BAB II

LANDASAN TEORI

2.1 *Market Basket Analysis*

Market Basket Analysis merupakan sebuah analisis terhadap kebiasaan belanja konsumen, dengan cara menemukan asosiasi di antara berbagai macam *item* yang dimasukkan konsumen di dalam keranjang belanja mereka (Achmad Fendi Suprastyo, 2012). Analisis ini sering diterapkan pada swalayan atau supermarket. Dalam keranjang belanja tersebut menginformasikan tentang *item-item* apa saja yang dibeli konsumen secara bersamaan yang akhirnya menuju kasir untuk membayarnya dan tersimpan dalam sistem komputerisasi yang terdapat di swalayan, secara sekilas tidak ada hal istimewa yang terjadi pada transaksi tersebut. Namun dari sisi swalayan tersebut sebenarnya telah memperoleh sebuah pengetahuan yang jarang diketahui sebelumnya yaitu tentang pola pembelian *item* yang dibeli oleh konsumen, yang dapat dilihat dari serangkaian *item-item* yang dibeli oleh konsumen.

Untuk beberapa kasus, pola dari *item-item* yang dibeli secara bersamaan oleh konsumen mudah untuk ditebak, misalnya susu dibeli bersamaan dengan roti. Namun, mungkin saja terdapat suatu pola pembelian *item* yang tidak pernah terpikirkan sebelumnya. Misalnya, pembelian minyak goreng dengan deterjen. Mungkin saja pola seperti ini tidak pernah terpikirkan sebelumnya karena minyak goreng dan deterjen tidak mempunyai hubungan, baik sebagai barang pelengkap maupun barang pengganti. Hal ini mungkin tidak pernah terpikirkan sebelumnya sehingga tidak dapat diantisipasi jika terjadi sesuatu, seperti kekurangan stok. Inilah salah satu manfaat yang dapat diperoleh dari *Market Basket Analysis*.

Dengan kata lain tujuan *Market Basket Analysis* adalah untuk merancang strategi penjualan atau pemasaran dengan memanfaatkan data transaksi penjualan yang telah tersedia di perusahaan, diantaranya :

Dengan mengubah *layout* toko, mendekatkan kategori-kategori *item* yang sering dibeli bersamaan pada satu waktu. Memberikan diskon pada pembelian paket *item* yang jarang dibeli dan juga dapat mengetahui item paling laku terjual dari periode tertu.

Market Basket Analysis mempunyai sebuah prosedur yaitu *Association Rule* (Achmad Fendi Suprastyo, 2012), penjelasan mengenai *Association Rule* dapat dilihat pada subbab berikutnya. Pada penelitian ini, data yang digunakan berasal dari data transaksi penjualan 4 bulan terakhir bulan januari yang tersimpan pada sistem komputerisasi CV.Anugrah Jaya Muliya.

2.2 Transaksi

Transaksi adalah satu atau beberapa aksi program aplikasi yang mengakses/mengubah *isi basisdata*.(Dita Fauzia, 2011). Transaksi merupakan bagian dari pengeksekusian sebuah program yang melakukan pengaksesan basis data dan bahkan juga melakukan serangkaian perubahan data. DBMS yang kita gunakan harus menjamin bahwa setiap transaksi harus dapat dikerjakan secara utuh atau tidak sama sekali. Tidak boleh ada transaksi yang hanya dikerjakan sebagian, karena dapat menyebabkan inkonsistensi basis data. Untuk itu transaksi selalu merubah basis data dari satu kondisi konsisten ke kondisi konsisten lain.

Sebuah transaksi berpeluang untuk ‘mengganggu’ integritas basis data yang dapat membuat kondisi/hubungan antar data tidak seperti seharusnya.

