BAB II
KAJIAN LITERATUR

2.1. Tinjauan Pustaka

Pada subbab ini, akan dijelaskan tinjauan pustaka yang berkaitan dengan penelitian yang akan dilakukan.

2.1.1. *E-Commerce*

E-commerce adalah istilah yang dapat digunakan pada semua jenis bisnis yang melibatkan transaksi jual beli secara elektronik, terutama internet. *E-commerce* terus meluas dan diperkirakan akan berlanjut tanpa berhenti karena memungkinkan pelanggan untuk melakukan pertukaran barang tanpa dibatasi oleh waktu dan jarak. Pengantaran barang juga cenderung lebih cepat, murah, dan nyaman dibandingkan bisnis konvensional (Baynal, K. dan Boyaci, A.I., 2016). Pengguna mengalami kesulitan dalam mencari barang secara cepat dalam bisnis konvensional sehingga fitur *search* hadir untuk mempercepat dan memudahkan pengguna dalam mendapatkan informasi yang dibutuhkan oleh pengguna (Santu, S.K.K., Sondhi, P., dan Zhai, C.X., 2017 dan Sondhi, P. et al., 2018).

Transaksi jual beli dalam bisnis global telah bergerak menuju ke *e-commerce* berbasis B2B (*business-to-business*) karena banyaknya kolaborasi TI dengan bidang-bidang lainnya. Para pengguna yang berubah menjadi pelanggan telah mengerti pasar global terbuka, bahkan telah menjadi kebiasaan untuk membandingkan harga dari beberapa *e-commerce* sebelum memulai transaksi. Transparansi menjadikan *e-commerce* berlomba-lomba menghadirkan mitra penjual dari hari ke hari sehingga pelanggan tidak hanya mendapatkan harga barang yang murah, tetapi juga variatif dan kualitas barang yang baik (Khan, A.G., 2016).

Perkembangan TI yang tidak pernah berhenti memberikan implikasi kepada pendidikan, kesehatan, ekonomi, industri, manufaktur, perdagangan, perbankan, dan hiburan. *E-commerce* adalah salah satu cabang penting dari TI yang terus berkembang mengikuti perkembangan TI tersebut. Pemerintah

Indonesia turut peduli terhadap perkembangan ini dan mewajibkan pengusaha *e-commerce* untuk mendaftarkan diri secara resmi ke website PSE (Penyelenggara Sistem Elektronik) sehingga implementasi *e-commerce* dapat dipastikan berjalan dengan baik (Nanehkaran, Y.A., 2013 dan RPP TPMSE).

Terdapat mekanisme yang berbeda dan unik antara *e-commerce* dan *traditional commerce* (bisnis konvensional). Dalam mekanisme *e-commerce*, komponen-komponen yang terlibat di dalamnya adalah (Turban, E. et al., 2015) :

1. Pembeli

   Pembeli merupakan para pengguna internet yang dapat dijadikan sebagai target pasar yang potensial untuk diberikan penawaran berupa produk, jasa, atau informasi oleh para penjual. Tujuan dasar kehadiran pembeli adalah aktivitas tawar-menawar, mencari barang-barang yang sesuai kebutuhan, mengoleksi barang, hiburan, dan lain-lain.

2. Penjual

   Penjual merupakan pihak yang menawarkan produk, jasa, atau informasi kepada para pembeli. Proses penjualan dapat dilakukan secara langsung melalui *platform* yang dimiliki oleh penjual tersebut atau melalui *e-commerce* atau *marketplace*.

3. Barang dan jasa

   Salah satu perbedaan antara *e-commerce* dan *traditional commerce* terletak pada produk yang dijual. Pada dunia maya, penjual dapat menjual produk *digital*. Produk *digital* dapat dikirimkan secara langsung melalui internet.

4. Infrastruktur

   Infrastruktur pasar yang menggunakan media elektronik meliputi perangkat keras, perangkat lunak, dan juga sistem jaringannya.

5. *Front end*

   Front end merupakan tempat yang disediakan untuk berinteraksi dengan pengguna, misalnya portal penjual, katalog elektronik, *shopping cart*, mesin pencari, halaman *payment gateway*, dan semua aktivitas yang berhubungan dengan aktivitas penempatan pesanan.

6. *Back end*

   Back end merupakan aktivitas-aktivitas yang berkaitan dengan pemesanan barang, manajemen inventori, proses pembayaran, *packaging*, dan pengiriman.

7. *Intermediary*

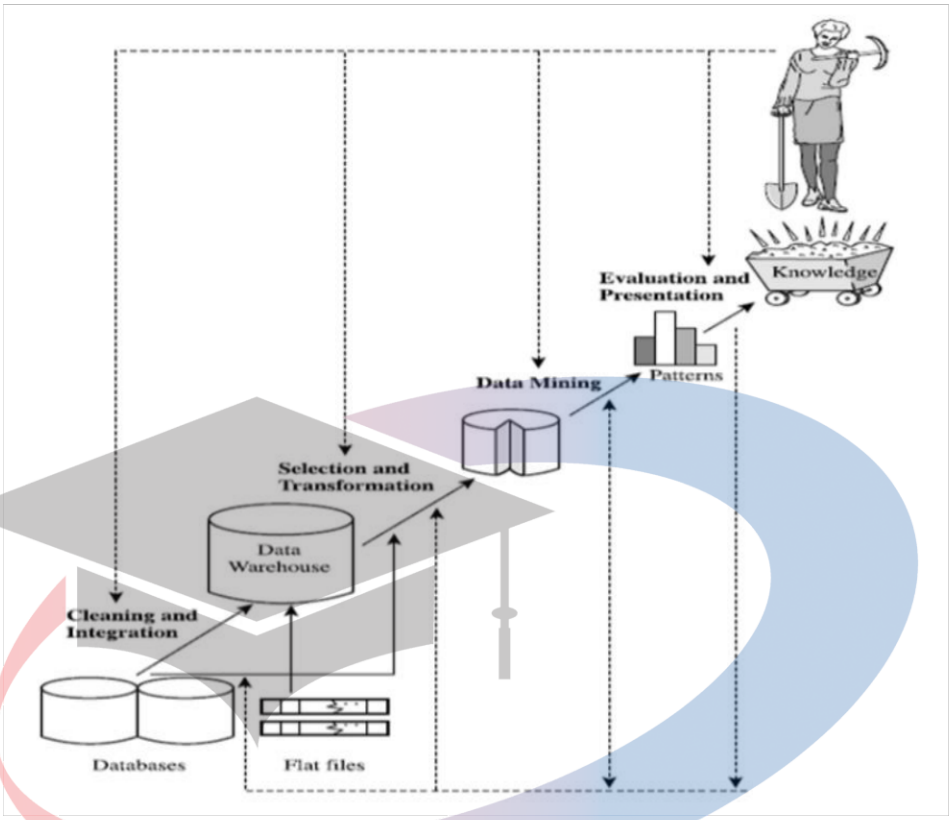
   Intermediary merupakan pihak ketiga yang membantu aktivitas pembeli dan penjual, dalam dunia pemasaran disebut sebagai perantara seperti grosir atau pengecer.

