunyai pertimbangan tersendiri untuk mengambil keputusan apakah permohonan kredit tersebut disetujui atau tidak, sehingga Bank dapat meminimalisasi risiko. Pengambilan keputusan tersebut bisa diputuskan dengan cara meng*clustering* nasabah menjadi beberapa kategori *cluster* seperti kategori kredit baik dan kredit buruk.

Ada banyak jenis *clustering* yang sering digunakan, dua diantaranya diklasifikasi menjadi *clustering* hirarki dan *clustering* non hirarki. Berdasar dari algoritma-algoritma yang sudah ada beberapa diantaranya merupakan algoritma yang telah mengalami perkembangan seperti algoritma DBSCAN, DBCLASD, dan DENCLUE.

Dari beberapa algoritma tersebut masing-masing memiliki keunggulan dan kelemahan tersendiri seperti metode *clustering* hirarki yang tidak perlu menentukan jumlah *cluster* yang diinginkan karena proses dapat langsung dihentikan pada saat jumlah *cluster* sesuai namun *clustering* jenis ini hanya menemukan *cluster* dengan menggabungkanpasangan*cluster* secara iteratifsampai banyaknya *cluster* yang diinginkan diperoleh (Guha, dkk., 2001). Sedangkan metode nonhirarki partisi seperti *K-means* unggul dalam mengelompokkan data dengan ukuran besar dan umum digunakan tetapi mempunyai kelemahan yang diakibatkan oleh penentuan pusat awal *cluster*. Hasil *cluster* yang terbentuk dari metode *K-means* ini sangat tergantung pada inisiasi nilai pusat awal *cluster* yang diberikan (Santosa, 2007). Selain itu partisi *k-means* hanya menemukan *cluster* yang terbentuk saja tanpa memperhatikan ukuran *cluster* dan densitasnya.

Algoritma partisi lainnya seperti DBSCAN (Ester, dkk., 1996) melakukan *clustering* dengan memperhatikan konektivitas dari densitas (Nagpal dan Mann, 2011). DBSCAN merupakan algoritma dengan tujuan memisahkan sampel densitas tinggi dengan sampel densitas rendah. Algoritma ini unggul dengan kemampuannya dalam mendeteksi *outlier/noise* namun *cluster* yang terbentuk tergantung dengan nilai input yang diberikan.

Dari sekian banyak algoritma *clustering* yang telah berkembang tentunya menjadi kerumitan tersendiri bagi seorang pengguna untuk menentukan algoritma mana yang lebih tepat digunakan. Oleh karena itu beberapa algoritma yang telah dibahas sebelumnya yaitu DBSCAN dibandingkan dengan*k-means* akan ditinjau berdasarkan beberapa kriteria yang akan menjadi patokan pertimbangan untuk mengetahui keefektifan dalam melakukan *clustering* terhadap data nasabah yang

memiliki fasilitas kredit (debitur). Kriteria tersebut yakni kompleksitas, bentuk cluster, parameter masukan, penanganan *noise*, dan *run time*-nya (Nagpal dan Mann (2011).

Berdasarkan latar belakang yang telah diuraikan, maka perumusan masalah dalam penelitian ini sebagai berikut:

1. Bagaimana algoritma *clustering* berdasarkan metode DBSCAN?
2. Bagaimana algoritma *clustering* berdasarkan metode *k-means*?
3. Bagaimana perbandingan final hasil *clustering* antara metode berdasarkan densitas (DBSCAN) dengan algoritma *clustering* klasik seperti *K-Means*?
4. Selanjutnya, tujuan dalam penelitian ini diuraikan sebagai berikut:
5. Mengetahuialgoritma*clustering* berdasarkanmetodeDBSCAN.
6. Mengetahuialgoritma*clustering* berdasarkanmetodeDBSCAN.
7. Membandingkan final hasil*clustering*antarametodeberdasarkan densitas (DBSCAN) dengan algoritma *clustering* klasik seperti *K-Means*.

B. Landasan Teori

Clustering Hirarki

Analisis *cluster* umumnya terbagi menjadi dua metode salah satunya yakni *clustering* hirarki. Metode yang biasa menggunakan alat bantu *dendogram* ini memulai pengelompokkan dengan dua atau lebih objek yang mempunyai kemiripan terdekat lalu diteruskan ke objek lain yang mempunyai kedekatan kemiripan selanjutnya. Demikian seterusnya hingga *cluster* membentuk semacam pohon dengan tingkatan-tingkatan (hirarki). Tingkatan tersebut terbentuk sesuai objek yang paling mirip sampai yang paling tidak mirip.

