# BAB II
# KAJIAN PUSTAKA DAN DASAR TEORI

## 2.1 Kajian Pustaka

Data *mining* merupakan kunci dalam pembentukan *classifier* yang membutuhkan akurasi dan efisiensi waktu yang tinggi untuk *database* dalam jumlah yang besar. *Dataset* akan terus-menerus membentuk *rule* dalam jumlah yang tidak sedikit [JUN-10]. Proses data mining menggunakan *association rules* dilakukan dengan cara membandingkan hubungan antar data *itemset* dari kumpulan data yang berjumlah besar. Algoritma yang digunakan dalam a*ssocation rule mining* diantaranya adalah *apriori* dan *Frequent Pattern*. Ketika melakukan proses *generate* pada *itemset, apriori* melakukan pengecekan transaksi pada *database* secara keseluruhan dengan beberapa kali perulangan untuk mendapatkan *frequent itemset* [LUO-10].

Algoritma yang juga sering digunakan untuk proses *mining* dalam *association rules* yaitu algoritma *FP-Growth*. Jika dibandingkan dengan *apriori*, *FP-Growth* memiliki efisiensi yang lebih tinggi [QIN-06]. *FP-Growth* menghitung *frequent itemset* dengan membentuk *FP-Tree*. *FP-Tree* adalah suatu model dari *prefix tree* [REZ-11] yang mendeteksi *frequent itemset* secara eksklusif dari kandidat *itemset* yang telah dibentuk [SIL-08]. *FP-Tree* mewakili *frequent pattern* dan memudahkan dalam masalah *multi-scan*. Algoritma ini melakukan proses *mining* tanpa *candidate generation* dan menggunakan teknik *divide and conquer* untuk mendapatkan *frequent itemset* [HAN-00].

## *2.2. Data Mining*

### 2.2.1 Pengertian *Data Mining*

Secara sederhana *data mining* adalah penambangan atau penemuan informasi baru dengan mencari pola atau aturan tertentu dari sejumlah data yang sangat besar [DAV-04]. *Data mining* adalah kegiatan menemukan pola yang menarik dari data dalam jumlah besar, data dapat disimpan dalam *database*, data *warehouse*, atau penyimpanan informasi lainnya. *Data mining* berkaitan dengan bidang ilmu-ilmu lain, seperti *database system*, *data warehousing*, statistik,

*machine learning, information retrieval,* dan komputasi tingkat tinggi. Selain itu data mining didukung oleh ilmu lain seperti *neural network,* pengenalan pola, *spatial data analysis, image database, signal processing* [HAN-00].

Beberapa pengertian tentang *data mining,* antara lain sebagai berikut :

1. *Data mining* adalah suatu proses dalam mencari korelasi, pola, atau tren yang bermanfaat dari sebuah data yang berukuran besar dengan menggunakan teknik statistika ataupun matematika [LAR-05].
2. *Data mining* adalah salah satu teknik dalam ilmu komputer yang digunakan untuk menggali dan mengambil suatu informasi pada banyak data [REN-04].
3. *Data mining* adalah mencocokkan data dalam suatu model untuk menemukan informasi yang tersembunyi dalam basis data [HAN-01].
4. *Data mining* adalah serangkaian proses untuk menggali nilai tambah dari suatu kumpulan data berupa pengetahuan yang selama ini tidak diketahui secara manual [PRA-03].
5. *Data mining* adalah proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, dan *machine learning,* untuk mengekstraksi dan mengindentifikasi informasi yang bermanfaat dan pengetahuan yang terkait di berbagai *database* [TUR-05].

Berdasarkan beberapa pengertian diatas, dapat disimpulkan bahwa penggalian atau penambangan data (*data mining*) adalah proses pencarian otomatis terhadap pola dalam data dalam jumlah besar untuk melihat informasi yang tersembunyi dengan menggunakan perangkat seperti klasifikasi, penggugusan (*clustering*), dan *association rule.*

### 2.2.2. Proses *Data Mining*

*Data mining* dan *knowledge discovery in database* (KDD) merupakan istilah yang memiliki konsep berbeda akan tetapi saling berkaitan karena *data mining* adalah bagian dalam proses *knowledge discovery in database* (KDD) [KUS-07]. Proses *knowledge discovery in database* (KDD) secara umum adalah sebagai berikut :

1. ***Data Selection***

   Merupakan tahap seleksi data, yang akan digunakan dalam proses *data mining*, dari sejumlah besar data operasional. Hasil dari seleksi data disimpan dalam suatu berkas terpisah dari *database* operasional.

2. ***Prepocessing* atau *Cleaning***

   Pada tahap ini, dilakukan pembuangan duplikasi data, pemeriksaan data yang tidak konsisten, dan memperbaiki kesalahan pada data, serta memperkaya data yang sudah ada dengan data atau informasi eksternal.

3. ***Transformation***

   Tahap transformasi sangat bergantung pada jenis atau pola informasi yang akan dicari dalam basis data. Menurut [HUD-10], transformasi merupakan proses pengubahan sesuai format yang dibutuhkan untuk digunakan dalam proses *data mining*.