2.1.2. *Data Mining*

Data mining didefinisikan sebagai berikut :

1. Proses menemukan pola yang menarik, dan pengetahuan dari data yang berjumlah besar (Han, J., Kamber, M., dan Pei, J., 2012).
2. Proses pencarian melalui data dengan jumlah yang besar, dalam sebuah usaha untuk menemukan pola, tren, dan hubungan (Dunstone, T. dan Yager, N., 2009).
3. Sebuah cara untuk mendapatkan pengetahuan pasar dari sejumlah data yang besar (Keating dan Berry, 2008).

Berdasarkan beberapa definisi di atas, *data mining* adalah suatu proses analisis untuk menggali informasi yang tersembunyi dalam sejumlah data yang berukuran besar, sehingga ditemukan suatu pola yang dapat dijadikan sebagai pengetahuan. Langkah-langkah tersebut akan dijelaskan pada gambar berikut.



Gambar 2.1. Tahap Penemuan Pengetahuan Pada *Data Mining*

(Han, J., Kamber, M., dan Pei, J., 2012)

Gambar di atas menggambarkan proses penemuan pengetahuan pada *data mining* yang terdiri dari beberapa tahap (Han, J., Kamber, M., dan Pei, J., 2012) :

a. *Data Cleaning*: untuk menghapus data yang tidak dipakai dan data yang tidak konsisten.
b. *Data Integration*: berbagai sumber data dapat digabungkan.
c. *Data Selection*: data yang bersangkutan pada tugas analisis diseleksi dan diambil kembali dari *database*.
d. *Data Transformation*: data diubah atau diperkuat menjadi bentuk yang seharusnya untuk diolah dengan menganalisis ringkasan atau jumlah total agregasi.
e. *Data Mining*: sebuah proses penting dimana metode *intelligent* diterapkan dengan tujuan untuk mengolah pola-pola data.

f. *Pattern Evaluation*: untuk mengidentifikasi pola-pola menarik yang menjelaskan mengenai ukuran dasar pengetahuan yang ada.
g. *Knowledge Presentation*: visualisasi dan teknik representasi pengetahuan digunakan untuk menyajikan pengetahuan yang telah diolah untuk pengguna.

Data mining telah diimplementasikan pada kecerdasan buatan (*artificial intelligent*), *machine learning*, statistik dan *database*. Beberapa teknik yang sering diterapkan dalam *data mining* antara lain *classification*, *association*, dan *clustering*, yang akan dijelaskan pada subbab berikut (Han, J., Kamber, M., dan Pei, J., 2012).

2.1.2.1. *Classification*

Satu bentuk analisis data yang menghasilkan model untuk mendeskripsikan kelas data yang penting adalah klasifikasi. Klasifikasi adalah tipe analisis data yang membantu proses pengklasifikasian setiap *item* dalam satu *set* data ke dalam salah satu *set class* yang telah ditentukan. Klasifikasi memprediksi kategori ke dalam label *class*. Proses ini digunakan untuk menemukan *class* target untuk setiap kasus dalam data, misalnya untuk mengidentifikasi tingkat resiko kredit pelanggan yang ingin melakukan peminjaman uang (Kesavaraj, G. dan Sukumaran, S., 2013).

Model klasifikasi dapat berupa *if-then-rules*, *decision tree*, formula matematis atau *neural network*. Analisis tersebut dapat membantu memberikan pemahaman yang lebih baik dari data pada umumnya. Klasifikasi telah diaplikasikan ke beberapa bidang seperti deteksi penipuan, target pemasaran, prediksi kinerja, manufaktur, dan diagnosa medis (Han, J., Kamber, M., dan Pei, J., 2012).

Salah satu konsep dan metode dari klasifikasi adalah *Decision tree*. *Decision tree* sebagai model prediksi menggunakan struktur pohon atau struktur berhirarki dengan mempertimbangkan atribut-atribut yang relevan. *Decision tree* menggunakan struktur *flowchart* yang menyerupai *tree* (pohon), dimana setiap simpul internal menandakan suatu tes pada atribut, setiap cabang

merepresentasikan hasil tes, dan simpul daun merepresentasikan kelas atau distribusi kelas. Alur pada *decision tree* ditelusuri dari simpul akar ke simpul daun yang memegang prediksi kelas untuk contoh tersebut (Han, J., Kamber, M., dan Pei, J., 2012)

2.1.2.2. *Association*

Association digunakan untuk mengenali kelakuan dari kejadian-kejadian khusus atau proses dimana *link* asosiasi muncul pada setiap kejadian. *Association* sangat membantu aktivitas yang berkaitan dengan pembuatan keputusan dengan mencari hubungan antar entitas dari suatu *dataset*. Hubungan tersebut memberikan pemahaman lebih mendalam seperti interpretasi yang menjelaskan pola semantik yang sering terjadi aktivitas belanja sehingga dapat dijadikan sebagai dasar pengetahuan (Ziauddin et al., 2012).

Aturan asosiatif yang paling umum digunakan adalah analisis keranjang belanja. Ketika seorang pelanggan membeli produk, produk apa yang dimasukkan ke dalam keranjang bersamaan dengan produk lainnya dapat ditentukan nilai probabilitasnya, sehingga manajer atau pemilik toko dapat mengatur posisi rak sesuai dengan pengetahuan yang diperoleh. Dengan demikian, pelanggan dapat menjangkau produk-produk dengan mudah dan berimplikasi kepada kenaikan tingkat penjualan. Misalnya pelanggan yang membeli komputer juga akan membeli perangkat lunak antivirus pada saat bersamaan yang didefinisikan dengan komputer => antivirus [*support* = 2%, *confidence* = 60%] (Yazgana, P. dan Kusakci, A.O., 2016 dan Han, J., Kamber, M., dan Pei, J., 2012).

Aturan *support* dan *confidence* adalah dua ukuran penting dalam aturan asosiatif yang mencerminkan kegunaan dan kepastian dari aturan yang ditentukan. Berdasarkan contoh pada paragraf sebelumnya, *support* sebesar 2% adalah hasil analisis yang menjelaskan bahwa ketika membeli komputer juga akan mendukung pembelian perangkat antivirus. Sementara *confidence* 60% menjelaskan bahwa tingkat kepercayaan ketika membeli komputer juga akan mendukung pembelian perangkat antivirus. Biasanya, aturan asosiatif dianggap menarik ketika keduanya memenuhi ambang dukungan minimum dan batas kepercayaan minimum, dimana

ambang batas ini dapat diatur oleh pengguna atau seorang pakar (Han, J., Kamber, M., dan Pei, J., 2012).