Terdapat beberapa kelebihan dari metode *clustering* hirarki ini diantaranya dapat mempermudah penafsiran, serta mempercepat pengolahan sehingga bisa menghemat waktu. Tetapi terdapat pula beberapa kelemahan seperti sering mengalami kesalahan pada data *outlier* apalagi pengelompokkan dengan skala besar, dan tidak pernah bisa membatalkan apa yang telah dilakukan sebelumnya.

Clustering Non hirarki

Metode lain dari analisis *cluster* selain *clustering hirarki* adalah *clustering* non hirarki. Metode ini jelas berlawanan dari metode *clustering* hirarki, metode non hirarki tidak membentuk pohon tingkatan seperti yang dilakukan metode hirarki. Metode ini menempatkan sekaligus objek-objek ke dalam *cluster* hingga terbentuknya sejumlah *cluster* tertentu.

Metode *clustering* non hirarki ini terus mengalami perkembangan. Salah satu cabang dari *clustering* tersebut adalah jenis *clustering* partisi *k-means.*

Algoritma *Clustering* Partisi *k-means*

K-means merupakan salah satu cabang dari *clustering* non hirarki yang mampu melakukan analisis sampel dalam ukuran yang lebih besar dengan lebih efisien namun sering terdapatnya kekurangan dalam hal penentuan pusat *cluster* (*centroid*) yang tidak menentu.

Dalam algoritma *k-means* inputnya berupa data dan *k* buah *cluster* yang diinginkan. Algoritma ini akan mengelompokkan data ke dalam *k* buah kelompok. Pada setiap *cluster* terdapat *centroid* yang merepresentasikan *cluster* tersebut.

Adapun algoritma dasar dari *clustering k-means* ini adalah sebagai berikut (Tan dan Kumar, 2005):

1. Pilih *k* buah titik sebagai *centroid* awal
2. *Clustering* 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
5. Pembaharuan suatu titik centroid dapat dilakukan dengan rumus berikut (Maimon dan Rokach, 2005)

$$\mu_k = \frac{1}{n_k} \sum_{y=1}^{N_k} x_y \quad \text{... (2.1)}$$

Dimana μ_k merupakan titik *centroid cluster* ke-*k*, n_k = banyaknya data pada *cluster* ke-k, dan x_y = data ke-y pada cluster ke-k.

Proses pengelompokkan data ke dalam suatu *cluster* dapat dilakukan dengan cara menghitung jarak terdekat dari suatu data ke sebuah titik *centroid*. Perhitungan jarak *Euclidean* dapat digunakan untuk menghitung tingkat kemiripan sampel. Jarak *Euclidean* dirumuskan sebagai berikut:

$$d_{ij} = \sqrt{\sum_{d=1}^{D} \left(x_{id} - y_{jd}\right)^2} \quad \text{... (2.2)}$$

Dimana d_{ij} merupakan jarak *euclidean* antara objek i dan j, *D* merupakan banyaknya dimensi variabel, x_{id} merupakan koordinat dari objek i pada dimensi d, dan x_{jd} merupakan koordinat dari objek j pada dimensi d.

Algoritma *Clustering* Berdasarkan Densitas (DBSCAN)

Density-Based Spatial Clustering of Application with Noise (DBSCAN) merupakan sebuah metode *clustering* yang membangun area berdasarkan densitas yang terkoneksi (*density connected*). DBSCAN adalah jenis *clustering* partisi dimana daerah yang densitasnya tinggi dianggap sebagai *cluster* sedangkan yang densitasnya rendah atau tidak tergabung dalam *cluster* dianggap sebagai *noise* (Nagpal dan Mann, 2011).

Dalam Algoritma ini dikenal beberapa istilah seperti berikut:

1. *Core* : Titik pusat dalam *cluster* didasarkan pada densitas dimana ada sejumlah titik yang harus berada dalam *Eps* (radius atau nilai ambang batas), M*inPts* (minimal titik dalam *cluster*) yang ditentukan pengguna.
2. *Border* : Titik yang menjadi batasan dalam kawasan titik pusat (*core)*.
3. *Noise* : Titik yang tidak dapat dijangkau oleh *core* dan bukan merupakan *border*.

$$noise = \{x \in X \mid \forall i : x \notin C_i\} \quad \text{... (2.3)}$$

Dimana *X*merupakan gugus data, dan C_i merupakan *cluster* ke-*i*.

