BAB II

KAJIAN TEORI

Bab ini berisi dasar-dasar teori tentang basis data (*database*), penambangan data (*data mining*), pengelompokan (*clustering*), algoritma *k-means*, klasifikasi, pohon keputusan, algoritma C4.5, WEKA, analisis penggunaan *e-learning*, BESMART dan penelitian yang relevan sebagai landasan pelaksanaan penelitian.

A. Basis Data (*Database*)

Basis data adalah sebuah gudang data, yang dirancang untuk mendukung penyimpanan, pengambilan, dan pemeliharaan data yang efisien. Basis data disediakan untuk memenuhi berbagai kebutuhan industri. Terdapat sistem untuk mengelola basis data yang dikenal sebagai *database management system* atau singkatnya DBMS. DBMS merupakan sebuah perangkat lunak yang mengontrol akses, mengatur, menyimpan, mengelola, mengambil dan menjaga data dalam basis data (Sharma, dkk, 2010: 23-24).

DBMS dapat diartikan sebagai program komputer yang digunakan untuk memasukkan, mengubah, menghapus, memodifikasi dan memperoleh data atau informasi dengan praktis dan efisien. Kelebihan dari DBMS antara lain adalah (Dzacko, 2007; 2-3) :

1. Kepraktisan

   DBMS menyediakan media penyimpan permanen yang berukuran kecil namun banyak menyimpan data jika dibandingkan dengan menggunakan kertas.

2. Kecepatan

   Komputer dapat mencari dan menampilkan informasi yang dibutuhkan dengan cepat.

3. Mengurangi kejemuan

   Pekerjaan yang berulang-ulang dapat menimbulkan kebosanan bagi manusia, sedangkan mesin tidak merasakannya.

4. *Update to date*

   Informasi yang tersedia selalu berubah dan akurat setiap saat.

Basis data merupakan sebuah model dari model pengguna terhadap aktivitas sebuah bisnis. Oleh karena itu untuk membangun sebuah basis data yang efektif tim pengembang harus mengetahui sepenuhnya model dari pengguna pada aktivitas bisnis tersebut. Ada dua cara umum untuk mengembangkan basis data yaitu :

1. *Top–down development*

   Pendekatan ini dimulai dari umum sampai detail. Dimulai dari mempelajari tujuan strategi sebuah organisasi, dimana untuk mencapai tujuan tersebut, informasi yang dibutuhkan oleh organisasi dapat dipenuhi oleh sistem. Dari langkah ini akan didapatkan sebuah perkiraan model data. Setelah didapatkan perkiraan tersebut tim pengembang akan bekerja menuju deskripsi

dan model yang lebih detail sehingga basis data yang diperlukan dapat diidentifikasi.

2. *Bottom–up development*

   Pendekatan ini berjalan sebaliknya, dimulai dari sistem tertentu. Tim pengembang kemudian akan mengumpulkan kebutuhan *input* dan *output* sistem berjalan, dengan menganalisis *form* dan *report* dari sistem manual dan melakukan wawancara dengan *user* untuk mengetahui apakah dibutuhkan *report, form, queries* atau kebutuhan baru lainnya. Jika sistem memiliki basis data maka tim akan menggunakan kebutuhan untuk membuat model data dan kemudian dari model data tersebut akan dibuat desain dan implementasi basis data.

   Pendekatan *bottom–up* lebih cepat dan tidak terlalu beresiko, meskipun tidak menghasilkan sistem yang terbaik tapi mampu menghasilkan sistem berguna secara cepat (Robby, Kwanentent & Wardana, 2009: 3).

Model data dalam basis data dibagi menjadi dua kelompok besar (Kusrini, 2007), yaitu:

1. Model data berbasis objek

   Model data berbasis objek merupakan himpunan data dan relasi yang menjelaskan hubungan logik antar data dalam suatu basis data berdasarkan objek data. Model data berbasis objek terdiri dari dua jenis, yaitu:

   a. Model hubungan entitas (*entity relationship model*)

      Model hubungan entitas (E-R) didasarkan atas persepsi terhadap dunia nyata yang terdiri dari sekumpulan objek, disebut entitas dan hubungan

antar objek tersebut. Entitas adalah objek di dunia yang bersifat unik. Setiap entitas mempunyai atribut yang membedakan dengan entitas lainnya. Contoh: entitas Mahasiswa, mempunyai atribut Nim, Nama, Alamat, dan Tanggal lahir.

b. Model berorientasi objek

Modelberorientasi objek berbasiskan kumpulan objek. Setiap objek berisi:

1) Nilai yang disimpan dalam variabel instan, dimana variabel melekat dengan objek itu sendiri.
2) Metode, operasi yang berlaku pada objek yang bersangkutan
3) Objek-objek yang memiliki tipe nilai dan metode yang dikelompokkan dalam satu kelas.
4) *Sending a message*, sebuah objek dapat mengakses data sebuah objek yang lain dengan memanggil metode dari objek tersebut

2. Model data berbasis *record*

Model data berbasis *record* mendasarkan pada *record* untuk menjelaskan kepada pengguna tentang hubungan logik antar data dalam basis data. Model data berbasis *record* terdiri dari tiga jenis, yaitu:

a. Model relasional

Model relasional menggunakan kumpulan tabel-tabel untuk merepresentasikan data dan relasi antar data-data tersebut. Setiap tabel terdiri atas kolom-kolom dan setiap kolom mempunyai nama yang unik.

b. Model hierarki

Model hierarki menyerupai pohon yang dibalik. Menggunakan pola hubungan orang tua-anak. Setiap simpul menyatakan sekumpulan medan. Simpul yang terhubung dengan level di bawahnya disebut orang tua. Setiap orang tua hanya bias mempunyai 1 anak, bisa banyak anak tetapi anak hanya mempunyai 1 orang tua. Simpul yang punya anak disebut akar, dan simpul yang tidak punya anak disebut daun. Hubungan antara orang tua dan anak disebut cabang.

c. Model jaringan

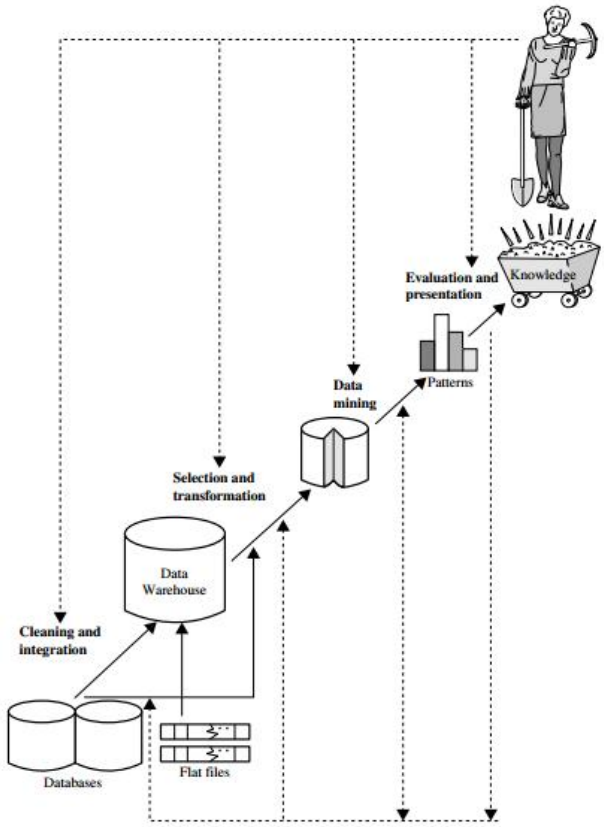
Data dalam model jaringan direpresentasikan dengan sekumpulan *record* dan relasi antar data direpresentasikan oleh *record* dan *link*. *Link* dipandang sebagai *pointer*. *Record-record* diorganisasikan sebagai graf. Model jaringan hampir sama dengan model hierarki. Perbedaannya pada model jaringan satu anak bisa mempunyai lebih dari 1 orang tua. Istilah orang tua dalam model jaringan disebut sebagai pemilik, sedangkan anak disebut sebagai anggota.

B. Penambangan Data (*Data Mining*)

Grup Gartner berpendapat (Larose, 2005: 2), *data mining* adalah sebuah proses menemukan korelasi baru, pola, dan tren bermakna pada penyimpanan data dengan jumlah besar, menggunakan teknologi pengenalan pola seperti statistik dan teknik matematika. Menurut Han, J., et al (2012: 8), *data mining* adalah proses menemukan pola dan pengetahuan dari data yang berjumlah besar. Sumber data merupakan hal dasar yang harus ada untuk dilakukan proses *data*

mining. Basis data pada penelitian ini berupa basis data dengan model relasional, dimana data berupa tabel-tabel yang terdiri dari sejumlah baris dan kolom yang menunjukkan atribut tertentu.

Data mining merupakan salah satu langkah penting untuk menemukan sebuah pengetahuan pada proses *Knowledge Discovery from Data* (KDD). KDD merupakan proses ekstraksi informasi yang berguna dan tidak diketahui sebelumnya dari sebuah kumpulan data (Bramer, 2007). Adapun proses KDD ditunjukkan pada Gambar 2.1.



Gambar 2.1 Tahap-tahap *Knowledge Discovery from Data* (KDD)

Proses dalam KDD secara runtut adalah sebagai berikut (Han, J., et al 2012: 6-8) :

1. Pembersihan Data (*Data Cleaning*)

   Pembersihan data bertugas untuk menghilangkan *noise* dan data yang tidak konsisten. Pada tahap ini dilakukan penghapusan pada data yang tidak memiliki kelengkapan atribut sesuai yang dibutuhkan.