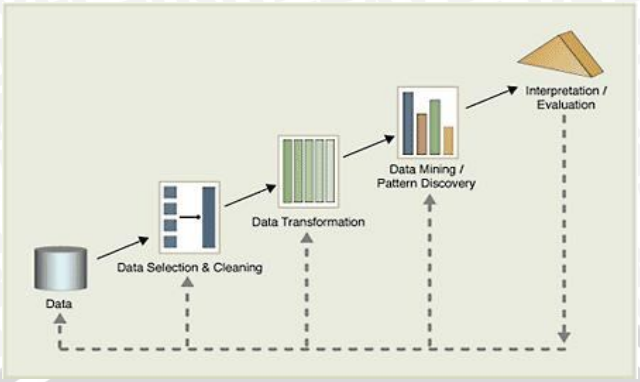
4. ***Data Mining***

   Merupakan tahap pencarian pola atau informasi dari data yang terpilih dengan menggunakan metode atau teknik tertentu. Ketepatan metode atau teknik yang dipilih sangat bergantung pada tujuan dari proses *knowledge discovery in database* (KDD) secara keseluruhan.

5. ***Interpretation***

   Pola informasi yang dihasilkan dari proses *data mining* perlu ditampilkan dalam bentuk yang mudah dimengerti oleh pihak yang berkepentingan. Tahap ini mencakup pemeriksaan apakah pola atau informasi yang ditemukan bertentangan dengan fakta atau hipotesa yang ada sebelumnya.

Keseluruhan dari proses *knowledge discovery in database* (KDD) diilustrasikan pada Gambar 2.1.




**Gambar 2. 1 Tahap – tahap Proses *Knowledge Discovery in Database* [HAN-01]**

Beberapa prosedur umum untuk menyelesaikan permasalahan dalam *data mining* [KAN-03] :

1. Merumuskan permasalahan

   Pada tahap ini di tetapkan sebuah rumusan masalah serta variabel-variabel yang terlibat.

2. Mengumpulkan data

   Pada prosedur ini, konsentrasi ditujuakan pada proses pembuatan atau pengumpulan data.

3. *Preprocessing* data

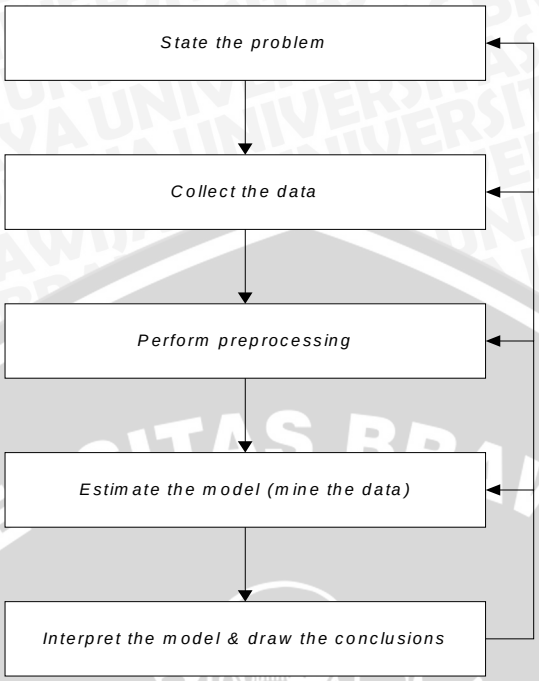
   Untuk menyeleksi data yang akan digunakan dalam proses.

4. Estimasi model

   Dapat disebut sebagai proses utama pada prosedur ini, sebab implementasi dari teknik *data mining* dilakukan pada prosedur ini.

5. Menafsirkan informasi yang dihasilkan dari proses sebelumnya

Beberapa prosedur umum untuk menyelesaikan permasalahan dalam *data mining* dijelaskan pada gambar 2.2.




**Gambar 2.2. Proses *Data Mining* [KAN-03]**

## 2.3 *Association Rule*

*Association rule* adalah teknik *data mining* untuk menemukan aturan hubungan asosiatif antara suatu kombinasi *item* dengan tujuan akhir mendapatkan kombinasi *item* yang dikatakan menarik *(interesting association rules). Association rule* meliputi dua tahap [GRE-05] :

1. Mencari kombinasi yang paling sering terjadi dari suatu *itemset.*
2. Mendefinsikan *Condition* dan *Result* (untuk *conditional association rule*).

Penting tidaknya suatu aturan asosiatif dapat diketahui dengan dua parameter [SAN-07], yaitu :

1. *Support,* suatu ukuran yang menunjukkan seberapa besar tingkat dominasi suatu *item* atau *itemset* dari keseluruhan transaksi. Ukuran ini menentukan apakah suatu *item* atau *itemset* layak untuk dicari *confidence factor*-nya.
2. *Confidence,* suatu ukuran yang menunjukkan hubungan antar dua atau lebih *item* secara *conditional* .