4. Densitas terjangkau langsung: Sebuah titik dikatakan titik terjangkau langsung apabila titik tersebut terhubung secara langsung dengan titik pusat (*core*).

$$x \in N_{Eps}(y) \wedge |N_{Eps}(y)| \geq MinPts \quad \text{... (2.4)}$$

Dimana N_{Eps} merupakan titik sekitar dari radius *Eps*, *MinPts* merupakan minimal titik dalam *cluster*.

5. Densitas terjangkau : Sebuah titik dikatakan titik terjangkau apabila titik tersebut terhubung secara tidak langsung dengan titik pusat (*core*).

6. Densitas terhubung : Sebuah titik dikatakan saling terhubung satu sama lain oleh titik lain.

   DBSCAN memerlukan dua parameter input yaitu *epsilon* (*Eps*) dan titik minimum (*MinPts*). *Eps*-titik sekitar didefinisikan sebagai:

   $$N_{Eps}(x) = \{y \in D \mid dist(x, y) \leq Eps\} \quad \text{... (2.5)}$$

   Dimana $N_{Eps}(x)$ merupakan titik sekitar dari x dalam radius *Eps*, *D* merupakan gugus data, *dist(x,y)* merupakan jarak *euclidean* dari objek x dan y, dan *Eps* merupakan radius atau ambang batas.

Keuntungan dari algoritma ini diantaranya (Mumtaz dan Duraiswamy, 2010):

1. Tidak mengharuskan untuk mengetahui jumlah *cluster*.
2. Dapat menemukan *cluster* berbentuk sewenang-wenang.
3. Mampu mengatasi *noise*.

DBSCAN secara umum dimulai dengan titik awal secara acak. Kemudian menemukan semua titik sekitar dalam *Eps* jarak titik awal. Jika jumlah titik sekitar lebih besar dari atau sama dengan *MinPts* maka *cluster* terbentuk. Jika jumlah titik sekitar kurang dari *MinPts* maka ditandai sebagai *noise.*

Adapun algoritma dari DBSCAN adalah sebagai berikut:

1. Pilih titik awal r secara acak
2. Inisialisasi parameter input: *minpts* dan *eps*
3. Hitung *Eps* atau semua jarak densitas terjangkau terhadap r menggunakan jarak *euclidean*.
4. Jika titik yang memenuhi *Eps* lebih dari *MinPts* maka titik r adalah titik pusat (*core*) dan *cluster* terbentuk dengan mengikuti kondisi berikut:

   $1.\forall x, y$: jika x $\in$ C (*cluster*) dan y densitas terjangkau dari x maka y $\in$ C. (*Maximality*) ... (2.6)

   $1.\forall x, y \in$ C: x densitas terhubung dengan y. (*Connectivity*) ... (2.7)
5. Ulangi langkah 3 – 4 hingga semua titik diproses
6. Jika r adalah titik *border* dan tidak ada titik yang densitas terjangkau terhadap r, maka proses dilanjutkan ke titik yang lain. (Anindya, dkk., 2015)

C. Hasil Penelitian dan Pembahasan

Cleaning Data dan Reduksi Dimensi

Dalam bagian ini akan dilakukan pembersihan data menggunakan *softwareMicrosoft Excel* 2013. Dimana dalam proses pembersihannya, data *missing* dan data *redundant* akan dihilangkan. Adapun variabel dalam data debitur akan direduksi menggunakan teknik *Principal Component Analysis* (PCA). PCA yang digunakan merupakan *toolbox* dari *software* Matlab 8.6.0. Sehingga data hasil *cleaning* dan reduksi dimensi tersaji dalam Tabel 4.1.

Tabel 4.1. Data Hasil *Cleaning* dan Reduksi

Nasabah	X_1	X_2
1	0,0057	-0,0064
2	-0,0422	0,0033
3	0,0321	-0,0169
4	0,0051	-0,0261
...	...	...
1000	0,0261	-0,0141

Sumber :Kaggle (2011)

Berdasarkan Tabel 4.1 data nasabah yang memiliki fasilitas kredit (debitur) direduksi sedemikian hingga menjadi dua variabel, sehingga proses selanjutnya hanya akan melibatkan dua variabel tersebut yang mewakili keseluruhan data debitur.