2.1.2.3. *Clustering*

Clustering adalah proses pengelompokan sekumpulan data yang memiliki kemiripan ke dalam *cluster* yang sama. *Clusters* yang dihasilkan dalam *clustering* kemudian dianalisis apakah dalam satu *cluster* sudah memiliki kemiripan atau belum, tetapi berbeda dengan *cluster* lainnya. Proses *clustering* dalam keseharian banyak digunakan untuk mendapatkan informasi, ekstraksi data, dan *data mining*. (Smythisri, M. dan Emadhiya, M., 2014 dan Ahmad, P.H. dan Dang, S., 2015).

Seiring dengan banyaknya penerapan *clustering* sehingga terdapat banyak teknik untuk implementasinya. *Clustering* secara luas dibagi menjadi (Ahmad, P.H. dan Dang, S., 2015) :

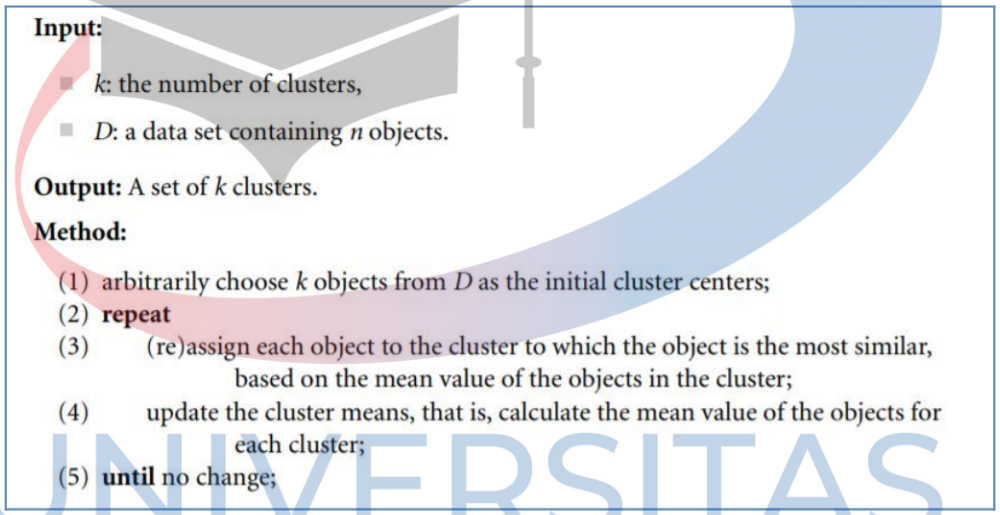
1. Partisi, yaitu mengidentifikasi *cluster* sebagai area yang sangat padat dengan data, contohnya *k-means*.
2. Hirarki, yaitu membangun *cluster* secara bertahap tetapi kurang sensitif terhadap *noise*.
3. *Density*, yaitu mengelompokkan data berdasarkan kepadatan, contohnya DBSCAN.

Analisis *cluster* telah diaplikasikan ke berbagai hal seperti *Business Intelligence*, *Image Pattern Recognition*, *Web Search*, *Biology*, dan *Security*. Di dalam *business intelligence*, *clustering* dapat mengelompokkan banyak pelanggan ke dalam banyak *cluster* yang memiliki kemiripan yang kuat. *Clustering* juga dikenal sebagai *data segmentation*, karena *clustering* mempartisi banyak *dataset* ke dalam banyak *cluster* berdasarkan kemiripannya. *Clustering* dapat juga sebagai *outlier detection*, dimana *outlier* bisa menjadi menarik dibandingkan kasus biasa. Aplikasinya adalah *Outlier Detection* yang digunakan untuk mendeteksi *card fraud* dan mengawasi aktivitas kriminal dalam *e-commerce* (Han, J., Kamber, M., dan Pei, J., 2012).

2.1.3. Algoritma *Clustering*

2.1.3.1. k-Means

Istilah *k-Means* pertama kali digunakan oleh James MacQueen pada tahun 1967. *k-Means* adalah suatu metode analisis *data mining* yang melakukan pemodelan secara *unsupervised* dan berdasarkan pada tingkat kemiripan. *k-Means* mengelompokkan data yang ada ke dalam beberapa kelompok dengan sistem partisi, dimana data dalam masing-masing kelompok memiliki kemiripan satu sama lainnya tetapi memiliki perbedaan dengan data yang ada di kelompok lainnya. Gambar berikut adalah algoritma *k-Means*.



Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) **repeat**
(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) update the cluster means, that is, calculate the mean value of the objects for each cluster;
(5) **until** no change;

Gambar 2.2. Algoritma *k-Means*

(Han, J., Kamber, M., dan Pei, J., 2012)

k-Means meminimalisasi fungsi objektif yang diatur dalam proses *clustering* dengan cara meminimalkan variansi antar data dalam suatu *cluster* dan memaksimalkan variansi dengan data yang ada di *cluster* lainnya. Namun terdapat beberapa permasalahan yang sering muncul pada saat menggunakan *k-Means* yaitu (Raykov, Y.P. et al., 2016) :

1. Ditemukan beberapa model *clustering* yang berbeda.

   Permasalahan ini umumnya disebabkan oleh perbedaan proses inisialisasi anggota masing-masing *cluster*. Proses inisialisasi yang sering digunakan

adalah proses *random*. Dalam suatu studi perbandingan, proses inisialisasi secara *random* mempunyai kecenderungan untuk memberikan hasil yang lebih baik dan independen, walaupun dari segi kecepatan untuk konvergensi lebih lambat.

2. Pemilihan jumlah *cluster* yang paling tepat.

   Pemilihan jumlah *cluster* merupakan masalah laten dalam *k-Means*. Beberapa pendekatan telah digunakan dalam menentukan jumlah *cluster* yang paling tepat untuk suatu *dataset* yang dianalisis termasuk di antaranya *Partition Entropy* (PE) dan *Gap Statistics*.

3. Kegagalan konvergensi.

   Permasalahan ini dapat terjadi dalam metode *Hard k-Means* maupun *Fuzzy k-Means*. Kemungkinan ini akan semakin besar terjadi pada metode *Hard k-Means*, karena setiap data di dalam *dataset* dialokasikan secara tegas (*hard*) untuk menjadi bagian dari suatu *cluster* tertentu. Perpindahan suatu data ke suatu *cluster* tertentu dapat mengubah karakteristik model *clustering* yang dapat menyebabkan data yang telah dipindahkan tersebut lebih sesuai untuk berada di *cluster* semula sebelum data tersebut dipindahkan. Demikian juga dengan keadaan sebaliknya. Kejadian seperti ini tentu akan mengakibatkan pemodelan tidak akan berhenti dan kegagalan untuk konvergensi akan terjadi. Untuk *Fuzzy k-Means*, walaupun ada, kemungkinan permasalahan ini untuk terjadi sangatlah kecil, karena setiap data dilengkapi dengan *membership function* (*Fuzzy k-Means*) untuk menjadi anggota *cluster* yang ditemukan.

4. *Outliers*.