Kedua ukuran ini nantinya berguna dalam menentukan *interesting association rules,* yaitu nilai suatu *item* dengan melihat peluang munculnya *item* berdasarkan support dari *item* tersebut dan kemudian dibandingkan dengan batasan (*threshold*)

yang ditentukan oleh *user*. Batasan tersebut umumnya terdiri dari *minimum support* dan *minimmum confidence*.

Menurut [KUS-07], metodologi dasar analisis asosiasi terbagi menjadi dua tahap, yaitu :

1. Analisis Pola Frekuensi Tinggi

   Tahapan ini mencari kombinasi *item* yang memenuhi syarat minimum dari nilai *support* dalam database. Nilai *support* sebuah *item* diperoleh dengan rumus berikut :

$$Support\ (A) = \frac{Jumlah\ transaksi\ mengandung\ A}{Total\ transaksi} \quad (2.1)$$

Sedangkan nilai *support* dari 2 *item* diperoleh dengan rumus sebagai berikut:

$$Support\ (A,B) = \frac{\sum Transaksi\ mengandung\ A\ dan\ B}{\sum Transaksi} \quad (2.2)$$

2. Pembentukan Aturan Asosiasi

Setelah semua pola frekuensi tinggi ditemukan, barulah dicari aturan asosiasi yang memenuhi syarat minimum untuk *confidence* dengan menghitung *confidence* aturan asosiatif A→B.

Nilai *confidence* dari aturan A→B diperoleh dari rumus berikut:

$$Confidence = \frac{\sum Transaksi\ mengandung\ A\ dan\ B}{\sum Transaksi\ mengandung\ A} \quad (2.3)$$

### 2.4. *FP-Tree*

*FP-tree* adalah sebuah *tree,* yang terdiri dari : satu *header table*, satu *root* yang diberi label "null", dan satu himpunan *item prefix subtree* sebagai *node* dari anak *root*. Masing-masing *entry* dalam *header table* adalah *frequent item*, dan setiap *record* terdiri dari dua atribut yaitu nama *item* (*item_name*) dan *head of node-link*. Sedangkan setiap *node* anak terdiri dari atribut : nama *item* ,*count*, dan *node-link*. Dimana *count* menunjukkan jumlah transaksi direpresentasikan oleh

cabang yang mengandung *node* tersebut, dan *node-link* menghubungkan *node* ke *node* berikutnya pada *tree* tersebut yang mempunyai nama *item* yang sama atau *null* jika tidak ada [MAR-06].

Pembentukan ***FP-Tree***

**Input :** Basis Transaksi (D) dan nilai *minimum support* (minsup);
**Output :** Ifrequent pattern tree, *FP-tree*;
**Method :** *FP-tree* dibentuk dengan cara berikut.

1. Transaksi di *database* (DB) ditelusuri sekali agar didapat himpunan *frequent item* (F) dan nilai support-nya. Lalu F diurutkan mulai dari yang mempunyai nilai *support* terbesar sampai terkecil , dan hasil pengurutannya dilambangkan dengan L (daftar *frequent item).*
2. Buat *root* dari FP-tree, T, dan diberi nilai *null*. ***Foreach*** transaksi *Trans* di DB lakukan langkah berikut.
   Pilih dan diurutkan *frequent items* dalam *Trans* (disebut [*p*|P]) mengacu pada L, dimana *p* adalah elemen pertama dan P adalah *item* berikutnya. Panggil ***insert_tree*([*p*|P],T).**

   Fungsi ***insert_tree*([*p*|P],T)** terdiri dari langkah-langkah berikut**.**

   **If** *T* punya anak *N* dimana *N*.nama-item =*p*.nama-item, maka nilai N bertambah 1; **else** buat node baru *N*, beri nilai 1, N.*parent.link = T,* dan *N.link* = node yang sama nama itemnya dihubungkan oleh struktur *node-link*.
   **If** P tidak kosong, **then** panggil *insert_tree* (P,N) secara rekursif.

**Gambar 2.3. Algoritma *FP-Tree* [HAN-00]**

## 2.5. *FP-Growth*

Salah satu algoritma yang tercepat dan terpopuler saat ini dalam penambangan *frequent itemset* adalah FP-*Growth* [HAN-00]. Algoritma *FP-Growth* adalah algoritma pencarian pola yang sering muncul (*frequent pattern*) berdasarkan struktur data *FP-tree* (*Frequent Pattern tree*). Algoritma *FP-Growth* menggunakan strategi *devided and conquer*, dimana membagi suatu permasalahan besar menjadi permasalahan-permasalahan yang lebih kecil (*devide*), pembagian dilakukan terus sampai ditemukan bagian masalah kecil yang mudah untuk dipecahkan (*conquer*) [FEB-09].