Hasil *Clustering* DBSCAN

DBSCAN diawali dengan menetapkan nilai dari parameter yang akan diinputkan. Dalam pemrogramnya nilai parameter *Eps* ditetapkan 0,0128 sedangkan parameter *MinPts* ditetapkan 5. Berikut rincian hasil *running* program DBSCAN:

Matrik Jarak

Matrik jarak akan berukuran sebanyak n x n. Sehingga untuk data nasabah diatas akan diperoleh matrik jarak berukuran 1000 x 1000.

Matrik jarak untuk data nasabah yang memiliki fasilitas kredit disajikan dalam Tabel 4.3 berikut:

Tabel 4.2. Matrik Jarak (Jarak *Euclidean*)

Jarak	1	2	3	4	…	1000
1	0	0,0489	0,0284	0,0197	...	0,0218
2	0,0489	0	0,0770	0,0557	..	0,0705
3	0,0284	0,0770	0	0,0285	..	0,0066
4	0,0197	0,0557	0,0285	0	...	0,0242
...	...	...	...	...	...	...
1000	0,0218	0,0705	0,0066	0,0242	...	0

Berdasarkan Tabel 4.2 dapat dilihat bahwa matrik jarak ini merupakan matrik simetri dengan nilai dari matrik segitiga bawah sama dengan matrik segitiga atas dan diagonalnya bernilai 0 (nol). Dikarenakan matrik jarak ini merupakan tolak ukur kemiripan yang digunakan untuk menggelompokkan data maka semakin kecil jarak *euclidean* akan semakin mirip pula kedua objek tersebut dan nilai 0 pada diagonal berarti bahwa tidak ada jarak antara objek atau dikatakan sangat mirip.

Penggolongan *noise*

Noise dalam data debitur ini dipisahkan dari *cluster-cluster* yang terbentuk dengan kriteria mengikuti Persamaan (2.3) dimana jumlah dari titik sekitar (*neighbour*) yang kurang dari *MinPts* maka titik itulah yang dikategorikan sebagai *noise*. Adapun penggolongan *noise* dalam data debitur tersebut disajikan dalam Tabel 4.3.

Tabel 4.3. Penggolongan *Noise*

Nasabah	*Noise*	Nasabah	*Noise*	Nasabah	*Noise*
12	*TRUE*	177	*TRUE*	375	*TRUE*
23	*TRUE*	185	*TRUE*	419	*TRUE*
25	*TRUE*	199	*TRUE*	439	*TRUE*
76	*TRUE*	204	*TRUE*	454	*TRUE*
...	...	...	...	...	...
173	*TRUE*	355	*TRUE*	983	*TRUE*

Berdasarkan Tabel 4.3 debitur dengan kategori *true* berarti bahwa debitur tersebut tergolong *noise*. Output dari Matlab menunjukkan ada 51 titik dari 1000 titik data yang tergolong noise dan berarti bahwa dari data tersebut 5,1% terdiri dari gangguan/*noise*.

***Cluster* final**

Hasil akhir dari algoritma DBSCAN ini yaitu terbentuknya *cluster-cluster* dengan *noise* yang sudah dipisahkan. Dengan *Eps* 0,0128 dan *MinPts* 5, c*luster-cluster* yang terbentuk disajikan dalam Tabel. 4.4

Tabel 4.4. Peng*cluster*-an Debitur (DBSCAN)