Transaksi bertujuan untuk mencegah dari kehilangan ataupun kerusakan data. Untuk menjamin agar integritas dapat tetap terpelihara maka setiap transaksi harus memiliki sifat-sifat:

1. Atomicity, dimana semua operasi dalam transaksi dapat dikerjakan seluruhnya atau tidak sama sekali.

2. Consistency, dimana eksekusi transaksi harus dapat menjamin data tetap konsisten setelah transaksi berakhir.

3. Isolation, jika pada sebuah sistem basis data terdapat sejumlah transaksi yang dilaksanakan secara bersamaan, maka semua transaksi yang dilaksanakan pada saat yang bersamaan tersebut harus dapat dimulai dan bisa berakhir.

4. Durability, dimana perubahan data yang terjadi setelah sebuah transaksi berakhir dengan baik, harus dapat bertahan bahkan jika seandainya sistem mati.

Terhentinya suatu transaksi tidak selalu diakibatkan oleh kegagalan insidental baik dari perangkat keras (crash) ataupun kemacetan sistem operasi (hang). Tapi lebih sering terjadi karena user sengaja menghentikan transaksi atau karena penghentian transaksi oleh DBMS akibat adanya kondisi tak diinginkan, seperti deadlock atau timeout.

Sebuah transaksi dapat menghasilkan dua kemungkinan:

a. Jika dilaksanakan lengkap seluruhnya, transaksi tersebut telah di *commit* dan basis data mencapai keadaan konsisten baru.
b. Jika transaksi tidak sukses, maka transaksi dibatalkan dan basis data dikembalikan ke keadaan konsisten sebelumnya (*rollback*).

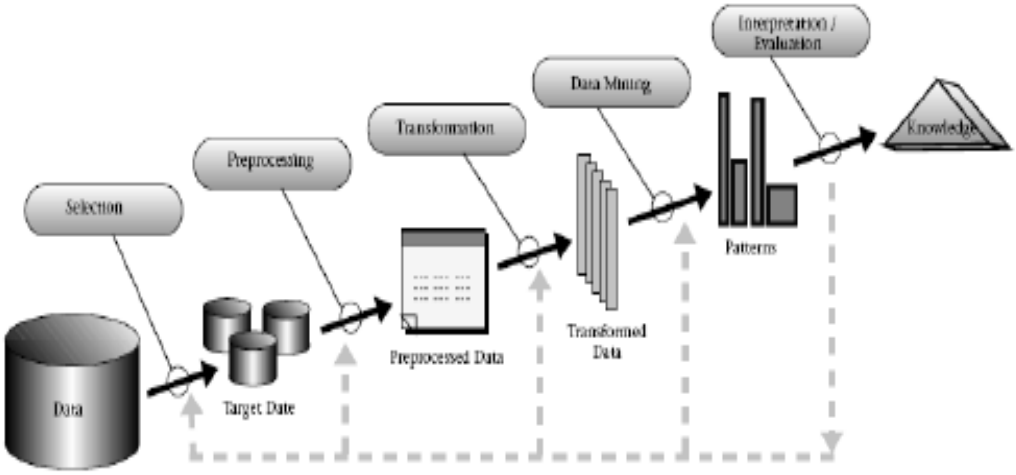
Transaksi yang sudah di commit tidak dapat dibatalkan lagi. Jika ada kesalahan, maka harus dilakukan transaksi lain yang membalik dampak transaksi sebelumnya (Dita Fauzia, 2011).

2.3 Knowledge discovery in databases (KDD)

Knowledge discovery in databases (KDD) adalah proses untuk menemukan interesting knowledge dari sejumlah besar data yang disimpan baik didalam databases, data warehouses atau tempat penyimpanan informasi lainnya (Gunawan, 2006).