2. Integrasi Data (*Data Integration*)

   Integrasi data merupakan proses kombinasi beberapa sumber data. Pada tahap ini dilakukan penggabungan data dari berbagai sumber untuk dibentuk penyimpanan data yang koheren.

3. Seleksi Data (*Data Selection*)

   Seleksi data merupakan proses pengambilan data yang berkaitan dengan tugas analisis dari basis data. Pada tahap ini dilakukan teknik perolehan sebuah pengurangan representasi dari data dan meminimalkan hilangnya informasi data. Hal ini meliputi metode pengurangan atribut dan kompresi data.

4. Transformasi Data (*Data Transformation*)

   Pada tahap ini data diubah dan dikonsolidasikan ke dalam bentuk yang sesuai untuk penambangan (*mining*) dengan melakukan ringkasan atau penggabungan operasi.

5. Penambangan Data (*Data Mining*)

   Data mining adalah inti pada proses KDD. *Data mining* adalah proses mencari pola atau informasi menarik dalam data terpilih dengan

menggunakan teknik tertentu. Pemilihan teknik dan algoritma yang tepat sangat bergantung pada tujuan dan proses KDD secara keseluruhan.

6. Evaluasi Pola (*Pattern Evaluation*)

   Tahap ini merupakan identifikasi kebenaran pola yang merupakan pengetahuan dasar pada langkah-langkah yang diberikan.

7. Representasi Pengetahuan (*Knowledge Presentation*)

   Pada tahap ini penemuan pengetahuan direpresentasikan secara visual kepada pengguna untuk membantu dalam memahami hasil *data mining* juga disertakan penyajian pengetahuan mengenai metode yang digunakan untuk memperoleh pengetahuan yang diperoleh pengguna.

Menurut Pramudiono (2003: 3) ada tiga teknik yang populer didalam *data mining,* yaitu:

1. Penambangan dengan Aturan Asosiatif (*Association rule mining*)

   Penambangan dengan aturan asosiatif adalah teknik mining untuk menemukan aturan assosiatif antara suatu kombinasi item. Aturan asosiatif digunakan untuk menunjukkan hubungan antara objek data. Aturan asosiatif memiliki dua langkah terpisah: Pertama, mencari *minimum support* yang diterapkan untuk menemukan semua pengulangan *itemset* dalam basis data. Kedua, melakukan pengulangan *itemset* dan menentukan batasan *minimum confidence* yang digunakan untuk membentuk aturannya (Aher & Lobo, 2012:1).

2. Klasifikasi (*Classification*)

   Klasifikasi adalah proses untuk menemukan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data, dengan tujuan untuk memperkirakan kelas dari suatu objek yang labelnya tidak diketahui. Model itu sendiri dapat berupa aturan jika-maka (*if-then)*, berupa pohon keputusan (*decision tree*), jaringan saraf tiruan (*neural network)*.

3. Pengelompokan (*Clustering*)

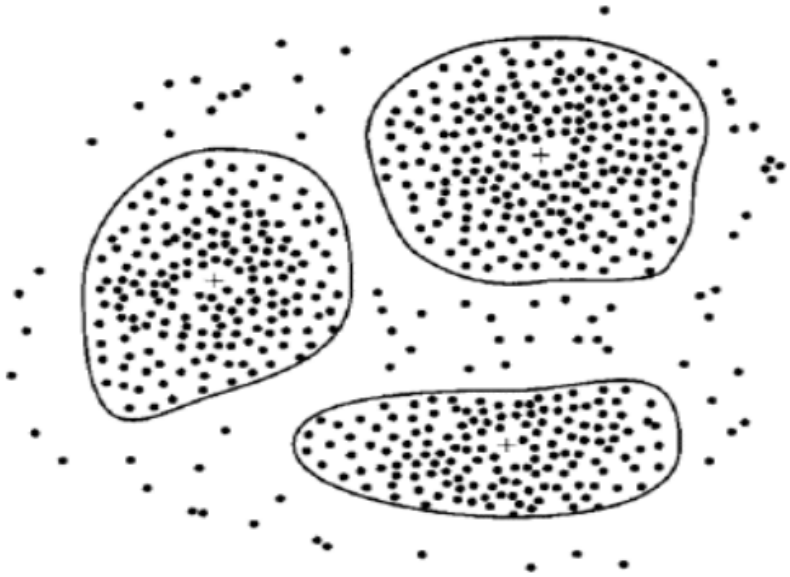
   Clustering melakukan pengelompokan data tanpa berdasarkan kelas data tertentu. *Clustering* dapat dipakai untuk memberikan label pada kelas data yang belum diketahui. Prinsip dari *clustering* adalah memaksimalkan kesamaan antar anggota satu kelas dan meminimumkan kesamaan antar kelas.

C. Pengelompokan (*Clustering*) dengan Algoritma *K-Means*

Clustering adalah proses pengelompokan satu set objek data ke dalam beberapa kelompok atau *cluster* sehingga objek dalam sebuah *cluster* memiliki jumlah kemiripan yang tinggi, tetapi sangat berbeda dengan objek di *cluster* lain. Ketidakmiripan dan kesamaan dinilai berdasarkan nilai atribut yang menggambarkan objek dan sering melibatkan perlakuan jarak. *Clustering* sebagai alat *data mining* dapat diterapkan pada berbagai bidang, seperti biologi, keamanan, intelijen bisnis, dan pencarian web (Han, J., 2012:443).

Prinsip dari *clustering* adalah memaksimalkan kesamaan antar anggota satu kelas dan meminimumkan kesamaan antar *cluster*. *Clustering* dapat dilakukan pada data yang memiliki beberapa atribut yang dipetakan sebagai ruang

multidimensi. Ilustrasi dari *clustering* dapat dilihat di Gambar 2.2 dimana lokasi, dinyatakan dengan bidang dua dimensi, dari pelanggan suatu toko dapat dikelompokan menjadi beberapa *cluster* dengan pusat *cluster* ditunjukkan oleh tanda positif (+). Beberapa algoritma pada teknik *clustering* menggunakan perhitungan jarak minimum untuk mengukur kemiripan antar data apabila data berupa data numeric (Kusnawi, 2007: 6).



Gambar 2.2 Contoh klasterisasi

Menurut Jiawei Han (2006: 401-430) secara umum metode pada *clustering* dapat digolongkan ke dalam beberapa metode berikut:

1. Metode partisi (*Partitioning Method*)

Langkah kerja metode partisi, yaitu apabila terdapat basis data sejumlah *n* objek atau data tupelo, selanjutnya data di partisi menjadi *k* partisi dari data, dimana setiap partisi mewakili sebuah *cluster* dan $k \leq n$. Adapun syarat yang harus terpenuhi sebagai berikut: (1) setiap kelompok harus berisi setidaknya satu objek, dan (2) setiap objek harus memiliki tepat satu kelompok.

Awalnya basis data dipartisi menjadi *k* partisi. Kemudian menggunakan teknik relokasi berulang, mencoba untuk memperbaiki partisi dengan memindahkan dari satu kelompok ke kelompok lain. Kriteria umum dari partisi yang baik adalah bahwa objek dalam satu *cluster* memiliki kemiripan yang sangat dekat, sedangkan objek dalam *cluster* yang berbeda memiliki kemiripan yang jauh berbeda.

Pencapaian optimalitas global dalam pengelompokan berbasis partisi akan memerlukan penghitungan lengkap dari semua partisi yang memungkinkan. Sebaliknya, sebagian besar aplikasi mengadopsi salah satu dari beberapa metode heuristik yang popular, seperti (1) algoritma *k-means,* dimana setiap segmen diwakili oleh nilai rata-rata dari objek dalam *cluster*, dan (2) algoritma *k-medoids*, dimana setiap segmen diwakili oleh salah satu objek yang terletak didekat *centroid*. Metode pengelompokan heuristik ini bekerja dengan baik untuk menemukan *cluster* berbentuk bola kecil untuk basis data yang berukuran sedang.

2. Metode Hirarki (*Hierarchi Method*)

Metode hirarki menciptakan dekomposisi hirarki dari himpunan objek data yang diberikan. Sebuah metode hirarki dapat diklasifikasikan sebagai salah satu *agglomerative* atau memecah belah, berdasarkan cara dekomposisi hirarki terbentuk. Pendekatan *agglomerative* memiliki dua cara pendekatan, yaitu *bottom–up* dan *top-down*. Pendekatan *bottom-up* berlangsung seperti berikut, awalnya setiap objek membentuk kelompok tersendiri. Berturut-turut menggabungkan objek atau kelompok yang dekat satu sama lain, sampai semua kelompok digabung menjadi satu (tingkat teratas dari hirarki), atau sampai terjadinya kondisi pemutusan hubungan. Sedangkan pendekatan *top-down*, dimulai dengan semua objek dalam *cluster* yang sama dibagi menjadi kelompok yang lebih kecil, sampai akhirnya setiap objek dalam satu *cluster* atau sampai terjadi kondisi pemutusan hubungan.