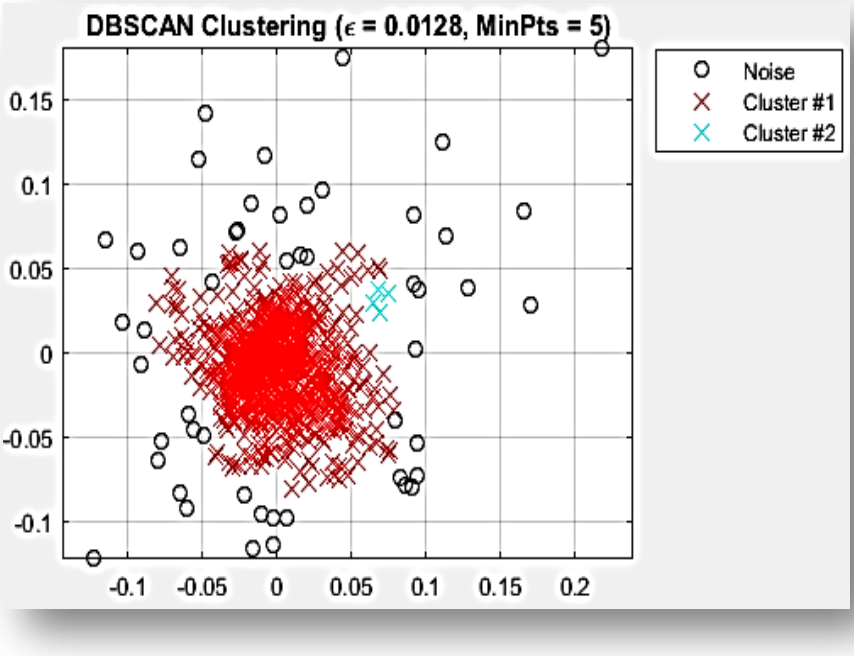
Nasabah	*Cluster ke*
1	1
2	1
...	...
261	2
...	...
1000	1

Berdasarkan Tabel 4.4 algoritma DBSCAN membentuk 2 *cluster* dengan anggota dari *cluster* 1 sebanyak 996 orang, *cluster* 2 sebanyak 4 orang, Adapun interpretasi dari masing-masing *cluster* tersebut disajikan dalam Tabel 4.5 berikut:

Tabel 4.5. Interpretasi *Cluster*

Cluster	Interpretasi
1	Debitur yang memiliki fasilitas kredit dengan kategori baik, sehingga pihak Bank bisa memberikan keputusan untuk menyetujui permohonan kredit yang diajukan
2	Debitur yang memiliki fasilitas kredit dengan kategori buruk, sehingga pihak Bank bisa memberikan keputusan untuk menolak permohonan kredit yang diajukan

Selanjutnya untuk memudahkan melihat pola yang terbentuk oleh *cluster-cluster* tersebut, secara visualisasi Gambar 4.1 menampilkan plot hasil *Clustering* menggunakan metode DBSCAN.



Gambar 4.1 *Clustering* menggunakan metode DBSCAN

Berdasarkan Gambar 4.1 pola yang terbentuk mengikuti pola globular tetapi secara umum DBSCAN mampu meng*cover* data yang berbentuk sembarang (*arbitrary*).

Hasil *Clustering K-means*

Algoritma *K-means* mensyaratkan untuk menentukan jumlah *cluster* yang akan dibentuk (Lampiran 3). Penetapan jumlah *cluster* akan disamakan dengan jumlah *cluster* yang terbentuk oleh algoritma DBSCAN yaitu sebanyak 2 *cluster*. Berikut rincian hasil *running* program *K-means*:

Centroid

Dalam metode *K-means centroid* akan terus diperbaharui/dihitung kembali sampai semua data diproses dan tidak ditemukannya data yang berpindah. *Centroid* (c)untuk data debitur adalah sebagai berikut: c_{11} = -0,0154, c_{12} = -0,0002, c_{21} = 0,0299, c_{22} = -0,0223. Nasabah dengan jarak terdekat dengan *centroid* akan dikelompokkan menjadi satu *cluster* dan sisanya menjadi anggota dari *cluster* lainnya. Proses diulangi sampai tidak ditemukannya nasabah yang berpindah *cluster*.