Knowledge discovery in databases (KDD) berhubungan dengan teknik integrasi dan penemuan ilmiah, interprestasi dan visualisasi dari pola-pola sejumlah kumpulan data. Knowledge discoveryin databases (KDD) adalah keseluruhan proses non-trivial untuk mencari dan mengidentifikasi pola (pattern) dalam data, dimana pola yang ditemukan bersifat sah, baru, dapat

bermanfaat dan dapat dimengerti. (Gunawan, 2006). Proses dari KDD dapat dilihat pada gambar 2.1 Fayyad(1996):



Gambar 2.1 Tahap *Knowledge Discovery in Databases* (KDD)

Sebagai suatu rangkaian proses, KDD dapat dibagi menjadi beberapa tahap yang diilustrasikan di Gambar 2.1. Tahap-tahap tersebut. bersifat interaktif di mana pemakai terlibat langsung atau dengan perantaraan *knowledge base.* Proses *knowledge discovery in databases* (KDD) dibagi dalam beberapa tahap (Tessy, 2010) .

1. Selection

Melakukan reduksi terhadap data yang ada di dalam *database*. Proses reduksi diperlukan untuk mendapatkan hasil yang lebih akurat dan mengurangi waktu komputasi terutama utuk masalah dengan skala besar (*large scaleproblem*). Beberapa cara seleksi, antara lain:

a) *Sampling*, adalah seleksi subset representatif dari populasi data yang besar.
b) *Denoising*, adalah proses menghilangkan *noise* dari data yang akan ditransformasikan.
c) *Feature extraction*, adalah proses membuka spesifikasi data yang signifikan dalam konteks tertentu.

2. *Preprocessing*

Pre-processing data adalah tahapan sebelum suatu data diproses, dapat berupa pembersihan data, transformasi data, atau yang lainnya.

a. *Data cleaning* (Pembersihan data)

Proses ini digunakan untuk membuang data yang tidak konsisten dan bersifat *noise* dari data yang terdapat di berbagai basis data yang mungkin berbeda format maupun *platform* yang kemudian diintegrasikan dalam satu *database data warehouse*.

Tahap data *cleaning* :

1. Menangani *missing values* (nilai-nilai yang hilang)
2. *Smoothing* (menghaluskan) data yang *noise*. *Noise* adalah kesalahan yang terjadi secara random atau karena variasi yang terjadi dalam pengukuran variabel. Penangan *noise* pada data dapat diatasi dengan *Binning, Clustering, dan Regression.*
3. Menjadikan data lebih konsisten.

b. Intergrasi Data

Integritas data merupakan suatu kondisi dimana semua nila-nilai yang dimasukkan atau disimpan didalam *database* telah sesuai dengan aturan yang ditetapkan untuk data tersebut. Jika data yang dimasukkan belum sesuai dengan ketentuan maka bisa dikatakan *database* belum terintegritasi.Batasan dalam penetuan integritas data :

3. Transformasi data

Beberapa teknik data mining membutuhkan format data yang khusus sebelum bisa diaplikasikan. Sebagai contoh beberapa teknik standar seperti analisis asosiasi dan klastering hanya bisa menerima input data kategorikal.

Transformasi data diperlukan sebagai tahap *pre-procecing*, dimana data yang diolah siap untuk ditambang. Beberapa cara transformsi, antara lain (Santosa, 2007):

A. *Centering*, mengurangi setiap data dengan rata-rata dari setiap atribut yang ada.
B. *Normalisation*, membagi setiap data yang di*centering* dengan standar deviasi dari atribut bersangkutan.
C. *Scaling*, mengubah data sehingga berada dalam skala tertentu diproses lebih lanjut.

4. Aplikasi Teknik Data Mining

Data mining merupakan proses untuk mencari pola atau informasi menarik dalam data terpilih dengan menggunakan teknik atau metode tertentu. Teknik, metode, atau algoritma dalam data mining sangat bervariasi. Pemilihan metode atau algoritma yang tepat sangat bergantung pada tujuan dan proses KDD secara keseluruhan.

5. Evaluasi pola yang ditemukan

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

6. Presentasi Pengetahuan

Presentasi pola yang ditemukan untuk menghasilkan aksi tahap terakhir dari proses data mining adalah bagaimana memformulasikan keputusan atau aksi dari hasil analisa yang didapat.