Metode hirarki memuat fakta bahwa setelah langkah penggabungan atau *split* dilakukan, proses memecah belah tidak dapat dibatalkan. Ada dua pendekatan untuk meningkatkan kualitas pengelompokan hirarki: (1) melakukan analisis yang cermat terhadap objek "*linkage*" pada setiap partisi hirarki, seperti di Chameleon, atau (2) mengintegrasikan *aglomerasi* hirarki dan pendekatan-pendekatan lain dengan terlebih dahulu menggunakan algoritma *agglomerative* hirarki objek kelompok ke dalam *microclusters*, dan kemudian melakukan *macroclustering* pada *microclusters* menggunakan metode pengelompokan lain seperti relokasi berulang. Salah satu algoritma yang tergolong kedalam metode hirarki yaitu BIRCH (*Balanced Iterative*

Reducing and Clustering Using Hierarchies). BIRCH merupakan salah satu algoritma pengelompokan hirarki yang terintegrasi. BIRCH memperkenalkan dua konsep, *clustering feature* dan *clustering feature tree* (CF tree), yang mana digunakan untuk menggambarkan ringkasan *cluster*.

3. Metode Berbasis Kerapatan (*Density Based Method*)

Sebagian besar metode *cluster* mempartisi objek berdasarkan jarak antara objek. Metode semacam itu hanya dapat menemukan *cluster* berbentuk bola dan mengalami kesulitan dalam menemukan *cluster* berbentuk sembarang. Metode pengelompokan lain telah dikembangkan berdasarkan gagasan kerapatan. Ide secara umumnya adalah terus tumbuhnya *cluster* yang diberikan selama densitas (jumlah objek atau pusat massa) di "*neighborhood* (lingkungan)" melebihi ambang batas tertentu, yaitu untuk setiap titik data dalam *cluster* tertentu, lingkungan radius tertentu setidaknya harus memuat minimal jumlah titik. Metode tersebut dapat digunakan untuk menyaring *outlier* dan menemukan bentuk kelompok sembarang.

Beberapa algoritma yang termasuk kedalam metode berbasis kerapatan, yaitu: DBSCAN, OPTICS, DENCLUE. DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) merupakan algoritma yang memperluas wilayah dengan kepadatan yang tinggi ke dalam cluster dan menempatkan *cluster* irregular pada basis data spasial dengan *noise*. Algoritma ini mendefiniskan *cluster* sebagai kumpulan maksimal dari titik-titik kepadatan yang terkoneksi. OPTICS (Ordering Points To Identify the Clustering Structure) merupakan algoritma pada metode hirarki yang diusulkan untuk

mengatasi kesulitan *user* dalam menentukan parameter yang digunakan untuk menemukan *cluster* yang bisa diterima. DENCLUE (*Density Based Clustering*) merupakan algoritma *clustering* yang berdasarkan suatu set fungsi distribusi kerapatan.

4. Metode berbasis *Grid*

   Metode berbasis *grid* mengkuantisasi ruang objek kedalam jumlah sel terbatas yang membentuk struktur jaringan. Semua operasi pengelompokan dilakukan pada struktur jaringan (yaitu, pada ruang terkuantisasi). Keuntungan utama dari pendekatan ini adalah waktu proses yang cepat, yang biasanya tergantung pada jumlah data objek dan bergantung hanya pada jumlah sel dalam setiap dimensi dalam ruang terkuantisasi. Algoritma yang termasuk kedalam metode berbasis *grid* diantaranya: STING, Wave Cluster, dan lainnya. STING (*Statistical Information Grid*) merupakan algoritma *clustering* yang bekerja dengan membagi daerah spatial menjadi sel-sel *rectanguleri*. Wave Cluster merupakan algoritma *clustering* yang melakukan sumarisasi data yang dilakukan dengan menentukan sruktur grid multidimensional terhadap *space* data.

5. Metode Berbasis Model

   Metode berbasis model membuat hipotesis sebuah model untuk masing-masing kelompok dan menemukan yang terbaik sesuai data yang diberikan model. Algoritma berbasis model dapat menemukan *cluster* dengan membangun fungsi kerapatan yang menggambarkan distribusi spasial titik data. Hal ini menyebabkan cara otomatis untuk menentukan jumlah *cluster*

berdasarkan standar statistik, membawa "*noise*" atau *outlier* kedalam perhitungan dan dengan demikian menghasilkan metode pengelompokan yang kuat. Algoritma yang termasuk ke dalam metode berbasis model yaitu COBWEB dan SOM. COBWEB adalah algoritma pembelajaran konseptual yang melakukan analisis probabilitas dan mengambil konsep sebagai model untuk *cluster*. SOM (mengorganisir diri berfitur peta) adalah algoritma berbasis jaringan saraf yang *cluster*nya dengan memetakan data dimensi tinggi ke peta fitur 2D atau 3D, yang juga berguna untuk visualisasi data.

Secara sederhana, *clustering* dapat dikonsentrasikan pada jarak *Euclidean* antar *record*:

$$d_{Euclidean}(x,y) = \sqrt{\sum_i (x_i - y_i)^2} \tag{2.1}$$

dimana $x = x_1, x_2, \dots, x_m$, dan $y = y_1, y_2, \dots, y_m$ yang melambangkan nilai atribut *m* dari dua catatan. Fungsi perhitungan matrik lainnya juga ada, seperti jarak *city-block*:

$$d_{cityblock}(x,y) = \sum_i |x_i - y_i| \tag{2.2}$$

atau jarak *Minkowski*, yang merupakan kasus umum dari dua metrik sebelumnya untuk eksponen *q* secara umumnya:

$$d_{Minkowski}(x,y) = \sum_i |x_i - y_i|^q \tag{2.3}$$

untuk kategori atribut, dapat didefinisikan "berbeda dari" fungsi untuk membandingkan nilai atribut ke *i* dari sepasang catatan.:

$$different\ (x_i, y_i) = \begin{cases} 0 & jika\ x_i = y_i \\ 1 & untuk\ yang\ lainnya \end{cases} \tag{2.4}$$

dimana x_i dan y_i adalah nilai kategorik. Kemudian dapat mengganti $different\ (x_i, y_i)$ untuk *i,* istilah ini dalam metrik jarak *Euclidean* diatas.

Perfoma yang optimal dari algoritma *clustering* sama seperti algoritma klasifikasi. Algoritma ini membutuhkan data yang akan dinormalisasi sehingga tidak ada variabel tertentu atau bagian dari variabel yang mendominasi analisis. Analisis dapat menggunakan salah satu dari *min-max* normalisasi atau standar *Z-Score*.

$$min - \max normalization: X^* = \frac{X - \min(X)}{Range(X)} \quad (2.5)$$

$$Z - score\ standardization: X^* = \frac{X - mean(X)}{SD(X)} \quad (2.6)$$

Secara umum teknik *clustering* memiliki tujuan pada identifikasi kelompok data dimana kesamaan dalam suatu kelompok data sangat tinggi sedangkan kesamaan dengan kelompok data lain sangat rendah (Larose, 2005:148-149).

Algoritma *k-means* adalah algoritma klasik untuk menyelesaikan masalah *clustering*, yang relatif sederhana dan cepat (Zhang & Fang, 2013: 193). Algoritma *k-means clustering* lebih sering dikenal karena kemampuannya dalam mengelompokkan data dalam jumlah besar dengan cepat dan efisien. Algoritma *k-mean* sangat rawan dipusat-pusat *cluster* awal, karena pusat *cluster* awal diproduksi secara acak. Algoritma *k-means* tidak menjanjikan hasil pengelompokan yang khas. Efisiensi keaslian algoritma *k-mean* sangat bergantung pada titik pusat *cluster* (*centroid*) awal (Yedla, et al, 2010: 121-122). Langkah kerja dari algoritma *k-means* adalah sebagai berikut :

1. Menanyakan kepada pengguna berapa banyak *k cluster dataset* yang akan dipartisi.
2. Menetapkan secara acak *k record* yang menjadi lokasi pusat *cluster* awal.

3. Setiap *record* dicari *centroid cluster* terdekatnya. Artinya setiap *centroid cluster* "memiliki" subset dari *record*, sehingga merepresentasikan sebuah partisi dari *dataset*. Didapatkan *k* cluster, $C_1, C_2, \ldots, C_k$.
4. Setiap *k cluster* dicari *centroid*nya dan memperbarui lokasi setiap pusat *cluster* untuk nilai *centroid* baru.
5. Ulangi langkah 3 sampai 5, sampai terjadi konvergensi atau terjadi penghentian.

Algoritma berakhir ketika titik pusat *cluster* tidak lagi berubah. Dengan kata lain, algoritma berakhir ketika dari seluruh *cluster* $C_1, C_2, \ldots, C_k$, semua *record* yang dimiliki oleh masing-masing pusat *cluster* tetap dalam *cluster* itu. Atau, algoritma dapat berhenti ketika beberapa kriteria konvergensi terpenuhi, seperti ada penyusutan yang tidak signifikan dalam jumlah kuadrat *error* (*sum of squared errors*):

$$SSE = \sum_{i=1}^{k} \sum_{p \in C_i} d\,(p, m_i)^2 \tag{2.7}$$

dimana $p \in C_i$ melambangkan setiap titik dalam *cluster i* dan m_i merupakan pusat *cluster i* (Larose, 2005:153).

Berikut contoh permasalahan yang menerapkan algoritma *k-means* untuk menyelesaikannya. Terdapat 4 jenis obat (A, B, C, D) dan setiap jenis obat memiliki dua atribut, seperti yang ditunjukkan pada Tabel 2.1. Tujuan dari kasus ini adalah untuk mengelompokkan jenis obat-obat tersebut menjadi *k*=2 kelompok berdasarkan dua fitur (pH dan indeks berat badan).