***Cluster* final**

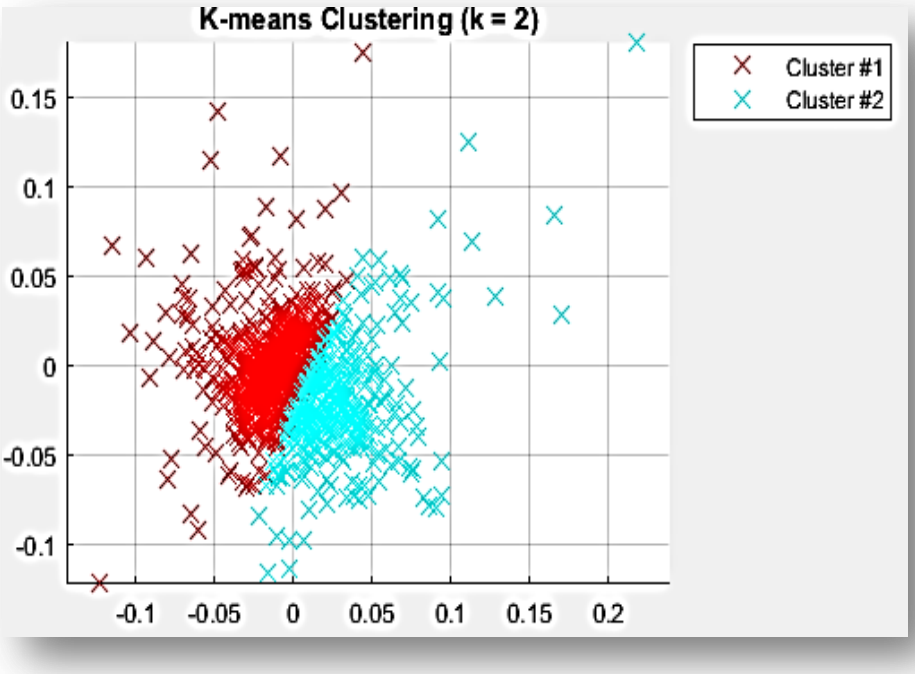
Setelah semua data diproses dan tidak ditemukan data yang berpindah maka proses pengelompokkan berdasarkan metode *k-means* berakhir. Adapun hasil *cluster* dari iterasi terakhir disajikan dalam Tabel 4.7

Tabel 4.6. Peng*cluster*-an Debitur (*K*-means)

Nasabah	*Cluster ke*
1	2
2	2
3	1
4	1
...	...
1000	1

Berdasarkan Tabel 4.6 algoritma *k-means* membentuk 2 *cluster* dengan anggota dari *cluster* 1 sebanyak 663 orang, dan *cluster* 2 sebanyak 337 orang.

Selanjutnya untuk memudahkan melihat pola yang terbentuk oleh *cluster-cluster* tersebut, secara visualisasi Gambar 4.2 menampilkan plot hasil *Clustering* menggunakan metode *k-means*.



Gambar 4.2 *Clustering* menggunakan metode *k-means*

Berdasarkan Gambar 4.2 *K-means* tidak memperhatikan ukuran dan bentuk dari *cluster*.

Perbandingan Final Hasil *Clustering* DBSCAN dengan *Clustering K-means*

Hasil *clustering* DBCAN akan dibandingkan dengan hasil dari *clustering K-means*, dimana akan dilihat berdasarkan 5 kriteria pembanding.

1. Kompleksitas (*Complexity*)

Ukuran efisiensi dari algoritma DBSCAN adalah O(n x log n) ketika tidak menggunakan indeks percepatan. Kompleksitas ini juga dipengaruhi *noise*, dimana ketika *noise* meningkat maka kompleksitas dari algoritma DBSCAN ini akan semakin memburuk. Sedangkan untuk algoritma *K-means* adalah O(kn).

2. Bentuk Cluster (Shape of Clusters)

Sebagaimana hasil dari *running* program, DBSCAN mendukung peng*cluster*an dari bentuk data yang tidak beraturan (*arbitrary*) sehingga bentuk *cluster* dari DBSCAN ini bebas sedangkan *k-means* merupakan algoritma yang tidak memeperhatikan bentuk *cluster*.

3. Parameter Input (*Input Parameters*)

DBSCAN bergantung pada dua parameter masukkan (*Input Parameter*) yaitu nilai dari *Eps* dan *MinPts.* Sedangkan *k-means* mesyaratkan penentuan jumlah *cluster (k)* yang akan dibentuk.

4. Kemampuan Menghadapi *Noise* (*Handling of Noise*)

Dalam kasus ini DBSCAN mampu menghadapi data yang mengandung *noise* baik sedangkan *k-means* tidak memperhatikan adanya *noise*.

5. Run Time

Waktu berjalannya suatu proses algoritma merupakan tolak ukur yang sering dijadikan pertimbangan penting oleh pengguna untuk memilih suatu algoritma. Adapun *run time* DBSCAN dengan data debitur tersebut adalah 0.305 detik. Sedangkan *run time* untuk *k-means* adalah 0.546 detik.