2.4 Konsep Data Mining

Secara sederhana data mining adalah ekstraksi informasi atau pola yang penting atau menarik dari data yang ada di *database*. *Data mining* merupakan bagian dari proses *Knowledge Discovery in Databases* (KDD).(Istiana, 2011) Banyak pihak-pihak lain yang telah mendefiniskan *data mining*. Berikut beberapa definisi *data mining*:

A. *Data Mining* adalah suatu istilah yang digunakan untuk menguraikan penemuan pengetahuan di dalam *database* yang prosesnya menggunakan teknik statistik, matematika, kecerdasan buatan, dan *machine learning*, untuk mengekstraksi dan mengidentifikasi informasi yang bermanfaat dan pengetahuan yang terkait dari berbagai *database* (Turban, dkk. 2005).

B. *Data Mining* merupakan bidang dari beberapa bidang keilmuan yang menyatukan teknik dari pembelajaran mesin, pengenalan pola, statistik, *database*, dan visualisasi untuk penanganan permasalahan pengambilan informasi dari *database* yang besar (Larose, 2005).

C. *Data mining* adalah ekstraksi informasi atau pola yang penting untuk menarik dari data yang ada di *database* yang benar sehingga menjadi informasi yang sangat berharga (Sucahyo, 2004).

Berdasarkan beberapa pengertian diatas dapat ditarik kesimpulan bahwa *data mining* merupakan proses untuk menggali nilai tambah dari suatu kumpulan data berupa pengetahuan yang selama ini tidak diketahui secara manual. Tujuan dari data mining adalah menemukan hubungan-hubungan atau pola-pola yang mungkin memberikan indikasi yang bermanfaat.

Data Mining merupakan konsep baru dalam penemuan nilai tambah dari sebuah data. Dengan data mining di dapatkan informasi-informasi yang selama ini tersembunyi dengan penemuan pola-pola dalam sebuah data. Bahkan dengan menggunakan *Data Mining* dapat memprediksi prilaku dan tren yang terjadi dalam perusahaannya atau memprediksi pola kebiasaan pelanggan sebagai konsumennya sehingga bisa membuat para pengusaha menjadi lebih proaktif dan dapat mengambil keputusan dengan benar.

2.4.1 Fungsi *Data Mining*

Fungsi *data mining* dapat diklasifikasikan menjadi dua kategori, yaitu: deskriptif dan prediktif. Yaitu (Tessy, 2010) :

2.4.1.1 Fungsi Prediktif

Model data mining prediktif Menggunakan beberapa variabel dari *database* untuk memprediksi suatu nilai yang akan datang atau untuk meramal masa depan. Pemodelan Prediksi memungkinkan user untuk mengirimkan record dengan beberapa field kosong, dan sistem akan menebak nilai yang kosong tersebut dengan pola-pola sebelumnya yang ditemukan dari basis data. Model prediktif diantaranya klasifikasi data, *decision tree, analisis time series, regresi*, prediksi, dan jaringan syaraf tiruan.

2.4.1.2 Fungsi Deskripsi

Melakukan ekstraksi data untuk penemuan informasi data yang tidak dapat dilakukakan secara manual. Hal pertama yang dilakukan yaitu pencarian norm dari data tersebut, kemudian mendeteksi item-item deviasi dari dari data yang biasa (usual) dengan *frequensi item* yang telah diberikan.

Mendapatkan pola penafsiran *(human-interpretable patterns)* untuk menjelaskan data. Pola dan trend data sering dideskripsikan. Deskripsi tersebut sangat membantu dalam menjelaskan pola dan trend yang terjadi. Model *data mining* harus setransparan mungkin, dimana hasilnya dapat mendeskripsikan pola dengan jelas. *Clustering data, summarization, association rule, sequence discovery* merupakan bagian dari metode deskripsi. Melalui tugas *descriptive mining,* dapat dilakukan penggolongan data dalam *database,* sedangkan melalui tugas *predictive mining,*yang ada dapat digunakan untuk membuat suatu prediksi.