Penggalian *itemset* yang *frequent* dengan menggunakan algoritma *FP-Growth* akan dilakukan dengan cara membangkitkan struktur data *Tree* atau disebut dengan *FP-Tree*. Metode *FP-growth* melibatkan tiga tahapan utama yaitu [HAN-00] :

1. Tahap Pembangkitan *Conditional Pattern Base*

   *Conditional pattern base* merupakan *sub-database* yang berisikan *prefix path* (himpunan *item* terurut yang mengawali k-*itemsets*), dan *suffix pattern* (k-*itemsets*). Misalnya, sebuah *itemset* yang telah terurut berdasarkan support *descending order* {I6, I3, I1, I13, I16}, apabila I16 merupakan *suffix pattern* maka I6, I3, I1, I13 adalah *prefix path.* Pembangkitan *conditional pattern base* didapatkan melalui *FP-tree* yang telah dibangun berdasarkan sebuah basis data transaksi.

2. Tahap Pembangkitan *Conditional FP-tree*

   Pada tahap ini, *support count* untuk tiap *item* pada setiap *conditional pattern base* akan dijumlahkan, kemudian *item* yang memiliki jumlah *support count* lebih besar sama dengan *min_sup* akan dibangkitkan menjadi sebuah *tree* yang disebut dengan *conditional FP-tree*.

3. Tahap Pencarian Frequent Itemset

   Pada tahap ini, apabila *Conditional Fp-Tree* merupakan *single path*, maka akan didapatkan *frequent itemsets* dengan melakukan kombinasi *item* untuk setiap *Conditional Fp-Tree*. Jika bukan *single path* maka akan dilakukan pembangkitan *FP-Growth* secara rekursif.

```
Input       : FP-Tree Tree
Output      : Rt sekumpulan lengkap pola frequent
Method      : FP-Growth (Tree, null)
Procedure   : FP-Growth (Tree, α)
{
01          : if Tree mengandung single path p;
02          :  then untuk tiap kombinasi (dinotasikan β)
              dari node-node dalam path P do
03          : bangkitkan pola β α dengan support = minimum
              support dari node-node dalam β;
04          : else untuk tiap a1 dalam header dari Tree do
              {
05          : bangkitkan pola
06          : bangun β = a1 α dengan support = a1.support
07          : if Tree β = Ø
08          : then panggil FP-Growth (Tree, β) }
            }
```

***Pseudocode* 2.1. Algoritma *FP-Growth* [HAN-00]**

Misalkan terdapat *itemset* seperti pada tabel 2.1 berikut dengan *minimal support* = 3.

**Tabel 2.1. *Itemset***

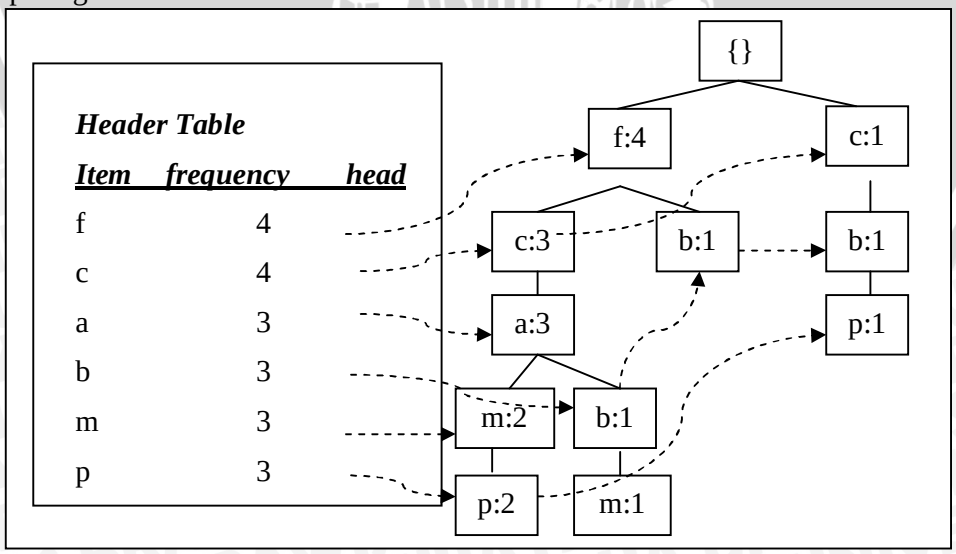
| TID | *Item* yang dibeli |
|---|---|
| 100 | {f, a, c, d, g, i, m, p} |
| 200 | {a, b, c, f, l, m, o} |
| 300 | {b, f, h, j, o, w} |
| 400 | {b, c, k, s, p} |
| 500 | {a, f, c, e, l, p, m} |

*Minimal support* dibutuhkan untuk mencari *item frequency*, dimana dalam tabel 2.1. didefinisikan *minimal support* = 3. Lalu akan dilakukan *scan* pada tabel tersebut agar didapatkan *item frequency* seperti pada tabel 2.2.