Tabel 2.1 Contoh kasus *clustering*

Objek	Atribut 1 (X): indeks berat badan	Atribut 2 (Y): pH
Obat A	1	1
Obat B	2	1
Obat C	4	3
Obat D	5	4

Dengan demikian, diketahui bahwa objek memiliki dua kelompok obat (*cluster* 1 dan *cluster* 2). Masalahnya sekarang adalah setiap obat berada pada *cluster* yang mana, apakah *cluster* 1 atau *cluster* lainnya. Setiap obat merepresentasikan satu titik dengan dua komponen koordinat. Dasar algoritma *k-means clustering* tergolong sederhana seperti yang ditunjukkan pada Gambar 2.3.

Algoritma: k-means. Algoritma untuk partisi, dimana setiap pusat cluster diwakili oleh nilai rata-rata objek dalam cluster.
Input:
- *k: jumlah cluster,*
- *D: satu set data yang berisi n objek.*

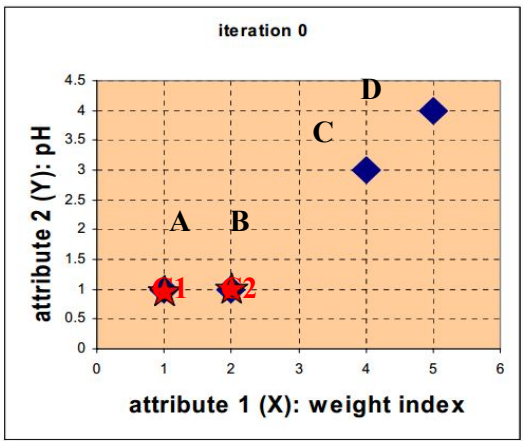
Output: satu set k cluster
Metode:
(1). *Ambil secara acak k objek dari D sebagai pusat cluster awal;*
(2). *Ulangi*
(3). *Kembali menetapkan setiap objek ke kelompok dengan objek yang paling mirip, berdasarkan nilai rata-rata objek dalam cluster;*
(4). *Memperbarui rata-rata cluster, yaitu dengan menghitung nilai rata-rata objek dalam cluster;*
(5). *Lakukan sampai tidak terjadi perubahan;*

Gambar 2.3 Algoritma *K-Means*

Uraian algoritma *k-means* yang ditunjukkan pada Gambar 2.3 di atas adalah sebagai berikut:

1. Tentukan jumlah *cluster k* pada *D* dan dapat diasumsikan sebagai *centroid* awal
2. Menghitung jarak masing-masing objek ke *centroid*
3. Mengelompokkan objek berdasarkan jarak minimum objek ke *centroid*
4. Menentukan *centroid* kembali apabila terjadi perpindahan kelompok dengan menghitung nilai rata-rata dari seluruh anggota pada masing-masing kelompok
5. Ulangi langkah 1 sampai 4 sampai tidak terjadi perpindahan kelompok lagi.

Berdasarkan pada kasus contoh diatas, jenis obat direpresentasikan oleh satu titik, dimana satu titik tersebut mengandung dua fitur, yang dapat dikatakan koordinat dalam ruang fitur, seperti yang ditunjukkan pada Gambar 2.4 berikut.



Gambar 2.4 Ruang Fitur pada Iterasi awal

Berikut penyelesaian contoh kasus di atas:

- Iterasi Awal (iterasi 0)

1. Menentukan nilai awal untuk *centroid*. Obat A dan obat B sebagai *centroid* pertama dan dilambangkan berturut dengan c_1 dan c_2 sebagai koordinat *centroid,* maka $c_1 = (1,1)$ dan $c_2 = (2,1)$
2. Menentukan jarak *centroid* objek, artinya menghitung jarak antara *centroid cluster* untuk setiap objek. Hal ini dapat dilakukan dengan menggunakan rumus jarak *Euclidean* yang kemudian akan didapatkan matriks jarak pada iterasi 0 (d^0) sebagai berikut:

   Bentuk matrik dari data objek,

$$\begin{matrix} A & B & C & D \end{matrix}$$
$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

   Bentuk matrik dari nilai jarak yang diperoleh pada iterasi 0,

$$d^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \text{ dengan } \begin{matrix} c_1 = (1,1) & kelompok\ 1 \\ c_2 = (2,1) & kelompok\ 2 \end{matrix}$$

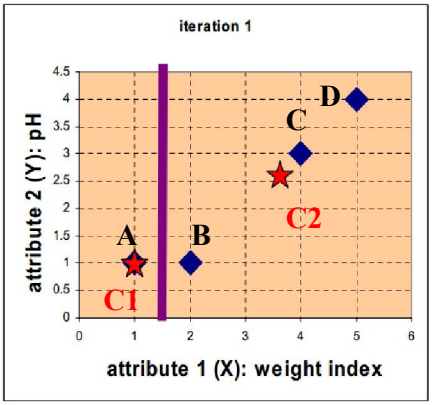
   Setiap kolom dalam matriks jarak melambangkan objek. Baris pertama dari matriks jarak sesuai dengan jarak masing-masing objek ke *centroid* pertama dan baris kedua adalah jarak dari setiap objek ke *centroid* kedua. Misalnya, jarak dari obat *C = (4,3)* ke *centroid* pertama $c_1 = (1,1)$ adalah $\sqrt{(4-1)^2 + (3-1)^2} = 3.61$ dan jarak ke *centroid* kedua c_2 adalah $\sqrt{(4-2)^2 + (3-1)^2} = 2.83$, dan seterusnya.
3. Langkah selanjutnya adalah pengelompokan objek. Awalnya menetapkan keberadaan setiap objek berdasarkan jarak minimum. Dengan demikian,

obat A ditetapkan ke kelompok 1, obat B ke kelompok 2, obat C pada kelompok 2 dan obat-obatan D ke kelompok 2. Unsur matriks dari kelompok matriks dibawah ini bernilai 1 jika dan hanya jika objek ditugaskan ke kelompok itu.

$$G^0 = \begin{matrix} A & B & C & D \\ \end{matrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} kelompok\ 1 \\ kelompok\ 2 \end{matrix}$$

- Iterasi ke 1

1. Langkah ini dilakukan dengan cara menentukan *centroid*nya terlebih dahulu. Setelah mengetahui anggota masing-masing kelompok, kemudian menghitung *centroid* baru dari masing-masing kelompok berdasarkan keanggotaan baru. Kelompok 1 hanya memiliki satu anggota sehingga *centroid*nya tetap, yaitu c_1= *(1,1)*. Kelompok 2 sekarang memiliki 3 anggota, sehingga *centroid*nya adalah rata-rata dari koordinat tiga anggota tersebut. $c_2 = \left(\frac{2+4+5}{3}, \frac{1+3+4}{3}\right) = \left(\frac{11}{3}, \frac{8}{3}\right)$. Perubahan posisi *centroid* dapat dilihat pada Gambar 2.5.



Gambar 2.5 Ruang Fitur untuk Iterasi ke 1

2. Iterasi 1 pada jarak *centroid* objek. Langkah selanjutnya adalah menghitung jarak dari semua objek ke *centroid* baru. Serupa dengan langkah 2, didapatkan matriks jarak pada iterasi 1 yaitu

$$d^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & kelompok\ 1 \\ c_2 = (\frac{11}{3}, \frac{8}{3}) & kelompok\ 2 \end{matrix}$$

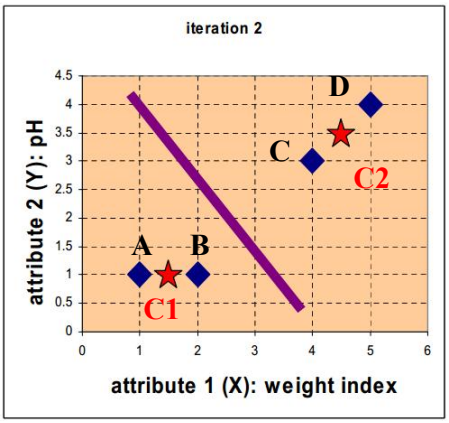
3. Pengelompokan objek (*object clustering*), menempatkan setiap objek berdasarkan jarak minimum. Berdasarkan matriks jarak yang baru, kita dapat memindahkan obat B ke kelompok 1 sedangkan semua objek lainnya tetap. Bentuk matriksnya ditunjukkan sebagai berikut:

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$G^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} kelompok\ 1 \\ kelompok\ 2 \end{matrix}$$

- Iterasi ke 2

1. Menghitung koordinat *centroid* baru berdasarkan pengelompokan iterasi sebelumnya. Kelompok 1 dan kelompok 2, keduanya memiliki dua anggota, sehingga didapatkan *centroid* baru $c_1 = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = \left(1\frac{1}{2}, 1\right)$ dan $c_2 = \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = \left(4\frac{1}{2}, 3\frac{1}{2}\right)$. Posisi *centroid* baru dan partisi kelompok ditunjukkan pada Gambar 2.6.