   Permasalahan yang umum yang terjadi hampir di setiap metode yang melakukan pemodelan terhadap data. Khusus untuk metode *k-Means* hal ini memang menjadi permasalahan yang cukup menentukan. Beberapa hal yang perlu diperhatikan dalam mendeteksi *outliers* dalam proses pengelompokan data termasuk bagaimana menentukan apakah suatu data item merupakan *outliers* dari suatu *cluster* tertentu dan apakah data dalam jumlah kecil yang membentuk suatu *cluster* tersendiri dapat dianggap sebagai *outliers*. Proses ini memerlukan suatu pendekatan khusus yang berbeda dengan proses

pendeteksian *outliers* di dalam suatu *dataset* yang hanya terdiri dari satu populasi yang homogen.

5. Bentuk *cluster*.

   Tidak seperti metode data *clustering* lainnya, *k-Means* umumnya tidak mengindahkan bentuk dari masing-masing *cluster* yang mendasari model yang terbentuk, walaupun secara natural masing-masing *cluster* umumnya berbentuk bundar. Untuk *dataset* yang diperkirakan mempunyai bentuk yang tidak biasa, beberapa pendekatan perlu untuk diterapkan.

6. *Overlapping*.

   Masalah *overlapping* adalah permasalahan terakhir yang sering diabaikan karena umumnya sulit terdeteksi. Hal ini terjadi untuk metode *Hard K-Means* dan *Fuzzy K-Means*, karena secara teori, metode ini tidak dilengkapi fitur untuk mendeteksi apakah di dalam suatu *cluster* terdapat *cluster* lain yang kemungkinan tersembunyi.

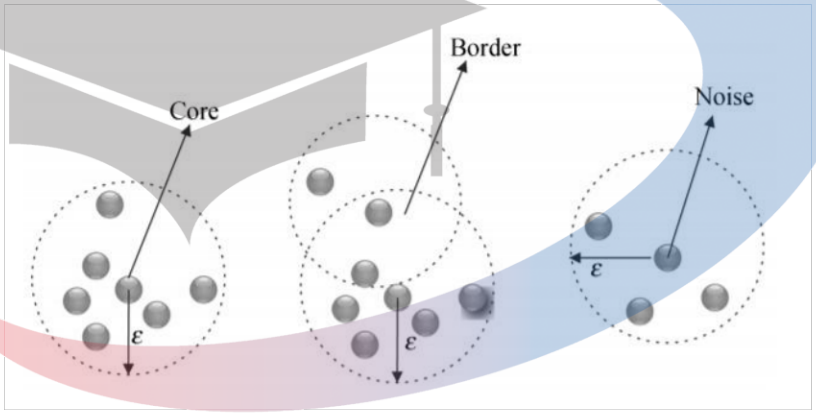
2.1.3.2. DBSCAN

Algoritma DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) pertama kali diusulkan oleh Martin Ester, Hans-Peter Kriegel, Jörg Sander and Xiaowei Xu pada tahun 1996. DBSCAN adalah algoritma *unsupervised* yang mampu menemukan objek *core* berdasarkan tingkat kerapatan data (berbasis *density*). DBSCAN menumbuhkan area-area dengan kepadatan yang cukup tinggi ke dalam *clusters* dan menemukan *clusters* dalam bentuk yang sembarang dalam suatu *database* spasial yang memuat *noise*. DBSCAN mendefinisikan *cluster* sebagai himpunan maksimum dari titik-titik kepadatan yang terkoneksi (*density-connected*). Semua objek yang tidak masuk ke dalam *cluster* manapun dianggap sebagai *noise* (Han, J., Kamber, M., dan Pei, J., 2012 dan Babur, I.H. et al., 2015).

DBSCAN menentukan sendiri jumlah *cluster* yang akan dihasilkan berdasarkan tingkat kerapatan (*density*) sehingga kita tidak perlu menentukan jumlah *cluster* yang diinginkan, tapi memerlukan 2 masukan lain, yaitu MinPts (minimal banyaknya *item* dalam suatu *cluster*) dan Eps (nilai untuk jarak antar

item yang menjadi dasar pembentukan *neighborhood* dari suatu titik *item*). *Neighborhood* yang terletak di dalam radius ($\in$) disebut $\in$-*neighorhood* dari objek data. Jika $\in$-*neighborhood* dari suatu objek berisi paling sedikit suatu angka yang minimum, MinPts dari suatu objek, objek tersebut disebut *core object*.

Neighborhood dari *border points* berisi jauh lebih sedikit *item* daripada *neighborhood* dari *core points*. Suatu *border point* bisa jadi termasuk ke dalam lebih dari 1 *core object* (Kanagala, H.K. dan Krishnaiah, J.R., 2016). Berikut ini gambar yang menunjukkan mana yang merupakan *border point* dan mana yang merupakan *core point*.



Gambar 2.3. *Core* dan *Border*

(Amini, A., Saboohi, H., dan Wah, T.Y., 2014)

Menurut definisi, terdapat 2 jenis titik (*points*) dalam suatu *cluster* yaitu di dalam *cluster* (*core points*) dan di tepian *cluster* (*border points*) dimana *neighborhood* dari *border points* berisi jauh lebih sedikit *items* daripada *neighborhood* dari *core points*. Suatu *border point* bisa jadi termasuk ke dalam lebih dari 1 *cluster* (Shah, G.H., 2012).

DBSCAN menemukan *clusters* dengan cara :

1. DBSCAN menelusuri *clusters* dengan memeriksa $\in$-*neighborhood* (Eps-*neighborhood*) dari tiap-tiap titik dalam *database*. Jika $\in$-*neighborhood* dari titik p mengandung lebih dari MinPts, *cluster* baru dengan p sebagai *core object* diciptakan.

2. Kemudian DBSCAN secara iteratif mengumpulkan secara langsung objek-objek *density-reachable* dari *core object* tersebut, dimana mungkin melibatkan penggabungan dari beberapa *clusters* yang *density-reachable*.

Kunci dari algoritma DBSCAN adalah bahwa untuk setiap titik dari sebuah *cluster*, *neighborhood* dari radius yang diberikan harus mengandung setidaknya jumlah minimum poin, yaitu, kepadatan *neighborhood* harus melebihi beberapa *threshold* yang telah ditetapkan (Ye, Gao and Zeng, 2003). Berikut adalah urutan algoritma DBSCAN secara umum :

1. *Arbitrary select a point p* (pilih titik p awal secara acak).
2. *Retrieve all points density-reachable from p wrt Eps and MinPts* (mengambil semua titik yang *density-reachable* terhadap titik p).
3. *If p is a core point, a cluster is formed* (Jika p adalah *core point* maka *cluster* terbentuk).
4. *If p is a border point, no points are density-reachable from p and DBSCAN visits the next point of the database* (Jika p adalah *border point*, tidak ada yang merupakan hubungan *density-reachable* dari p dan DBSCAN akan mengunjungi titik selanjutnya dari *database*).
5. *Continue the process until all of the points have been processed* (Melanjutkan proses sampai semua titik telah diproses).
6. *Result is independent of the order of processing the points* (hasil yang didapatkan tidak tergantung kepada urutan dari proses titik yang diambil).