Fungsi data mining dan jenis-jenis pola yang dapat ditemukan adalah sebagai berikut (Han, Kamber, 2001) :

1) ***Concept/Class Description: Characterization and Discrimination***

   Data dapat diasosiasikan dengan *class* atau *concept*. *Concept/class description* dapat diperoleh melalui *data characterization*, data

discrimination, atau kedua-duanya. Data *characterization* adalah ringkasan dari karakter atau ciri umum dari *target class*. Sedangkan data *discrimination* adalah perbandingan ciriciri umum dari *target class* dari data *object* dengan ciri-ciri umum dari *object* dari satu atau serangkaian *class* yang kontras.

2) ***Association Analysis***

Association analysis adalah penemuan *association rule* yang menunjukkan pola-pola yang sering muncul dalam data. Terdapat nilai *support* dan *confidence* yang dapat menunjukkan seberapa besar suatu *rule* dapat dipercaya. *Support* adalah ukuran dimana seberapa besar tingkat dominasi suatu *item* atau *itemset* terhadap keseluruhan transaksi. Cara perhitungannya adalah dengan rumus:

$$support\ (A \rightarrow B) = \rho\ (A\ Y\ B) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ (2.1)$$

Sedangkan *confidence* adalah ukuran yang menunjukkan hubungan antara dua *item* secara *conditional*. Cara perhitungannya adalah dengan rumus:

$$confidance\ (A \rightarrow B) = p\frac{(A\ Y\ B)}{p(A)} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ (2.1)$$

Untuk dapat lebih memahami apa yang dimaksud dengan *support* dan *confidence*, dapat dilihat contoh di bawah ini:

membeli(T,komputer) → membeli (T, Softwere)
[*support* 1 % *confidance* 50 %]

Arti dari *rule* di atas adalah jika pada sebuah transaksi, T, membeli“*computer*”, ada peluang sebesar 50% bahwa pada transaksi

tersebut juga membeli "*software*", dan pada keseluruhan transaksi terdapat peluang 1% keduanya sama-sama dibeli.

3) Classification and Prediction

Classification adalah proses menemukan model yang mendeskripsikan *class* dan *concept* dari data. *Classification* juga dapat digunakan untuk memprediksi *class label* dari *data object*. Pada banyak aplikasi, *user* lebih menginginkan memprediksi *missing* atau *unavailable data value* daripada *class label*. Hal ini biasanya terjadi pada kasus dimana *value* yang akan diprediksi adalah berupa data numerik.

4) Cluster Analysis

Berbeda dengan *classification* dan *prediction*, *cluster analysis* dilakukan tanpa mengetahui *class label*. *Cluster* dari *object* dibentuk jika *object* di dalam suatu *cluster* memiliki kemiripan yang tinggi dengannya, dan memiliki ketidakmiripan dengan *object* di *cluster* lainnya.

5) Outlier Analysis

Sebuah *database* dapat mengandung data *object* yang tidak sesuai atau menyimpang dari model data. *Data object* ini disebut *outlier*. Banyak metode data mining yang menghilangkan *outlier* ini. Padahal, pada beberapa aplikasi seperti *fraud detection*, kejadian yang jarang terjadi justru lebih menarik untuk dianalisa daripada kejadian yang sering terjadi. Analisa dari *outlier* data disebut sebagai *outlier mining*.

6) Evolution Analysis

Data evolution analysis mendeskripsikan model yang beraturan atau *trend* untuk *object* yang sifatnya terus menerus berubah.

2.5 *Association Rule Mining*

Association rule mining merupakan salah satu model data mining deskriptif. *Association rule* adalah suatu prosedur untuk mencari hubungan antar *item* dalam suatu *data set* yang ditentukan.

Association rule meliputi dua tahap (Han,Kamber, 2001) :

1. Mencari kombinasi yang paling sering terjadi dari suatu *itemset*.
2. Mendefinisikan *condition* dan *result* untuk *conditional association rule*.