Gambar 2.6 Ruang Fitur untuk Iterasi ke 2

2. Menghitung jarak dari semua objek ke *centroid* baru maka didapatkan matriks jarak baru pada iterasi ke 2 ini seperti berikut:

$$d^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} c_1 = (1\frac{1}{2}, 1) & kelompok\ 1 \\ c_2 = (4\frac{1}{2}, 3\frac{1}{2}) & kelompok\ 2 \end{matrix}$$

3. Pengelompokan objek, menempatkan setiap objek berdasarkan jarak minimum. Pada iterasi kedua ini diperoleh matriks kelompok sebagai berikut:

$$\begin{matrix} & A & B & C & D \\ G^2 = & \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} & \begin{matrix} kelompok\ 1 \\ kelompok\ 2 \end{matrix} \end{matrix}$$

Hasil yang diperoleh bahwa $G^1 = G^2$. Dengan membandingkan pengelompokan hasil iterasi terakhir, iterasi ini menunjukkan bahwa objek tidak berpindah kelompok lagi. Dengan demikian, perhitungan pengelompokan *k*-

means telah mencapai stabilitas dan tidak diperlukan iterasi lagi. Sehingga didapatkan hasil akhir sebagai berikut (Teknomo, 2007; 1-5):

Tabel 2.2 Hasil iterasi terakhir

Objek	Atribut 1 (X): indeks berat badan	Atribut 2 (Y): pH	Kelompok (hasil)
Obat A	1	1	1
Obat B	2	1	1
Obat C	4	3	2
Obat D	5	4	2

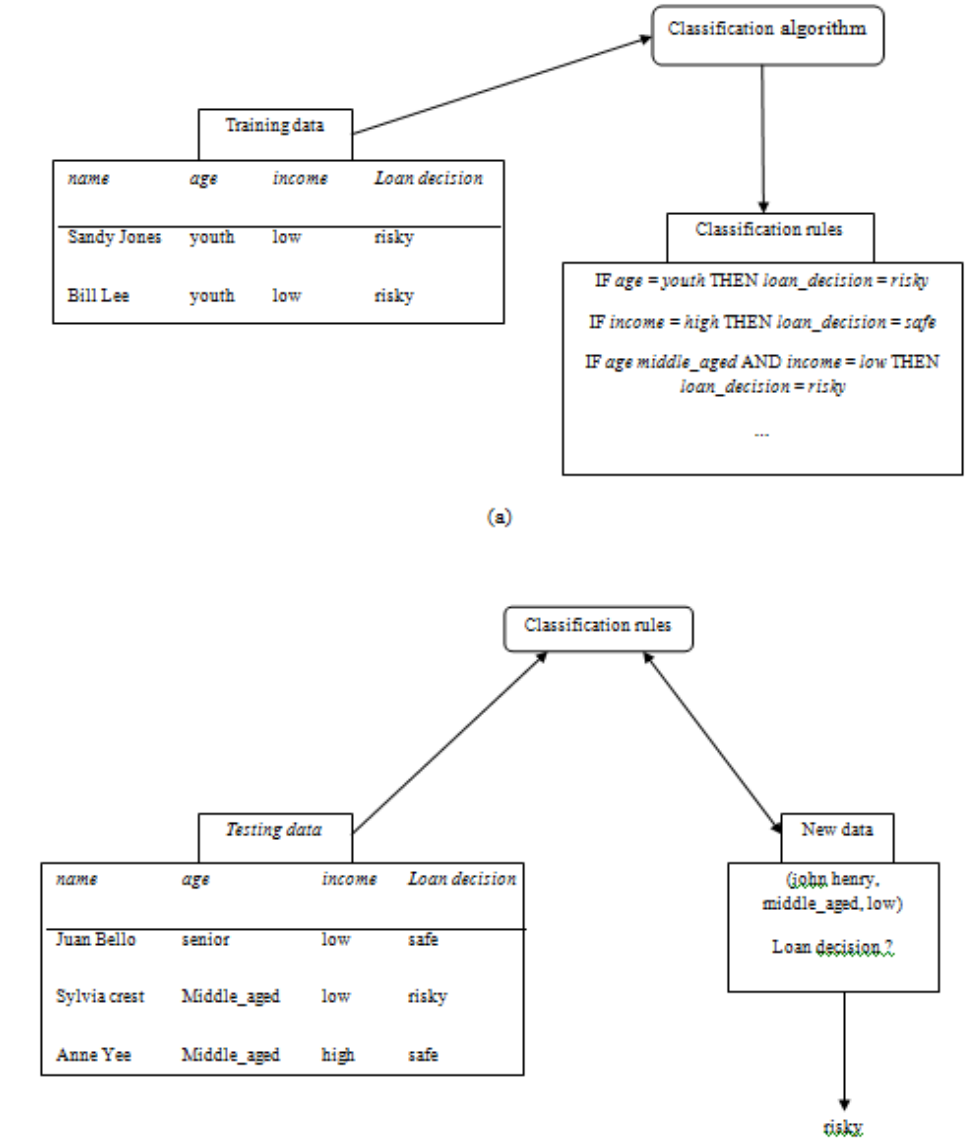
D. Klasifikasi (*Classification*) dengan Algoritma C4.5

Klasifikasi merupakan salah satu teknik *data mining* dengan melihat pada kelakuan dan atribut dari kelompok yang telah didefinisikan (Kusnawi, 2007). Klasifikasi menganalisis data dengan mengekstrasi model ke dalam kelas-kelas data. Cara kerja klasifikasi dibedakan menjadi dua proses, yang terdiri proses pembelajaran dan proses klasifikasi. Proses pembelajaran merupakan proses dimana model klasifikasi dibentuk atau biasa disebut proses *data training*. Proses klasifikasi merupakan proses dimana model yang didapatkan digunakan untuk memprediksi label kelas untuk data yang diberikan, yang biasanya disebut proses *data testing*.

Proses klasifikasi dapat dicontohkan seperti yang ditunjukkan pada Gambar 2.7 pada halaman 34. Kasus yang diilustrasikan pada Gambar 2.7 merupakan permasalahan yang dialami seorang petugas peminjaman dari sebuah bank. Petugas tersebut perlu melakukan analisis pada data yang diperoleh dari

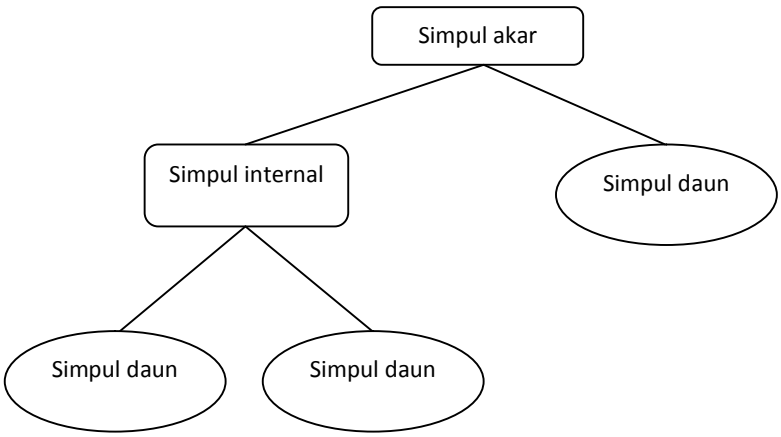
konsumennya untuk mengetahui peminjam tersebut aman (*safe*) atau beresiko (*risky*) untuk memperoleh pinjaman dari bank tersebut. Petugas tersebut menganalisis didasarkan atas dua atribut, yaitu umur (*age*) dan pendapatan (*income*). Atribut umur dikelaskan menjadi 3 kelas, yaitu muda (*youth*), sedang (*middle age*), tua (*senior*). Sedangkan atribut pendapatan dikelaskan menjadi dua kelas, yaitu rendah (*low*) dan tinggi (*high*). Awalnya dilakukan *training data* yang dilanjutkan dengan proses klasifikasi dengan algoritma yang digunakan. Setelah dihasilkan *rules* selanjutnya hasil tersebut digunakan untuk memprediksi pada data baru.

Pada Gambar 2.7 terdapat poin (a) dan (b), dimana (a) menunjukkan proses pembelajaran (*learning)* dan (b) klasifikasi. Proses klasifikasi digunakan untuk mengestimasi keakurasian dari *classification rules* yang dihasilkan. Apabila akurasi dapat diterima maka aturan yang diperoleh dapat digunakan pada klasifikasi data baru. Teknik klasifikasi pada *data mining* yang umum digunakan yaitu pohon keputusan (*decision tree*) (Han, J., et al, 2012: 327 - 330).



Gambar 2.7 Contoh Konsep Kerja Klasifikasi

Rokach dan Maimon (2008: 8) menyatakan bahwa pohon keputusan merupakan pengklasifikasian yang dinyatakan sebagai partisi rekrusif dari ruang contoh. Pohon keputusan dibangun oleh beberapa simpul, yaitu simpul akar (*node*), simpul internal (*test*), simpul daun (*leaves*).



Gambar 2.8 Bentuk Pohon Keputusan

Berdasarkan Gambar 2.8 terdapat tiga jenis simpul yang membangun sebuah pohon keputusan (Rokach dan Maimon, 2008: 27):

1. Simpul akar

   Simpul akar merupakan simpul yang tidak memiliki *input* akan tetapi memiliki *output*. Simpul ini dapat memiliki *output* lebih dari satu yang akan menjadi cabang.