Secara ringkas perbandingan final hasil *clustering* DBSCAN dengan *clustering K-means* ditampilkan dalam Tabel 4.7 berikut:

Tabel 4.7. Analisis Komparatif Algoritma DBSCAN dan *K-means*

Nama Algoritma	**Kompleksitas**	**Bentuk *Cluster***	**Parameter Input**	**Kemampuan Menghadapi *Noise***	***Run Times***
DBSCAN	O (n x log n)	*Arbitrary*	*Eps* dan *MinPts*	Ya	0.305 s
K-means	O (kN)	Tidak diperhatikan	k	Tidak	0.546 s

D. Kesimpulan

Berdasarkan hasil dan pembahasan yang telah dilakukan yaitu dengan mengelompokkan (*clustering*) data nasabah yang memiliki fasilitas kredit menggunakan metode DBSCAN dan *k-means* dapat ditarik beberapa kesimpulan, diantaranya :

1. DBSCAN dengan *Eps* 0,0128dan *MinPts* 5, dapat mengelompokkan data debitur menjadi 2 *cluster* dengan *cluster* 1 merupakan kelompok debitur dengan kategori kredit baik, sedangkan *cluster* 2 merupakan kelompok debitur dengan kategori kredit buruk.
2. Dari data debitur *k-means* dengan jumlah *cluster* (k=2) yang memiliki *centroid* (c_{11} = -0,0154, c_{12} = -0,0002, c_{21} = 0,0299, c_{22} = -0,0223) berarti bahwa dari setiap kelompokdiperoleh rata-rata *cluster* sebesar *centroid* ke-*ij*.
3. Berdasarkan lima kriteria pembanding, DBSCAN lebih unggul daripada *k-means* dalam hal kompleksitas O(n x log n) dibandingkan dengan O(kn), serta mampu mengcover bentuk *cluster* yang tidak beraturan, kemampuannya menghadapi *noise*, dan *run time* 0,305 detik dibandingkan 0,546 detik dalam meng*clustering* data nasabah yang memiliki fasilitas kredit (debitur).

E. Saran

1. Disarankan kepada Bank-bank yang ada di Indonesia untuk mempertimbangkan metode DBSCAN dalam hal menggelompokkan nasabah yang memiliki fasilitas kredit (debitur) seperti kredit macet di Indonesia.
2. Kepada peneliti/pengguna lain, disarankan untuk meneliti metode DBSCANlebih dalam lagi apabila terdapat banyak *noise* dalam data dan ditemukannya berbagai densitas.

Daftar Pustaka

Anindya Santika Devi, N. M., Gede Darma Putra, I. K., & Sukarsa, I. (2015). *Implementasi Metode Clustering DBSCAN pada Proses Pengambilan Keputusan*. Lontar Komputer, 6(3).

B. Santosa. (2007). *Data Mining. Teknik Pemanfaatan Data untuk Keperluan Bisnis, First Edition ed.* Yogyakarta: Graha Ilmu.

Ester, M., Kriegel, H.P., Sander, J. and Xu, X. (1996) *A Density Based Algorithm for Discover Clusters in Large Spatial Datasets with Noise*. *Proceedings of International Conference on Knowledge Discovery and Data Mining*, 226-231.

Guha, S., Rastogi, R. and Shim. K, (2001) *Cure: An Efficient Algorithm for Large Databses,Information Systems*, 26, 35-58.

Kaggle. 2011. *Give Me Some Kredit*. https://www.kaggle.com/c/GiveMeSomeCredit/ data diunduh 26 Juli 2016.

Laeli, S. (2014). *Analisis cluster dengan average linkage method dan ward's method untuk data responden nasabah Asuransi Jiwa Unit Link* (Doctoral dissertation, UNY).

Maimon, O., & Rokach, L. (Eds.). (2005). *Data mining and knowledge discovery handbook* (Vol. 2). New York: Springer.

Mumtaz, K. and Duraiswamy, K., (2010). *An analysis on density based clustering of multi dimensional spatial data*. *Indian Journal of Computer Science and Engineering, 1*(1), pp.8-12.

Nagpal, P. B., & Mann, P. A. (2011). Comparative study of density based clustering algorithms. *International Journal of Computer Applications*, *27*(11), 44-47.

Tan, M. Steinbach, and V. Kumar, (2005) *Introduction to Data Mining, (First Edition)*. Boston, MA, USA: Addison-Wesley Longman Publishing CO., Inc.

Yan, B., and Deng, G. (2015). *Improved Clustering Algorithm Based on Density-Isoline. Open Journal of Statistics, 5*(04), 303-310.