Adapun kelebihan dari DBSCAN yaitu (Ahmad, P.H. dan Dang, S., 2015):

1. Mampu menemukan *cluster* sembarang.
2. Mampu menemukan *cluster* yang sama walaupun dikelilingi oleh *cluster* yang berbeda.
3. Kuat dalam mendeteksi *outlier* (*noise*).
4. Hanya membutuhkan dua titik yang tidak sensitif dengan urutan titik dalam *database*.

Selain kelebihan, terdapat juga kelemahan dari DBSCAN yaitu (Ahmad, P.H. dan Dang, S., 2015 dan Dharshini, D.A.P. et al., 2016) :

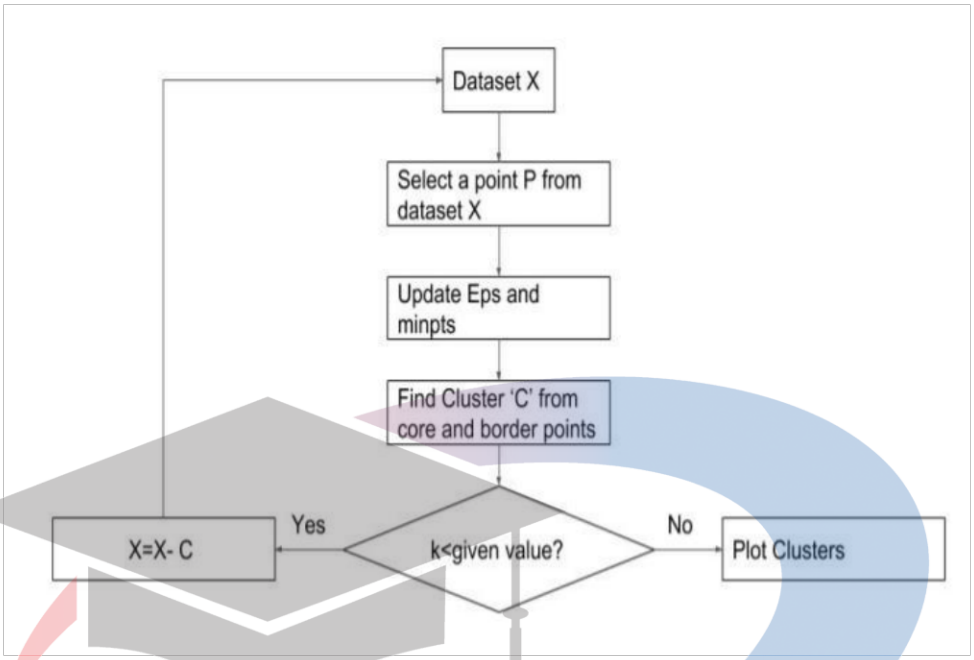
1. Parameter nilai Eps dan MinPts yang sensitif.
2. Tidak dapat dipartisi untuk sistem multi prosesor.
3. Kesulitan dalam menemukan *cluster* dengan kepadatan yang berbeda.
4. Gagal mengidentifikasi *cluster* jika kepadatannya bervariasi atau *dataset* yang terlalu jarang.
5. *Sampling* yang mempengaruhi ukuran kerapatan.

2.1.3.3. *Adaptive* DBSCAN

Untuk mengatasi kelemahan DBSCAN dalam menemukan *cluster* dalam *dataset* dengan berbagai kepadatan data, DBSCAN dimodifikasi sehingga dapat mengadaptasi nilai Eps dan MinPts pada distribusi kepadatan dari *cluster* serta mengurangi jumlah *cluster* dan outlier seiring dengan peningkatan iterasi nilai Eps (Zhou, H., Wang, P., dan Li, H., 2012). *Adaptive* DBSCAN dapat menentukan nilai Eps dan MinPts yang sesuai secara otomatis.

Adaptive DBSCAN pertama dimulai dengan nilai acak dari Eps. Jika terjadi kegagalan dalam mendeteksi *cluster*, nilai Eps akan ditingkatkan sebesar 0.5. Jika dalam iterasinya ditemukan kemiripan data melebihi 10%, maka dianggap bahwa *cluster* telah ditemukan. Kemudian *cluster* itu disimpan secara terpisah dan dikeluarkan dari *dataset* primer.

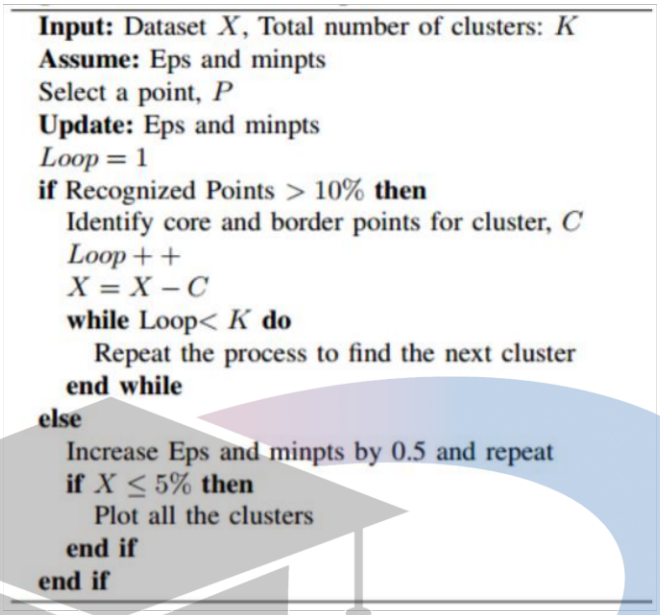
Algoritma *Adaptive* DBSCAN akan meningkatkan lagi nilai Eps dan MinPts untuk mendeteksi *cluster* berikutnya. Ketika 95% data telah selesai diproses oleh *Adaptive* DBSCAN, algoritma mengasumsikan bahwa semua *cluster* telah berhasil terdeteksi. Kemudian titik-titik yang tersisa akan dianggap sebagai titik *noise* atau *outlier*, dan *cluster* yang disimpan akan dilakukan *plotting*. Berikut adalah gambar *flowchart* cara kerja dari algoritma *Adaptive* DBSCAN.



Gambar 2.4. *Flowchart* Algoritma *Adaptive* DBSCAN

(Khan, M.M.R. et al., 2018)

Berbeda dengan DBSCAN, *Adaptive* DBSCAN membutuhkan predeterminasi dari jumlah *cluster* data dalam suatu *dataset*. Diberikan k = 1,2, ..., C. C = jumlah *cluster* data dalam *dataset*. Dimulai dengan *dataset* X dan nilai awal Eps, ε dan Minpts. Pertama, diambil titik sembarang P. Kemudian memperbarui ε dan Minpts untuk mendapatkan *cluster* terpadat C. Setelah didapatkan kandidat *cluster* terpadat, akan disimpan titik datanya dan titik-titik dari *dataset* tersebut akan dihapus. Kemudian dilakukan pembaharuan ε dan Minpts lagi untuk mendapatkan *cluster* padat berikutnya. Itulah cara untuk mendapatkan semua *cluster* yang memiliki kepadatan yang berbeda satu per satu (Khan, M.M.R. et al., 2018). Berikut adalah detail dari algoritma *Adaptive* DBSCAN.