2. Simpul internal

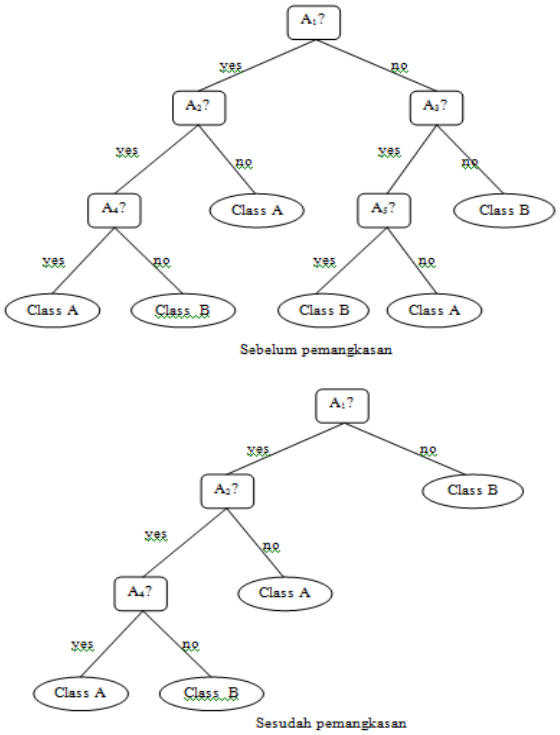
   Simpul internal merupakan simpul hasil percabangan dari simpul akar. Simpul ini memiliki satu *input* dan memiliki minimal dua *output*.

3. Simpul daun

   Simpul daun merupakan simpul terakhir dan tidak memiliki *output*. Simpul ini biasa dikenal sebagai simpul terminal atau keputusan.

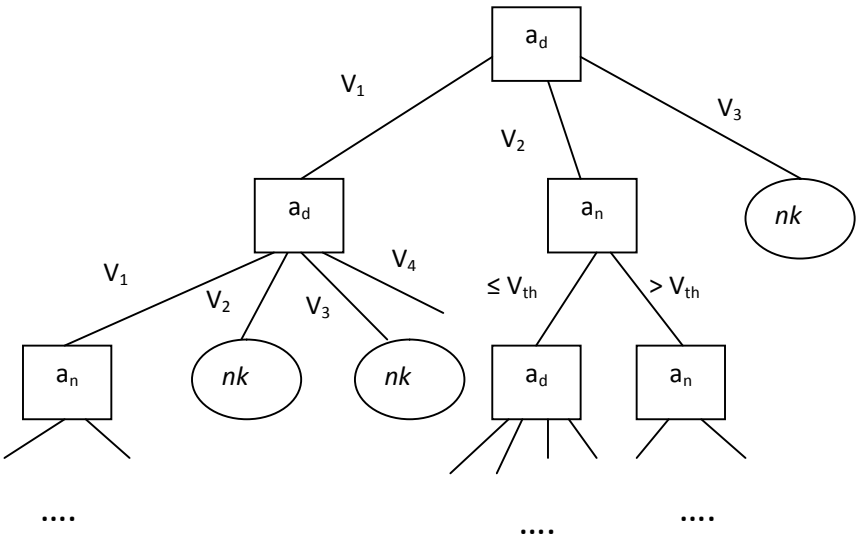
Pada saat membuat konstruksi pohon keputusan dilakukan seleksi atribut pada partisi tupel terbaik ke dalam kelas yang berbeda. Ketika pohon keputusan

dibangun akan banyak ditemui cabang yang *noise* atau *outlier* pada *data training*. Pada pohon keputusan terdapat eksekusi pemangkasan cabang. Pemangkasan tersebut bertujuan untuk meningkatkan akurasi klasifikasi data yang tidak terlihat. Contoh pemangkasan pada pohon keputusan ditunjukkan pada Gambar 2.9.



Gambar 2.9 Pohon Keputusan Sebelum Pemangkasan dan Sesudah Pemangkasan

Pembentukan pohon keputusan juga didasarkan pada nilai atribut yang terdapat pada *dataset*. Nilai atribut memiliki sifat heterogen, yaitu nilainya bersifat diskrit dan kontinu. Ilustrasi pohon keputusan ditunjukkan pada Gambar 2.10.



Gambar 2.10 Ilustrasi Pohon Keputusan Berdasarkan Sifat Nilai Atributnya

Berdasarkan Gambar 2.10, apabila *A* adalah atribut yang memiliki sejumlah *v* nilai yang berbeda. Jika atribut prediktor bersifat diskrit (a_d), maka cabang simpul dibuat untuk setiap nilai pada atribut diskrit tersebut (v_1, v_2, v_3, ... v_d). Apabila atribut prediktor bersifat kontinu atau numerik (a_n), maka cabang simpul dibuat untuk dua buah nilai $a_n \leq v_{th}$ dan $a_n > v_{th}$, v_{th} merupakan nilai ambang dari a_n. Atas dasar hal tersebut, diberikan konsep dasar algoritma dari pohon keputusan yang ditunjukkan pada Gambar 2.11.

Algoritma: Pembentukan_pohon_keputusan. *Pembentukan pohon keputusan dari tupel pelatihan pada partisi data, D.*

Masukan:

- *Partisi data, D, yang merupakan satu set data tupel pelatihan dan label kelas yang berkaitan;*
- *Daftar_atribut, kumpulan beberapa atribut;*
- *Metode_seleksi_atribut, sebuah prosedur untuk menentukan kriteria terbaik pemecahan data tupel ke dalam kelas masing-masing. Kriteria ini terdiri dari pemecahan_atribut dan kemungkinannya, baik pemecahan simpul atau pemecahan bagian.*

Hasil: *pohon keputusan.*

Metode:

(1) Bentuk sebuah simpul N;
(2) ***jika*** *tupel di D ada pada kelas yang sama, C,* ***maka***
(3) kembali N sebagai simpul daun yang diberi label kelas C;
(4) ***jika*** *daftar_atribut kosong* ***maka***
(5) kembali N sebagai simpul yang diberi label dengan kelas terbanyak di D; // kelas terbanyak
(6) berlaku ***metode_seleksi_atribut*** *(D, daftar_atribut) untuk* ***menemukan*** *pemecahan kriteria terbaik;*
(7) beri label N dengan kriteria_pemecahan;
(8) ***jika*** *pemecahan_atribut bernilai diskrit* ***dan***
beberapa pemecahan diperbolehkan ***maka*** *// tidak terbatas untuk pohon biner*
(9) daftar_atribut ← daftar_atribut – pemecahan_atribut; // hapus pemecahan_atribut
(10) ***Untuk setiap*** *j dari pemecahan_kriteria*
(11) kemudian D_j menjadi kumpulan data tupelo di D dengan hasil kepuasan j; // partisi
(12) ***jika*** *D_j kosong* ***maka***
(13) Lampirkan sebuah simpul daun dengan label kelas terbanyak di D untuk simpul N;
(14) ***Untuk lainnya*** *lampirkan simpul kembali dengan pembentukan_pohon_keputusan (D_j,daftar_atribut) pada N;*
berakhir untuk
(15) kembali N;

Gambar 2.11 Algoritma Dasar Pohon Keputusan

Berdasarkan Gambar 2.11, input algoritma dasar terdiri partisi data *D,* daftar atribut (*attribute list*), metode seleksi atribut (*attribute selection method*). Langkah-langkah untuk mengbangun sebuah pohon keputusan seperti yang ditunjukkan pada Gambar 2.11 diatas adalah sebagai berikut:

1. Pohon dimulai dengan simpul tunggal *N* yang merepresentasikan tupel *training* pada *D* (langkah 1).
2. Jika semua tupel di *D* berasal dari kelas yang sama, maka simpul *N* menjadi daun dan diberi label kelas tersebut (langkah 2 dan 3). Langkah 4 dan 5 merupakan kondisi akhir. Semua kondisi akhir dijelaskan pada akhir algoritma.
3. Jika tidak, maka metode seleksi atribut digunakan untuk memilih atribut *split*, yaitu atribut terbaik dalam memisahkan tupel ke dalam kelas masing-masing (langkah 6). Atribut tersebut menjadi atribut tes pada simpul *N* (langkah 7).
4. Terdapat dua kemungkinan yang dapat mempartisi *D*. Apabila *A* atribut *split* pada simpul *N* dan *A* memiliki sejumlah *v* nilai yang berbeda $\{a_1, a_2, \ldots, a_v\}$ maka pada *data training* dapat terjadi (langkah 8 dan 9):
   a. Jika *A* memiliki nilai-nilai bersifat diskrit, maka sebuah cabang dibentuk untuk setiap nilai *A*. Nilai total cabang yang akan dibentuk sebanyak *v* cabang. Partisi D_j terdiri dari *record* yang terdapat pada *D* yang memiliki nilai a_j untuk atribut *A*. Selanjutnya atribut *A* dihapus dari daftar atribut.

b. Jika *A* memiliki nilai yang bersifat kontinu, maka hasil pengujian simpul *N* akan menghasilkan dua cabang. Kedua cabang tersebut adalah *A*< *split point* dan *A*≥ *split point*. *Split point* merupakan keluaran metode seleksi atribut sebagai bagian dari kriteria untuk melakukan partisi. Selanjutnya *D* dipartisi, sehingga D_1 terdiri dari *record* dimana *A*< *split point* dan D_2 adalah sisanya.