2.5.1 Ukuran *Association Rule Mining*

Dalam menentukan suatu *association rule*, terdapat ukuran yang menyatakan bahwa suatu informasi atau *knowledge* dianggap menarik (*interestingness measure*). Ukuran ini didapatkan dari hasil pengolahan data dengan perhitungan tertentu. Umumnya ada dua ukuran keprcayaan (*interestingness measure)* yang digunakan dalam menentukan suatu *association rule*, yaitu (Han, kamber, 2001) :

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*-nya (misalnya, dari keseluruhan transaksi yang ada, seberapa besar tingkat dominasi yang menunjukkan bahwa *item* A dan B dibeli bersamaan).

$$support\ (A \rightarrow B) = (A \cup B) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.3)$$

2. ***Confidence***

Suatu ukuran yang menunjukkan hubungan antar dua *item* secara *conditional* (misalnya, seberapa sering *item* B dibeli jika orang membeli *item* A). Perhitungan *confidence* menggunakan rumus :

$$confidence\ (A \cup B) = \frac{support\ count\ (A \cup B)}{support\ count\ (A)} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.4)$$

support dan *confidence* nantinya berguna dalam menentukan *interesting association rules*, yaitu untuk dibandingkan dengan batasan (*treshold*) yang ditentukan oleh *user*. Batasan tersebut umumnya terdiri dari *min_support* dan *min_confidence*. Bila memenuhi kedua batasan maka sebuah *rule* dapat disebut *interesting rule*.

Minimum *support count* adalah nilai minimum transaksi yang terlibat dalam setiap pembelian *itemset* (group variasi produk). Sedangkan *confidence* adalah besar nilai keyakinan atau kepastian bahwa suatu *itemset* lain akan turut pada saat bersamaan dalam suatu *itemset* tertentu.

Aturan asosiasi menggambarkan item atau kejadian dalam data berukuran besar yang berisi data transaksi. Dengan kemajuan teknologi, data penjualan dapat disimpan dalam jumlah besar yang disebut dengan "basket data." Aturan asosiasi yang didefinisikan pada basket data, digunakan untuk keperluan promosi, desain katalog, segmentasi customer dan target pemasaran. Secara tradisional, aturan asosiasi digunakan untuk menemukan trend bisnis dengan menganalisa transaksi
customer.

Akan tetapi karena konsep *association rule* merupakan konsep yang mencari frequensi suatu pola yang muncul secara bersamaan, *Association rule* tidaka hanya di terapkan untuk *market basket analysis* hal ini juga dapat diterapkan dalam data absensi karyawan yakni mencari pola kebiasaan absensi karyawan (Istiana, 2011), dan hal ini juga dapat di terapkan pada data transaksi penjualan pada prusahaan yang bergerak dibidang eksport garmen, mencari dan menentukan produk terunggul untuk mengembangkan produk produk yang terlaris guna meminimalisir kurun waktu yang menandang.

2.6 Pembangunan *Frequent Pattern tree (FP-TREE)*

Algoritma *FP-Growth* merepresentasikan transaksi dengan menggunakan struktur data *FP-Tree*. *FP-tree* merupakan struktur penyimpanan data yang dimampatkan (data yang disimpan secara ditimpakan). *FP-tree* dibangun dengan memetakan setiap data transaksi ke dalam setiap lintasan tertentu dalam *FP-tree*. Karena dalam setiap transaksi yang dipetakan, mungkin ada transaksi yang memiliki item yang sama, maka lintasannya memungkinkan untuk saling menimpa. Semakin banyak data transaksi yang memiliki item yang sama, maka proses pemampatan dengan struktur data *FP-tree* semakin efektif. Kelebihan dari
FP-tree adalah hanya memerlukan dua kali tahapan data transaksi yang terbukti sangat efisien. (Samuel, 2008)