**Tabel 2.2 *Header Table***

| Item | Frequency |
|---|---|
| f | 4 |
| c | 4 |
| a | 3 |
| b | 3 |
| m | 3 |
| p | 3 |

Dari tabel 2.2. menunjukkan jumlah *item* dalam *database* transaksi dan dari *header table* diatas dapat dibuat *F-List* yaitu f-c-a-b-m-p. *F-list* ini nantinya akan menjadi panduan dalam membuat *FP-Tree*. Misal pada transaksi ID 100 didapatkan *item* yang sering dibeli adalah f, c, a, m, p; maka dimasukkan f-c-a-m-p seperti gambar 2.6 dimana {} menandakan *root*. Selanjutnya dimasukkan data pada transaksi ID 200: f, c, a, b, m; karena jalur f-c-a sudah ada maka ditambahkan jumlah f, c, a, sebanyak satu dan buat jalur baru untuk b dan m. Begitu seterusnya hingga semua transaksi dimasukkan dan didapatkan *FP-Tree* seperti gambar 2.4 berikut:



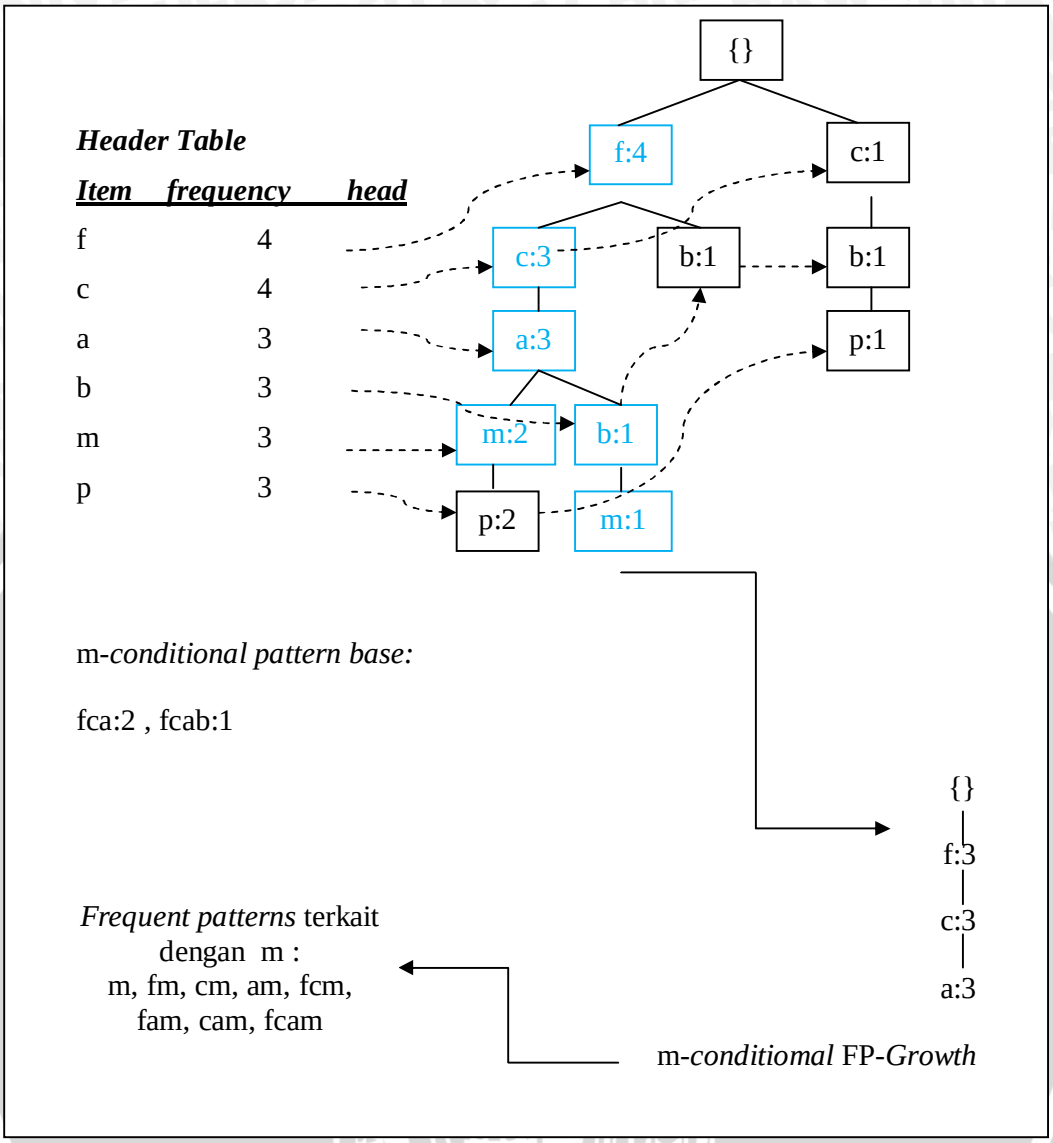

**Gambar 2.4. *FP-Tree* [HAN-00]**

Dari *FP-Tree* pada gambar 2.4 dapat dibuat *conditional pattern*. Misalkan c didapatkan melalui jalur f:3, a didapatkan melalui jalur fc:3, b didapatkan melalui fca:1, f:1, c:1, dst. f tidak ditulis karena dia merupakan item pertama dalam *F-List* dan diatas jalur f merupakan *root*. Cara mendapatkan *conditional pattern* yakni dilihat dari *FP-Tree*, misal a:3 pada *header table*, maka ditelusuri jalur diatasnya dari *root* untuk dapat menuju harus melalui f dan c. Maka didapatkan *conditional pattern* untuk a yakni fc:3.