Input: Dataset X, Total number of clusters: K
Assume: Eps and minpts
Select a point, P
Update: Eps and minpts
$Loop = 1$
if Recognized Points $> 10\%$ **then**
  Identify core and border points for cluster, C
  $Loop++$
  $X = X - C$
  while Loop$< K$ **do**
    Repeat the process to find the next cluster
  end while
else
  Increase Eps and minpts by 0.5 and repeat
  if $X \leq 5\%$ **then**
    Plot all the clusters
  end if
end if

Gambar 2.5. Algoritma *Adaptive* DBSCAN

(Khan, M.M.R. et al., 2018)

2.1.4. *Search Query*

Search query adalah dasar dihasilkannya pengetahuan mengenai apa yang menjadi objek pencarian dari pengguna, dengan dimotivasi oleh rasa keingintahuan, *time killing*, ataupun pembelajaran. *Query* disimpan dan diolah menjadi suatu kumpulan, kelompok, atau kategori yang dapat dipelajari lebih lanjut mengenai pola di balik *query* pengguna. Panjang karakter dari *search query* bervariasi sehingga tidak dapat secara langsung dijadikan sebagai bahan untuk diolah menjadi suatu pengetahuan. Kesulitan tersebut terletak pada bagian klasifikasi karena perlu dilakukan kombinasi pada *search query* sehingga metode yang digunakan dalam menghasilkan keluaran *search* mampu memberikan hasil yang adaptif (Taksa, I., Zelikovitz, S., dan Spink, A., 2016 dan Dan Wu, Man Zhu, dan Aihua Ran, 2016).

Terdapat tiga klasifikasi utama yang telah divalidasi secara empiris oleh *search engine query* dalam menjelaskan model dari hasil *search query*, yaitu (Jansen, B.J., Booth, D. and Spink, A., 2008) :

1. *Navigational query*, yaitu pengguna berharap *query* yang digunakan dapat membawanya ke halaman tunggal dari entitas tertentu, misalnya Youtube.
2. *Informational query*, yaitu pengguna berharap mendapatkan informasi sebanyak mungkin mengenai topik tertentu untuk menjawab kebutuhannya.
3. *Transactional query*, yaitu *query* yang mencerminkan maksud tertentu dari pengguna selain untuk mendapatkan informasi, seperti membeli barang atau mengunduh sesuatu.

Ketiga model tersebut cukup konsisten pada keseluruhan *search engine* dan mampu mencerminkan pola dari *query*, terutama *e-commerce* (Jansen, B.J., Booth, D. and Spink, A., 2008 dan Su, N. et al., 2018). Semakin akurat hasil *search* berdasarkan model *search query*, maka kepuasan pelanggan dan potensi transaksi juga akan semakin besar. *Insight* dari pengolahan *search query* ini adalah melakukan peningkatan terhadap cara kerja fitur *search* dan keakuratan prediksi yang mampu memberikan manfaat bagi pengguna. Manfaat tidak harus berujung kepada keberhasilan terjadinya transaksi, keberhasilan memberikan informasi hasil *search* yang akurat juga merupakan suatu manfaat bagi pengguna (Vakkari, P. et al., 2019).

Query pengguna terus mengalami perubahan sesuai dengan tren dan informasi dari hasil *query*. Relevansi keluaran hasil *query* mempengaruhi kepuasan pelanggan dan berimplikasi kepada tingkat konversi pelanggan. Kepentingan ini menjadi dasar dalam memprediksi keluaran *query* yang sesuai atau mirip (dalam *cluster* yang sama) dengan produk-produk yang ada di *e-commerce* (Lin YC., Datta, A., dan Fabbrizio, G.D., 2018 dan Sondhi, P. et al., 2018).

2.1.5. *Search Query Clustering*

Pemahaman mengenai perilaku *search* adalah aktivitas yang penting untuk melakukan optimisasi pada hasil keluaran *search engine*. Untuk memahaminya dibutuhkan pengolahan *search query* yang tersimpan di dalam *logs* atau *database*. Dalam penelitian yang dilakukan oleh Sondhi, P. et al., *search query* pada *e-*

commerce dibentuk menjadi taksonomi dengan menggunakan algoritma *k-means* untuk menghasilkan model dasar pengetahuan bisnis. Salah satu contoh model dasar pengetahuannya adalah sebagai faktor pendorong untuk mencari mitra yang menjual produk yang sering dicari oleh pengguna (Sondhi, P. et al., 2018 dan Singh, S. et al., 2018).

Untuk mendapatkan taksonomi, masing-masing *cluster* yang dihasilkan harus dihitung rataan dari nilai *click rate*, *add-to-cart ratio*, *order ratio*, *token*, dan *browsed page per query* dengan menggunakan rumus-rumus berikut :

1. *Click rate* = countclickperquery / impression.
2. *Add-to-cart ratio* = countatc / countclickperquery.
3. *Order ratio* = countorderfromatc / countatc.
4. *Token* = banyaknya token selama melakukan *query*.
5. *Browsed page per query* = banyaknya *page* yang dibuka selama melakukan *query*.

Berdasarkan perhitungan-perhitungan di atas, peneliti kemudian menentukan taksonomi-taksonomi berikut (Sondhi, P. et al., 2018) :

1. *Shallow exploration queries*

   Query ini merupakan inisial awal bagi pengguna dalam melakukan pencarian untuk mendapatkan inspirasi formulasi ulang *query*. Pengguna melakukan *browse* beberapa *top product* tanpa melakukan *click* lebih jauh sehingga *query* hanya berkaitan dengan beberapa kategori produk saja. *Shallow exploration* ini harus menghasilkan keluaran produk dan navigasi yang beragam sehingga pengguna bisa terbawa untuk melakukan eksplorasi lebih lanjut. Adapun tolak ukur dari *shallow exploration* adalah :

   a. Token rataan terkecil.
   b. *Click rate* terkecil.
   c. *Add-to-cart ratio* terkecil.
   d. *Order ratio* terkecil.

2. *Targeted purchase queries*

   Pengguna sudah familiar dengan *query* yang digunakan sehingga tidak membutuhkan waktu yang lama untuk membuat keputusan. Adapun tolak ukur dari *target purchase* adalah :

   a. *Click rate* terbesar.
   b. *Add-to-cart ratio* terbesar.
   c. *Order ratio* terbesar.