Proses penyusunan *FP-Tree* dari mulai representasi awal transaksi, pengurutan dengan hanya mempertahankan *frequent 1-itemset*, dan penyimpanannya di *FP-Tree.* Setelah *FP-Tree* terbentuk, langkah selanjutnya adalah memperoleh *frequent itemset* tanpa melakukan *candidates generation.* Misal I= {a1, a2, …, an} adalah kumpulan dari item. Dan basis data transaksi DB = {T1, T2, …, Tn}, di mana Ti (i € [1..n]) adalah sekumpulan transaksi yang mengandung item di I. Sedangkan *support* adalah penghitung (*counter*) frekuensi kemunculan transaksi yang mengandung suatu pola. Suatu pola dikatakan sering muncul (*frequent pattern*) apabila *support* dari pola tersebut tidak kurang dari suatu konstanta ξ (batas ambang minimum *support*) yang telah didefinisikan sebelumnya. Permasalahan mencari pola frequent dengan batas ambang minimum *support count* ξ inilah yang dicoba untuk dipecahkan oleh *FP-Growth* dengan bantuan Struktur *FP-tree*.

Adapun *FP- tree* adalah sebuah pohon dengan definisi sebagai berikut (Samuel, 2008) :

FP-Tree dibentuk oleh sebuah akar yang diberi label *null*, sekumpulan upapohon yang beranggotakan item-item tertentu, dan sebuah tabel *frequent header*. Setiap simpul dalam *FP-tree* mengandung tiga informasi penting,

yaitu label item, menginformasikan jenis item yang direpresentasikan simpul tersebut, *support count*, merepresentasikan jumlah lintasan transaksi yang melalui simpul tesebut, dan pointer penghubung yang menghubungkan simpul-simpul dengan label item sama antar-lintasan, dintandai dengan garis panah putus-putus.

2.7 Algoritma *Frequent Pattern Growth* (*FP-GROWTH)*

Setelah tahap pembangunan *FP-tree* dari sekumpulan data transaksi, akan diterapkan algoritma *FP-growth* untuk mencari *frequent itemset* yang signifikan.

Informasi yang disimpan sebuah *node FP-Tree* berupa *Item, Index parent, Support,* dan *Next* (*Pointer*). Ketika selesai membuat *FP-Tree*, kita tidak begitu saja bisa mendapatkan *frequent itemset* yang terdapat dalam *dataset*. Suatu kombinasi *itemset* bisa saja berada di beberapa *path* yang berbeda. Untuk mendapatkan suatu *pattern* dalam *FP-Growth* langkah yang lebih mudah adalah mencari arah dari ujung suatu *path*, kemudian kita mencari mulai dari *header* untuk *item* di ujung tersebut, barulah kemudian dibuat berdasarkan tiap *node* berisi *item* tersebut dicari arah *path node* ke atas.

Hal ini tentu lebih cepat dari pada *up-down* karena *pointer* langsung yang dimiliki tiap *node* adalah *pointer* ke *parent*. *Path-path* yang dieksplorasi hanyalah *path-path* yang memiliki *node* yang sedang dicari. Jadi dalam struktur *FP-Tree* ada *link* dari suatu *item* ke *path-path* yang memiliki *item* tersebut, sehingga ketika dibutuhkan pencarian *pattern* untuk suatu *item* tertentu, hanya mencari dari *path-path* tersebut saja.

Algoritma *FP-Growth* merupakan pengembangan dari algoritma Apriori. Algoritma *Frequent Pattern Growth* adalah salah satu alternatif algoritma yang dapat digunakan untuk menentukan himpunan data yang paling sering muncul (*frequent itemset*) dalam sebuah kumpulan data. Pada algoritma *FP-Growth* menggunakan konsep pembangunan *tree*, yang biasa disebut *FP-Tree*, dalam pencarian *frequent itemsets* bukan menggunakan *generate candidate* seperti yang dilakukan pada algoritma *Apriori*. Dengan

menggunakan konsep tersebut, algoritma *FP-Growth* menjadi lebih cepat daripada algoritma *Apriori*.