5. Cabang akan dibuat untuk setiap nilai pada atribut tes dan tupel pada *data training* akan dipartisi lagi (langkah 10 dan langkah 11).
6. Proses pembentukan ini menggunakan proses rekursif untuk membentuk pohon pada setiap data partisi (langkah 14).
7. Proses rekursif akan berhenti jika telah mencapai kondisi sebagai berikut:
   a. Semua tupel pada simpul berada didalam satu kelas (langkah 2 dan 3).
   b. Tidak ada atribut lainnya yang dapat digunakan untuk mempartisi tupel lebih lanjut (langkah 4). Selanjutnya dalam hal ini, akan diterapkan jumlah terbanyak (langkah 5). Hal tersebut berarti mengubah sebuah simpul menjadi daun dan memberi label dengan kelas pada jumlah terbanyak. Sebagai alternatifnya, distribusi kelas pada simpul ini dapat disimpan.
   c. Tidak ada tupel yang digunakan untuk mencabang, suatu partisi D_j kosong (langkah 12). Selanjutnya dalam hal ini, sebuah daun dibuat dan diberi label dengan kelas yang memiliki kelas terbanyak di *D* (langkah 13).
8. Kembali menghasilkan pohon keputusan (langkah 15).

Algoritma yang sering digunakan pada pohon keputusan salah satunya C4.5. Pada penelitian ini akan digunakan algoritma tersebut (Han, J., et al, 2012: 331-336; Dwi A.S, 2013).

Algoritma C4.5 berlangsung secara rekursif mengunjungi setiap simpul keputusan, memilih split optimal, sampai tidak ada pemisahan yang terjadi lagi (Larose, 2005:116). Algoritma C4.5 menggunakan konsep *information gain* atau *entropy reduction* untuk memilih *split* optimal.

Misalkan *N* merupakan simpul partisi dari *D*. Apabila terdapat nilai *information gain* tertinggi maka akan terpilih sebagai atribut pemisah untuk simpul *N*. Perhitungan informasi yang dibutuhkan untuk mengklasifikasi pada tupel *D* dinyatakan sebagai berikut:

$$Info(D) = -\sum_{i=1}^{m} p_i \, log_2(p_i) \quad (2.8)$$

dimana $p_i = \frac{|C_{i,D}|}{|D|}$ yang merupakan probabilitas dari tupel *D* yang mempunyai kelas C_i. Sebuah fungsi log basis 2 digunakan karena informasi disandikan ke dalam bentuk *bits*. *Info (D)* merupakan rata-rata dari informasi yang dibutuhkan untuk mengetahui label kelas dari tupel *D*.

Misalkan *A* memiliki *v* nilai yang berbeda $\{a_1, a_2, ..., a_v\}$. Atribut *A* dapat digunakan untuk membagi *D* ke dalam *v* partisi $\{D_1, D_2, ..., Dv\}$, dimana D_j memuat tupel *D* yang memiliki nilai a_j dari *A*. Nilai *entropy* dari subset *A* dapat dihitung dengan persamaan berikut:

$$Info_A(D) = E(A) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info\,(D_j) \quad (2.9)$$

dimana $\frac{|D_j|}{|D|}$ merupakan bobot dari subset *j* dan jumlah sampel pada subset yang mempunyai nilai a_j dari *A*, dibagi dengan jumlah tupel dari *D*. *Entropy* merupakan informasi harapan yang dibutuhkan untuk mengklasifikasi suatu tupel dari *D* berdasarkan partisi dari atribut *A*.

Menurut Han, J., et al (2012:337), nilai *information gain* dari atribut *A* pada subset *D* dapat dihitung dengan persamaan berikut:

$$Gain\ (A) = info\ (D) - E(A) \tag{2.10}$$

Information gain didefinisikan sebagai perbedaan diantara informasi asli yang dibutuhkan (berupa proporsi kelas) dengan jumlah informasi baru yang didapatkan dari partisi *A*. Atribut *A* yang memiliki nilai *information gain* tertinggi dipilih sebgai pemisah atribut pada simpul *N*.

Proses untuk menghitung nilai *E(A)* bergantung dari nilai suatu atribut. Jika a_d adalah atribut diskrit, maka tupel *D* dubagi menjadi sub tupel $D_1 \ldots D_n$, dimana *n* jumlah nilai unik pada atribut a_d dan D_i adalah sub tupel yang memiliki nilai atribut $a = i$. Jika a_n adalah atribut kontinu, maka sub tupel *D* dibagi menjadi dua sub tupel D_{v1} dan D_{v2} dengan $D_1 = \{v_j | v_j \leq sp\}$ dan $D_2 = \{v_j | v_j > sp\}$, dimana *sp* merupakan sebuah nilai ambang (*split point*). Selanjutnya untuk memperoleh nilai *sp* mula-mula tupel di *D* diurutkan berdasarkan nilai dari a_n. Misalkan nilai yang berturut adalah $sp_1 \ldots sp_m$. Jika $i \in [1, m-1]$ dan nilai $sp = \frac{(v_j + v_{i+1})}{2}$, maka tupel yang dipecah dapat dinyatakan dengan $D_1 = \{v_j | v_j \leq sp\}$ dan $D_2 = \{v_j | v_j > sp\}$. Nilai *split information* digunakan pada pencarian nilai *gain ratio*

untuk mengatasi bias terhadap atribut yang memiliki banyak nilai unik. Persamaan *split information* dan *gain ratio* dinyatakan sebagai berikut:

$$Split\ Info\ (A) = -\sum_{i=j}^{m} p_j log_2(p_j) \quad (2.11)$$

$$Gain\ Ratio\ (A) = \frac{Gain\ (A)}{Split\ Info\ (A)} \quad (2.12)$$

Apabila atribut tersebut memiliki nilai *gain ratio* terbesar maka atribut tersebut terpilih sebagai atribut *split* pada konstruksi pohon keputusan (Dwi A.S, 2013:26-28; Han, J., et al, 2012:337-339).

E. *Waikato Environment For Knowledge Analysis* (WEKA)

WEKA merupakan sebuah sistem *data mining* yang dikembangkan oleh Universitas Waikato di Selandia Baru yang mengimplementasikan algoritma *data mining* (Aksenova, 2004: 2). WEKA adalah sebuah koleksi mesin pembelajaran algoritma untuk tugas-tugas *data mining*. Algoritmanya dapat diterapkan secara langsung ke data set atau dipanggil dari kode Java sendiri. WEKA berisi alat-alat untuk data pra-pengolahan (*pre-processing*), klasifikasi, regresi, *clustering,* aturan asosiasi, dan visualisasi. WEKA juga sesuai untuk mengembangkan skema pembelajaran mesin baru (www.cs.waikato.ac.nz). WEKA menyediakan inplementasi dari pembelajaran algoritma yang dapat dengan mudah untuk diterapkan pada *dataset*. Implementasi tersebut juga mencakup bebagai alat untuk mengubah *dataset*, *pre-process dataset*, memberikan skema pembelajaraan, dan menganalisis klasifikasi yang dihasilkan dan kinerjanya tanpa harus menuliskan kode program (Witten, 2011: 403).

Salah satu penggunaan WEKA adalah untuk menerapkan metode pembelajaran untuk *dataset* dan menganalisis *output* untuk mempelajari data secara lebih lanjut. Penggunaan lainnya adalah digunakan sebagai model pembelajaran untuk memprediksi pada sebuah kasus baru. Penerapan lainnya dilakukan pada beberapa pembelajaran yang berbeda dan membandingkan kinerja dari mereka dan dipilih salah satu untuk digunakan dalam memprediksi. Pada tampilan utama dapat anda pilih metode pembelajaran yang diinginkan pada menu. Banyak metode yang memiliki parameter yang selaras, yang dapat diakses melalui lembar properti atau editor objek. Sebuah modul evaluasi umum digunakan untuk mengukur kinerja semua pengklasifikasian (Witten, 2011: 404). Berikut tampilan utama pada WEKA ditunjukkan pada gambar 2.12.

Gambar 2.12 Tampilan utama WEKA

Seperti yang ditunjukkan pada Gambar 2.12 WEKA GUI Chooser memiliki empat tombol utama, yaitu :

1. *Explore*

Explore merupakan sebuah pilihan bidang untuk menjelajahi data dengan WEKA. *Explore* memiliki enam jenis tab didalamnya dengan tugas sebagai berikut (Witten, 2011: 404-416):

a. *Preprocess* merupakan bidang pemilihan data set dan modifikasinya dengan berbagai cara.
b. *Classify* merupakan pelatihan pembelajaran skema yang melaksanakan klasifikas atau regresi dan evaluasinya.
c. *Cluster* merupakan pembelajaran *cluster* atau pengelompokan untuk *dataset*.
d. *Associate* merupakan pembelajaran aturan asosiasi untuk data dan evaluasinya.
e. *Select attributes* merupakan bidang pemilihan aspek yang paling relevan dalam *dataset*.
f. *Visualize* merupakan bidang tampilan plot dari dua dimensi yang berbeda dari data tersebut dan interaksinya.

2. *Experimenter*

Experimenter merupakan sebuah pilihan bidang untuk melakukan eksperimen dan melakukan uji statistik antara skema pembelajaran. *Experimenter* memungkinkan pengguna untuk membuat percobaan dalam skala besar., mulai dari percobaan dijalankan sampai percobaan selesai hingga dilakukan analisis kinerja secara statistik terhadap apa yang telah diperoleh.