**Tabel 2.3. *Conditional Pattern***

| ***Item*** | ***Conditional Pattern Base*** |
|---|---|
| c | f:3 |
| a | fc:3 |
| b | fca:1, f:1, c:1 |
| m | fca:2, fcab:1 |
| p | fcam:1, cb:1 |

Misalkan ingin didapatkan *frequent pattern* pada m. Gunakan *header table* dan *conditional pattern* maka didapatkan *m-conditional FP-Growth*. Untuk mencapai m harus melewati jalur fca:1 dan fcab:2, diambil demikian karena *minimal support* = 3. Dari data tersebut didapatkan f, c, a untuk *m-conditional FP-Growth*, lalu dari *m-conditional FP-Growth* dibuat *frequent pattern* yang terkait dengan m seperti gambar 2.5.




**Gambar 2.5. *Mining FP-Tree* (Han,2000)**

## 2.6. *Lift Ratio*

Selain menggunakan *confidence* untuk mengukur tingkat kebenaran, dikenal juga *lift ratio* untuk melihat kuat tidaknya aturan asosiasi dengan membandingkan nilai *confidence* dengan nilai *benchmark confidence*. Dimana diasumsikan kejadian *item* dari *consequent* (*item* setelah maka) dalam suatu transaksi adalah *independent* dengan kejadian dari *antecedent* (*item* setelah jika) dari suatu asosiasi. Istilah *antecedent* untuk mewakili bagian "jika" dan

*consequent* untuk mewakili bagian "maka". Dalam keadaan independen, maka *support* adalah [SAN-07] :

P(*antecedent* dan *consequent*) = P(*antecedent* * P (*consequent)* (2.4)

Dan *confidence benchmark* dinyatakan dengan :

$$\frac{P(antecedent)*P(consequent)}{P(antecedent)} = P(antecedent) \quad (2.5)$$

Nilai estimasi dari *confidence benchmark* dihitung dari data dari suatu aturan dihitung dengan :

$$Confidence\ benchmark = \frac{\text{Jumlah transaksi dengan } item \text{ dalam } consequent}{\text{Jumlah transaksi dalam } database} \quad (2.6)$$

Jadi *lift ratio* adalah perbandingan antara *confidence* untuk sauatu aturan dibagi dengan *confidence*, dimana diasumsikan *consequent* dan *antecedent* saling *independent.*

$$Lift\ Ratio = \frac{confidence}{Benchmark\ confidence} \quad (2.7)$$

*Lift* merupakan sebuah angka *ratio* yang menunjukkan berapa banyak kemungkinan menemukan sebuah atribut muncul bersama dengan atribut lainnya dibandingkan dengan seluruh kejadian adanya atribut yang terpenuhi [AMI-11].

*Lift* menunjukkan adanya tingkat kekuatan *rule* atas kejadian acak dari *antecedent* dan *consequent* berdasarkan pada *support*nya masing-masing. Hal ini akan memberikan informasi tentang perbaikan dan peningkatan probabilitas dari *consequent* berdasarkan *antecedent. Lift* didefinisikan sebagai berikut :

$$Lift = \frac{Confidence}{Expected\ Confidence} \quad (2.8)$$

Dimana,

$$Expected\ Confidence = \frac{Jumlah\ transaksi\ memiliki\ item\ consequent}{Total\ jumlah\ transaksi} \quad (2.9)$$

Atau dengan cara :

$$Lift = \frac{\Pr(A|C)}{\Pr(C)} \quad (2.10)$$

Ketika *lift* sama dengan 1 maka A dan B adalah *independent* karena Pr (C|A)=Pr(C). Ketika probabilitas C terjadi dipengaruhi oleh terjadinya A, maka *lift* lebih besar dari 1. Ketetapan *lift ratio* adalah apabila hasil perhitungan berada di bawah 1, maka *item-item* tersebut tidak menunjukkan adanya saling keterkaitan antara *antecedent* dan *consequent* [AMI-11].

## 2.7. Pengujian Akurasi

Pengujian akurasi *rule* yang terbentuk yaitu dengan membandingkan *rule* hasil proses data mining pada data latih dengan data hasil pada data uji. Penghitungan akurasi dilakukan dengan menggunakan persamaan [SAN-07] :

$$\text{Error} = \frac{|support\ data\ uji - support\ data\ latih|}{support\ data\ latih}\ x\ 100\% \quad (2.11)$$

$$Akurasi = 100\% - error \quad (2.12)$$

## 2.8. Klasifikasi

Klasifikasi adalah suatu fungsionalitas data *mining* yang akan menghasilkan model untuk memprediksi kelas atau kategori dari objek-objek di dalam basis data [KUS-07]. Proses klasifikasi biasanya dibagi menjadi dua tahap, yaitu [PRA-03] :

1. *Learning* (Pelatihan)

   Pada fase ini sebagian data yang telah diketahui kelas datanya diumpankan untuk membentuk model perkiraan.