3. *Major-item shopping queries*

   Pengguna melakukan *query* atas produk-produk yang nominalnya mahal sehingga membutuhkan eksplorasi serius tetapi pilihannya terbatas. *Query* ini didominasi oleh kategori produk elektronik dan perabotan. Pengguna cenderung tidak melakukan penelusuran lebih mendalam, melainkan hanya berharap mendapatkan keluaran *query* yang terbatas dan membuat keputusan berdasarkan detail produk yang didapatkannya tersebut. Adapun tolak ukur dari *major-item shopping* adalah :

   a. *Click rate* terbesar.
   b. *Add-to-cart ratio* yang lebih kecil.
   c. *Order ratio* yang lebih kecil.

4. *Minor-item shipping queries*

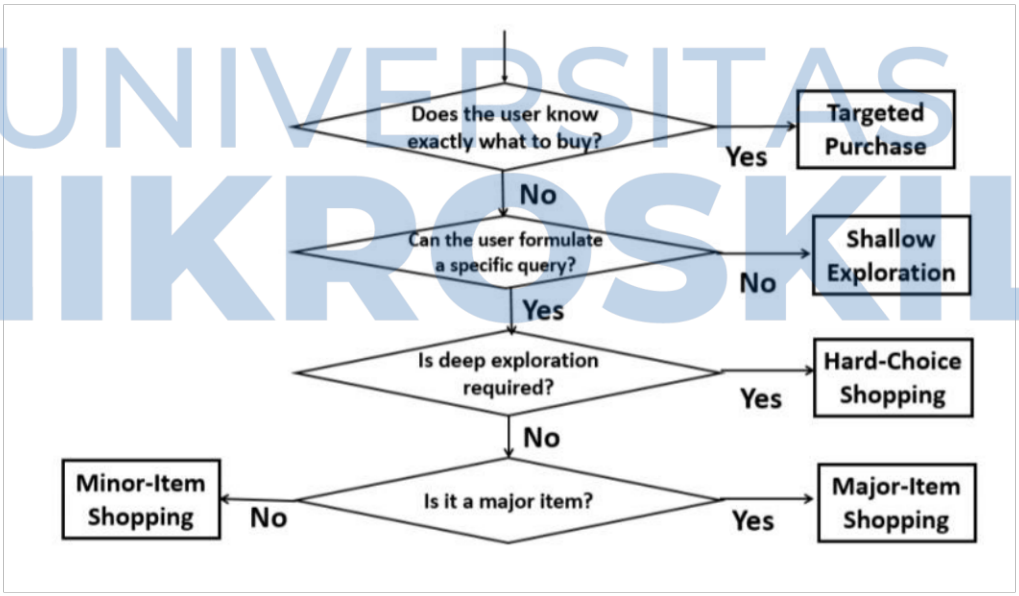
   Pengguna melakukan *query* atas produk-produk yang nominalnya tidak terlalu mahal tetapi masih membutuhkan eksplorasi. *Query* ini didominasi oleh produk-produk yang relatif tahan lama dan berkaitan dengan dapur, peralatan kamar mandi, dan dekorasi. Dalam jenis *query* ini, pengguna peduli terhadap aspek *stylist* dan estetika dari produk sehingga jika dibandingkan dengan *targeted purchase queries* cenderung lebih lama. Adapun tolak ukur dari *minor-item shopping* adalah :

   a. *Click rate* moderasi.
   b. *Add-to-cart ratio* moderasi.
   c. *Order ratio* moderasi.

5. *Hard-choice shopping queries*

   Pengguna melakukan eksplorasi yang mendalam terhadap keluaran *query* sebelum melakukan transaksi pembelian, dan dilakukan perbandingan dengan penuh kehati-hatian. *Query* ini didominasi oleh pakaian, aksesoris, dan dekorasi rumah karena banyak produk relevan yang dapat dipilih, pengguna bersedia menghabiskan waktu untuk eksplorasi, dan kesulitan pengguna dalam mengungkapkan spesifikasi kebutuhannya dalam *query*. Pengguna bersedia untuk melakukan penelusuran dalam *query* selama produk yang dicari belum ditemukan. Adapun tolak ukur dari *hard-choice shopping* adalah *browsed page per query* yang besar.

Taksonomi di atas digunakan sebagai dasar peningkatan fitur *search* dalam menghasilkan keluaran *query* yang lebih informatif untuk menjawab kebutuhan pengguna seperti *list*, *playlist*, dan istilah informatif lainnya (Kathuria, A. et al., 2010). Kondisi-kondisi yang menentukan *search query* dari pengguna masuk ke dalam taksonomi yang mana dapat dilihat pada gambar di bawah ini (Sondhi, P. et al., 2018).



Gambar 2.6. Taksonomi *Search Query*

(Sondhi, P. et al., 2018)

Berdasarkan hasil penelitian mengenai taksonomi di atas kemudian dibahas tingkat stabilitas dari hasil *clustering query*, tetapi belum membahas tingkat keakuratannya. Dengan menggunakan penelitian Sondhi, P. et al. sebagai referensi, peneliti-peneliti bernama Lin, YC., Datta, A., dan Fabbrizio, G.D. mengatakan bahwa *search query* pada *e-commerce* umumnya tidak terlalu panjang, ambigu, dan terus berubah seiring dengan aktivitas *search*, serta kekurangan data *training* merupakan suatu kesulitan dalam melakukan analisis *query*. Analisis mereka menggunakan SVM linear untuk melakukan *clustering* hasil *query* tetapi tidak melanjutkan penelitian Sondhi, P. et al. yang belum membuktikan tingkat keakuratannya (Lin, YC., Datta, A., dan Fabbrizio, G.D., 2018).

Berbeda dengan penelitian *clustering* menggunakan algoritma DBSCAN yang dilakukan oleh Meng, L. et al., penelitian mereka tidak menggunakan dataset *query*, melainkan hanya berfokus kepada akurasi dan kompleksitas algoritma (Meng, L. et al., 2015). Kesulitan DBSCAN dalam menentukan nilai Eps dan MinPts dijadikan sebagai dasar penelitian Dharshini, D.A.P. et al. menggunakan algoritma *adaptive* DBSCAN. Untuk setiap nilai Eps, dilakukan modifikasi terhadap nilai MinPts. Dengan berbagai nilai Eps dan Minpts, DBSCAN yang dimodifikasi dapat menemukan *cluster* dengan tingkat kepadatan yang berbeda. Hasil dari DBSCAN yang didapatkan, dilakukan pengelompokan kembali *noise* yang tersisa dengan nilai Eps dan MinPts dari *adaptive* DBSCAN untuk mencoba menemukan *cluster* lain. Ternyata kehadiran *adaptive* DBSCAN mampu menyelesaikan kelemahan DBSCAN tersebut (Dharshini, D.A.P. et al., 2016).

2.1.6. *Confusion Matrix*

Evaluasi terhadap suatu klasifikasi umumnya dilakukan menggunakan sebuah himpunan data uji, yang tidak digunakan dalam pelatihan klasifikasi tersebut, dengan suatu ukuran tertentu. Terdapat sejumlah ukuran yang dapat digunakan untuk menilai atau mengevaluasi model klasifikasi, diantaranya adalah: *accuracy* atau tingkat pengenalan, *error rate* atau tingkat kesalahan atau

kekeliruan klasifikasi, *recall* atau *sensitivity* atau *true positive*, *specificity* atau *true negative* dan *precision*.