3. *Knowledge Flow*

   Knowledge Flow merupakan sebuah pilihan bidang yang mendukung fungsi dasar yang sama seperti *explore* tetapi dengan antarmuka *drag* dan *drop*. Salah satu keuntungannya adalah *knowledge flow* mendukung adanya pembelajaran tambahan. Pada tampilan utama *knowledge flow* ini pengguna dapat melihat tata letak sebuah kinerja dari proses yang dilakukannya, pengguna dihubungkan kedalam sebuah graf berarah yang memproses dan menganalisis data. Pada bagian ini merupakan gambaran secara jelas bagaimana data berjalan dalam sistem dimana tidak disediakan dalam *explore.*

4. *Simple CLI*

   Simple CLI merupakan sebuah pilihan bidang yang memberikan tampilan garis perintah sederhana yang memungkinkan adanya perintah langsung eksekusi dari WEKA untuk sistem operasi yang tidak memberikan tampilan garis perintahnya sendiri. Dibalik tampilan interaktif *explore, experimenter, knowledge flow* pada WEKA terdapat fungsi dasar yang dapat diakses secara langsung pada tampilan garis perintah. Garis perintah terdapat pada *simple CLI*, pada tampilan utama WEKA panel *simple CLI* terletak disebelah kanan bawah.

WEKA sebagai mesin pembelajaran yang memiliki tugas dalam penggunaan sebuah metode, WEKA memiliki beberapa metode utama dalam permasalahan *data mining*, yaitu regresi, klasifikasi, *clustering, association rule mining,* dan pemilihan atribut. Pengenalan data merupakan bagian yang tidak terpisahkan dari

sebuah pekerjaan, dan banyak fasilitas visualisasi data dan alat data *pre-processing* yang disediakan. Semua algoritma dalam WEKA mengambil *input* dalam bentuk tabel relasional tunggal dalam format *Attribute Relation File Format* (*ARFF*), yang dapat dibaca dari sebuah *file* atau dihasilkan oleh permintaan basis data (Singhal & Jena, 2013), (Witten, 2011:407-519).

F. BESMART Universitas Negeri Yogyakarta

Jaya Kumar berpendapat (Suyanto, 2005), *e-learning* merupakan pengajaran dan pembelajaran yang menggunakan rangkaian elektronik (LAN, WAN, atau internet) untuk menyampaikan isi pembelajaran, interaksi, atau bimbingan. *E-learning* memberikan sebuah konsep pendidikan yang baru dan berbeda dari konsep pendidikan lama, penyampaian yang dilakukan secara digital dan pembelajarannya berorientasi untuk pengajar dan pelajar. *E-learning* menjadi alat penting untuk mendukung sistem pembelajaran, seperti yang diterapkan oleh UNY.

E-learning UNY diimplementasikan dengan paradigma pembelajaran pembelajaran *online* terpadu menggunakan *Learning Management System* (LMS) Moodle. Sistem *e-learning* UNY yang telah berfungsi sejak tahun 2006 kini dikenal dengan nama BESMART. Beberapa kegiatan yang dapat dilakukan di BESMART, yaitu *user, course, assignment, upload, resource, quiz, message, survey, chat, glossary, forum, vicon, scorm.* Penjelasan masing-masing kegiatan diberikan sebagai berikut:

1. *User*

   User dilabelkan pada kegiatan pengguna yang hanya melakukan *log in* saja tetapi tidak melakukan kegiatan lain.

2. *Course*

   Kegiatan *course* merupakan kegiatan perkuliahan yang dilaksanakan secara *online.*

3. *Assignment*

   Assignment digunakan untuk menyatakan kegiatan dosen dalam memberikan tugas kepada mahasiswa. Sebaliknya mahasiswa juga melakukan transaksi atau pengiriman tugas.

4. *Upload*

   Kegiatan yang dilakukan untuk meng-*upload* materi perkuliahan.

5. *Resource*

   Resource merupakan tempat-tempat materi kuliah yang telah disediakan oleh dosen dan mahasiswa dapat mengunduhnya.

6. *Quiz*

   Quiz merupakan kegiatan dimana mahasiswa mengerjakan tes/*quiz* yang telah disiapkan oleh dosen.

7. *Message*

   Kegiatan berkirim pesan secara pribadi yang dilakukan oleh dosen dan mahasiswa serta antar mahasiswa secara *real time* (bila sedang *online*).

8. *Survey*

   Kegiatan survei dengan instrument yang baku.

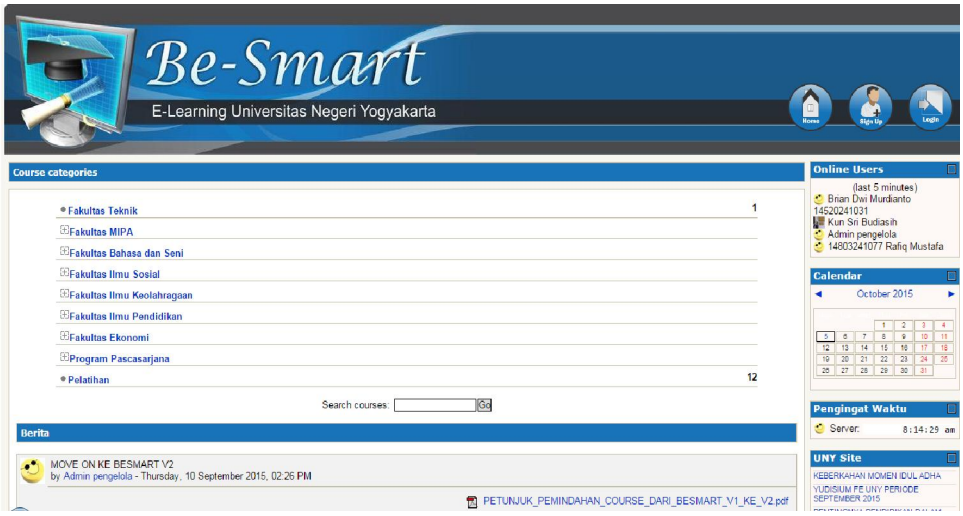
9. *Chat*

   Kegiatan interaksi yang dilakukan antara sesama mahasiswa dan dosen yang digunakan untuk berkomunikasi dalam bentuk tulisan (teks) secara *real time*.

10. *Forum*

    Forum merupakan kegiatan interaksi melalui forum diskusi yang digunakan sebagai wahana tukar menukar informasi.

Selain beberapa kegiatan yang telah diuraiakan diatas, masih terdapat tugas lainnya yang dapat dilakukan di BESMART, diantaranya menyusun silabi, mengolah nilai mahasiswa, dan lainnya. Berikut diberikan tampilan utama BESMART yang dapat dilihat pada Gambar 2.13 (TIM ICT UNY, 2011: 16-17).



Gambar 2.13 Tampilan Utama BESMART UNY

Semua aktivitas yang terjadi di dalam BESMART terekam dalam riwayat *log*. Riwayat *log* merupakan catatan dari setiap pengguna yang *log in* ke BESMART. Catatan tersebut merangkum identitas pengguna yang diketahui melalui *user id,* nama pengguna, email pengguna, waktu akses, aktivitas yang dilakukan.

Berdasarkan riwayat *log* tersebut maka dapat diketahui informasi mengenai pengguna BESMART.

G. Penelitian Yang Relevan

Penelitian mengenai penerapan *data mining* dengan teknik *clustering* untuk sistem pembelajaran *online* sudah banyak dilakukan. Diantaranya memanfaatkan *log history* yang terjadi setiap harinya. Seperti yang dilakukan oleh Ningtyas, dkk pada tahun 2008 dalam jurnalnya "*Analisis Perilaku Pengguna Sistem E-Learning Universitas Gunadarma*". Pada penelitian tersebut digunakan 600.000 *record* kemudian dilakukan *cleaning* sejumlah 55% dari *record*. Log penggunanya berisi pengguna V-class (NPM), waktu akses (tanggal dan jam), alamat IP, kode mata kuliah, nama mata kuliah, jenis kegiatan, dan alamat URL yang diakses. Penelitian ini bertujuan untuk melihat kecenderungan waktu akses pengguna, kegiatan yang paling sering dilakukan, rerata waktu. Pada penelitian yang dilakukan oleh Ningtyas, dkk ini teknik *clustering* digunakan sebanyak dua kali. Hasilnya bahwa rerata akses V-class Universitas Gunadarma lebih banyak dilakukan waktu kuliah (antara pukul 07.30-17.30). Pada *clustering* kedua dilihat jurusan mana yang paling banyak mengakses V-class dan hasilnya adalah mahasiswa jurusan Sistem Informasi.

Selanjutnya Firman Harjuan Jaya dalam tugas akhir skripsinya *Analisis Cluster dan Association Rule Pada Data Web Log Server* melakukan penelitian pada studi kasus data *web log server* di situs www.fschool.us dengan menggunakan proses multivariat. Data yang digunakan pada penelitian ini ada 5 variabel yaitu *Newsletter, Free e-book, Free Jurnal, Software* Statistik, dan

Software Teknik Informatika. Obyek yang digunakan ada 91 jenis IP (*Internet Protocol*) pengunjung situs web www.fschool.us. Proses *clustering* dilakukan dengan SPSS 17, tahap *clustering* dilakukan dengan dua metode yaitu *single linkage* dan *k-means.* Hasilnya terdapat 4 *cluster* yang dihasilkan masing-masing berisi kelompok IP yang cenderung meng-*click content Free e-Book*, *Newsletter, Newsletter, Free e-Book, Free Jurnal, Software* Statistik, dan *Software* Teknik Informatika, *Free Jurnal dan Software* Teknik Informatika. Pada proses asosiasi ditemukan 5 *rule*.