2. *Test* (Pengujian)

   Pada fase ini model yang sudah terbentuk diuji dengan sebagian data lainnya untuk mengetahui akurasi dari model tersebut. Bila akurasinya mencukupi, model ini dapat dipakai untuk prediksi kelas data yang belum diketahui.

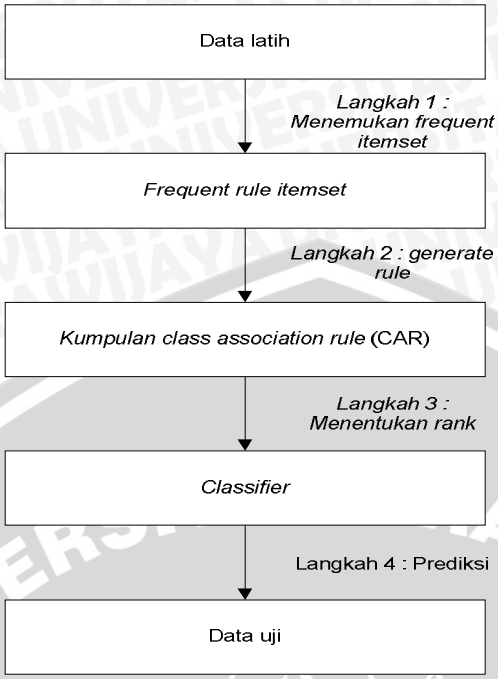
Terdapat beberapa metode klasifikasi, diantaraanya *decision tree*, Bayesian, *neural network, k-nearest neighbour, associative classification* dan lain-lain. Pada skripsi ini digunakan metode klasifikasi yaitu *associative classification*.

## 2.9. *Associative Classification*

*Associative classification* merupakan kasus khusus dalam pencarian *association rule* yang mana bagian *right-hand-side* atau konsekuen adalah sebuah atribut kelas. Salah satu keuntungan menggunakan *associative classification* yaitu bahwa algoritma ini menghasilkan aturan *if-then* atau "jika-maka" sederhana yang mudah dimengerti dan diinterpretasikan oleh *end-user* [THA-07].

Membangun sebuah *classifier* menggunakan *associative classification* dapat dibagi menjadi empat langkah sebagai berikut [THA-07] :

1. Menemukan semua *frequent rule items.*
2. Membangkitkan semua *classification association rule* (CAR) yang memiliki *confidence* di atas *minconf threshold* dari *frequent rule items* yang diekstrak pada langkah 1.
3. Pemilihan salah satu *subset* dari CAR untuk membentuk *classifier* dari yang dihasilkan pada langkah 2.
4. Mengukur kualitas dari *classifier* yang diperoleh pada objek data uji.




**Gambar 2.6. Langkah-Langkah *Associative Classification* [THA-07]**

*Rule* yang dibangkitkan akan dilatih menggunakan data latih berdasarkan pada ukuran kecocokan (disebut *discriminant function* atau DF) antara *rule* dan data latih untuk membangun *classifier*. Misalnya terdapat aturan r:F → C dan sebuah *record* d, *confidence* untuk mengklasifikasikan d dengan aturan r yaitu :

$$DF = \mu_F (d) * conf (r) \quad (2.13)$$

Jika R adalah kumpulan *rule* dan D adalah data latih maka proses membangun *classifier* dapat dilihat pada gambar 2.7.

```
1. for each d” € D” do
2. for each r € R do
3. r.DF =µF(d) * conf(r)
4. r.use=0
5. r.rightN=0
6. r.wrongN=0
7. end for
8. sort R by r.DF desc
9. flag=0
10.   while (flag=0) do
11.   r=first rule of R
12.   if r.use=1 then
13.   break
14.   r.use=1
15.   if class(r, d”) then
16.   r.rightN++
17.   flag=1
18.   break
19.   else
20.   r.wrongN++
21.   move r to the bottom of R
22.   end for
```

**Gambar 2.7. Prosedur Pelatihan *Rule* [CHE-08]**

Proses pelatihan *rule* terdiri dari tiga tahap yaitu [CHE-08] :

1. Menghitung DF setiap aturan dan mengurutkannya dari terbesar hingga terkecil (baris 2-8)
2. Memilih aturan secara beraturan hingga *record* terklasifikasi dengan benar (baris 9-22), untuk setiap aturan `RightN` dan `WrongN` digunakan untuk menyimpan jumlah data yang diklasifikasikan benar atau salah
3. Mengulangi langkah 1 dan 2 untuk setiap *record*. Aturan terburuk (*r.wk)* yaitu aturan yang nilai `RightN /(RightN+WrongN)` terkecil akan dihapus dari R.