Model klasifikasi yang dibuat adalah pemetaan dari suatu baris data dengan keluaran sebuah hasil prediksi kelas / target dari data tersebut. Klasifikasi yang memiliki dua kelas sebagai keluarannya disebut dengan klasifikasi biner. Kedua kelas tersebut biasa direpresentasikan dalam {0,1}, {+1,-1} atau {positive; negative} (Suyanto, 2019).

Empat istilah penting untuk memahami semua ukuran evaluasi adalah sebagai berikut :

1. TP atau *True Positive* adalah jumlah tuple positif yang dilabeli dengan benar oleh *classifier*. Yang dimaksud tuple positif adalah tuple aktual yang berlabel positif, seperti tuple dengan label Bonus=’Ya’.
2. TN atau *True Negative* adalah jumlah tuple negatif yang dilabeli dengan benar oleh *classifier*. Yang dimaksud tuple negatif adalah tuple aktual yang berlabel negatif, seperti tuple dengan label Bonus=’Tidak’.
3. FP atau *False Positive* adalah jumlah tuple negatif yang salah dilabeli oleh *classifier*. Misalnya, sebuah tuple pelanggan yang berlabel Bonus=’Tidak’ tetapi oleh classifier dilabeli Bonus=’Ya’.
4. FN atau *False Negative* adalah jumlah tuple positif yang salah dilabeli oleh *classifier*. Misalnya sebuah tuple pelanggan yang berlabel Bonus=’Ya’, tetapi oleh classifier dilabeli Bonus=’Tidak’.

Keempat istilah di atas dapat direpresentasikan sebagai matriks 2 x 2 yang disebut *confusion matrix* yang ditunjukkan oleh gambar 2.7 berikut.

		Aktual	
	Class	***Positive***	***Negative***
Prediksi	***Positive***	*True Positive (TP)*	*False Negative (FN)*
	Negative	*False Positive (FP)*	*True Negative (TN)*

Gambar 2.7. *Confusion Matrix*

(Suyanto, 2019)

Confusion matrix berguna untuk menganalisis kualitas *classifier* dalam mengenali tuple-tuple dari kelas yang ada. TP dan TN menyatakan bahwa *classifier* mengenali tuple dengan benar, artinya tuple positif dikenali sebagai positif dan tuple negatif dikenali sebagai negatif. Sebaliknya, FP dan FN menyatakan bahwa *classifier* salah dalam mengenali tuple, tuple positif dikenali sebagai negatif dan tuple negatif dikenali sebagai positif. Terdapat beberapa rumus umum yang dapat digunakan untuk menghitung performa klasifikasi. Hasil dari nilai akurasi, presisi, dan recall biasa ditampilkan dalam persentase.

a. *Accuracy*

Akurasi adalah jumlah proporsi prediksi yang benar. Adapun rumus penghitungan akurasi.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

b. *Precision*

Precision adalah proporsi jumlah dokumen teks yang relevan terkenali diantara semua dokumen teks yang terpilih oleh sistem. Rumus *precision*.

$$Precision = \frac{TP}{TP+FP}$$

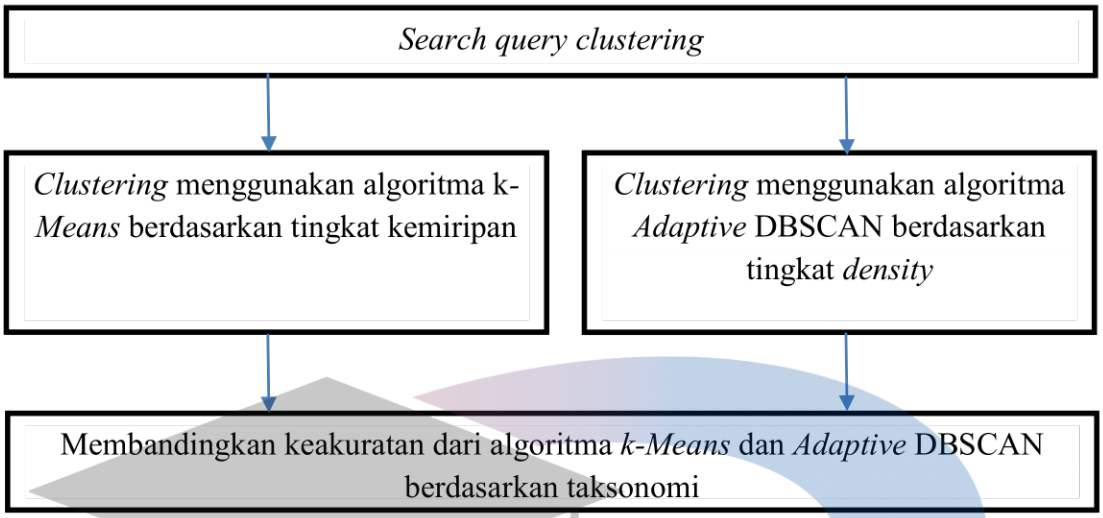
c. *Recall*

Recall adalah proporsi jumlah dokumen teks yang relevan terkenali diantara semua dokumen teks relevan yang ada pada koleksi. Rumus *recall*.

$$Recall = \frac{TP}{TP+FN}$$

2.2. Kerangka Konsep Pemecahan Masalah

Kerangka konsep pemecahan masalah menggambarkan bagaimana masalah penelitian dapat diselesaikan melalui solusi-solusi yang diusulkan serta dari solusi tersebut diharapkan memiliki dampak yang dapat menyelesaikan permasalahan penelitian. Berikut adalah gambaran kerangka konsep pemecahan masalah dari penelitian yang akan dilakukan :



Gambar 2.8. Kerangka Konsep Pemecahan Masalah

Search query clustering dilakukan menggunakan *dataset* yang berbeda dengan penelitian sebelumnya, tetapi menggunakan algoritma yang sama yaitu *k-means*, kemudian dilakukan hal yang sama dengan menggunakan *adaptive* DBSCAN. Hasil dari kedua algoritma tersebut akan diteliti keakuratannya berdasarkan taksonomi *shallow exploration queries*, *targeted purchase queries*, *major-item shopping queries*, *minor-item shopping queries*, dan *hard-choice shopping queries*.

Berdasarkan penelitian yang pernah dilakukan, *k-means* belum pernah diteliti tingkat keakuratannya dalam melakukan *clustering search query*, sementara *adaptive* DBSCAN hanya digunakan untuk menyelesaikan kelemahan dari DBSCAN dan bukan dalam ruang lingkup *clustering search query*. Oleh karena itu, Peneliti membandingkan keakuratan masing-masing algoritma tersebut dalam melakukan *search query clustering* berdasarkan taksonomi.