## 2.10. Nomor Unik Pendidik dan Tenaga Kependidikan

Nomor Unik Pendidik dan Tenaga Kependidikan (NUPTK) adalah nomor identitas yang bersifat nasional untuk seluruh PTK (Pendidik dan Tenaga Kependidikan). NUPTK terdiri dari 16 angka yang bersifat tetap karena NUPTK yang dimiliki seorang PTK tidak akan berubah meskipun yang bersangkutan telah berpindah tempat mengajar atau terjadi perubahan data periwayatan [ANO-11].

NUPTK diberikan kepada seluruh PTK baik PNS maupun Non-PNS sebagai

Nomor Identitas yang resmi untuk keperluan identifikasi dalam berbagai pelaksanaan program dan kegiatan yang berkaitan dengna pendidikan dalam rangka peningkatan mutu pendidik dan tenaga kependidikan [ANO-11].

Badan Pengembangan Sumber Daya Manusia Pendidikan dan Kebudayaan dan Penjamin Mutu Pendidikan atau biasa disingkat BPSDMPPMP menjabarkan manfaat untuk tenaga pendidik yang memiki NUPTK adalah sebagai berikut :

1. Berpartisipasi dalam sebuah proses/mekanisme pendataan secara nasional sehingga dapat membantu pemerintah dalam merencanakan berbagai program peningkatan kesejahteraan bagi tenaga pendidik.
2. Mendapatkan nomor identifikasi resmi dan bersifat resmi dan bersifat nasional dalam mengikuti berbagai program/kegiatan yang diselenggarakan oleh pemerintah pusat/daerah.

Setiap kebijakan yang berhubungan dengan program-program pemberdayaan, pemberian kesejahteraan dan peningkatan kompetensi, kualifikasi, peningkatan profesionalisme, dan program sertifikasi yang diberikan oleh pemerintah pusat melalui BPSDMPPMP kepada PTK bersumber pada informasi yang diperoleh dari NUPTK [AMI-11].

## 2.11. Sertifikasi Guru

Sertifikasi guru adalah proses pemberian sertifikat pendidik kepada guru yang telah memenuhi persyaratan. Sertifikasi guru bertujuan untuk menentukan kelayakan guru dalam melaksanakan tugas sebagai pendidik profesional, meningkatkan proses dan hasil pembelajaran, meningkatkan kesejahteraan guru, serta meningkatkan martabat guru dalam rangka mewujudkan pendidikan nasional yang bermutu [ANO-09].

Dalam menentukan kelayakan seorang PTK untuk memperoleh sertifikasi, diperlukan syarat-syarat tertentu dalam menetapkan kelulusan sertifikasi guru, antara lain [ANN-12] :

1. Guru yang belum memiliki sertifikat pendidik dan masih aktif mengajar di sekolah di bawah binaan Kementerian Pendidikan dan Kebudayaan.
2. Memiliki kualifikasi akademik sarjana (S-1) atau diploma empat (D-IV).

3. Guru yang diangkat dalam jabatan pengawas dengan ketentuan:
    a) Diangkat menjadi pengawas satuan pendidikan sebelum berlakunya Peraturan Pemerintah Nomor 74 Tahun 2008 tentang Guru (1 Desember 2008)
    b) Memiliki usia setinggi-tingginya 50 tahun pada saat diangkat sebagai pengawas satuan pendidikan.
4. Guru yang belum memiliki kualifikasi akademik S-1/D-IV apabila:
    a) Pada 1 Januari 2013 sudah mencapai usia 50 tahun dan mempunyai pengalaman kerja 20 tahun sebagai guru
    b) Mempunyai golongan IV/a atau memenuhi angka kredit kumulatif setara dengan golongan IV/a (dibuktikan dengan SK kenaikan pangkat).
5. Sudah menjadi guru pada saat Undang-Undang Nomor 14 Tahun 2005 tentang Guru dan Dosen ditetapkan tanggal 30 Desember 2005.
6. Guru bukan PNS pada sekolah swasta yang memiliki SK sebagai guru tetap minimal 2 tahun secara terus menerus dari penyelenggara pendidikan (guru tetap yayasan), sedangkan guru bukan PNS pada sekolah negeri harus memiliki SK dari Bupati/Walikota.
7. Pada tanggal 1 Januari 2014 belum memasuki usia 60 tahun.
8. Sehat jasmani dan rohani dibuktikan dengan surat keterangan sehat dari dokter. Jika peserta diketahui sakit pada saat datang untuk mengikuti PLPG yang menyebabkan tidak mampu mengikuti PLPG, maka LPTK berhak melakukan pemeriksaan ulang terhadap kesehatan peserta tersebut. Jika hasil pemeriksanaan kesehatan menyatakan peserta tidak sehat, LPTK berhak menunda atau membatalkan keikutsertaannya dalam PLPG.
9. Memiliki nomor unik pendidik dan tenaga kependidikan (NUPTK).