Algoritma *FP-growth* dibagi menjadi tiga langkah utama, yaitu (Soelaiman, 2006) :

1. Tahap Pembangkitan *Conditional Pattern Base*

   Conditional Pattern Base merupakan sub*database* yang berisi *prefix path* (lintasan prefix) dan *suffix pattern* (pola akhiran). Pembangkitan *conditional pattern base* didapatkan melalui *FP-tree* yang telah dibangun sebelumnya.

2. Tahap Pembangkitan *Conditional FP-tree*

   Pada tahap ini, *support count* dari setiap item pada setiap *conditional pattern base* dijumlahkan, lalu setiap item yang memiliki jumlah support count lebih besar sama dengan minimum *support count* ξ akan dibangkitkan dengan *conditional FP-tree.*

3. Tahap Pencarian *frequent itemset*

   Apabila *Conditional FP-tree* merupakan lintasan tunggal (*single path),* maka didapatkan *frequent itemset* dengan melakukan kombinasi item untuk setiap *conditional FP-tree*. Jika bukan lintasan tunggal, maka dilakukan pembangkitan *FP-growth* secara rekursif. Ketiga tahap tersebut merupakan langkah yang akan dilakukan untuk mendapat *frequent itemset*, yang dapat dilihat pada algoritma berikut : (Soelaiman, 2006).

Input: Output: R sekumpulan lengkap pola

t frequent

growth(Tree,null)

Method: FP-Procedure: FP-growth(Tree,α) {

01:ee mengandung *single path* P; if Tr02: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 tuk tiap a_i dalam header dari

unTree do {

05:bangkitkan pola

06:bangun β= a_i α dengan

support = a .support

i07:if ee β=O

Tr08:then panggil FP-growth(Tree,β) } }

Gambar 2.2 Algoritma *FP-Growth*

2.8 Contoh Implementasi metode *FP-Growth* pada usaha retail

1. *Dataset*

Tahap ini adalah tahap dimana memilih *dataset* yang digunakan untuk proses *data mining* (Dyah, Dkk 2014).

Tabel 2.1 Contoh Tabel Penjualan barang untuk analisa

Tanggal	No. Transaksi	Nama Barang
10/2/2013	21020130001	Gula
10/2/2013	21020130001	Kopi
10/2/2013	21020130001	Teh
10/2/2013	21020130002	Gula
10/2/2013	21020130002	Kopi
10/2/2013	21020130002	Teh
10/2/2013	21020130003	Gula
10/2/2013	21020130003	Susu
10/2/2013	21020130003	Roti
10/2/2013	21020130004	Roti
10/2/2013	21020130004	Gula
10/2/2013	21020130004	Air
10/2/2013	21020130005	Gula
10/2/2013	21020130005	Susu
10/2/2013	21020130005	Kopi

2. FP-Tree

Tahap ini adalah tahap dimana dataset yang telah dibatasi dengan menggunakan *support count* yang telah ditentukan, kemudian dibangun menjadi sebuah *Tree*. Berikut ini adalah tabel dengan semua barang yang dalam satu transaksi sudah pindai.

Tabel 2.2 Data Transaksi yang disederhanakan

No. Transaksi	Nama Barang
21020130001	Gula, Kopi, Teh
21020130002	Gula, Kopi, Teh
21020130003	Gula, Susu, Roti
21020130004	Roti, Gula, Air
21020130005	Gula, Susu, Kopi

Kemudian menentukan frekuensi setiap *item* dari keseluruhan transaksi.

Tabel 2.3 Frekuensi kemunculan item

Nama Barang	Jumlah
Gula	5
Kopi	3
Teh	2
Susu	2
Roti	2
Air	1

Setelah frekuensi setiap *item* diperoleh, kemudian dibatasi dengan *support count*. Jika frekuensi *item* tidak kurang dari *support count*, maka *item* tersebut akan dihapus dan tidak dipakai dalam proses *data mining*. Misalkan ditentukan *support count* ξ =2, maka hasilnya adalah:

Tabel 2.4 Frekuensi Item Setelah Proses Filter

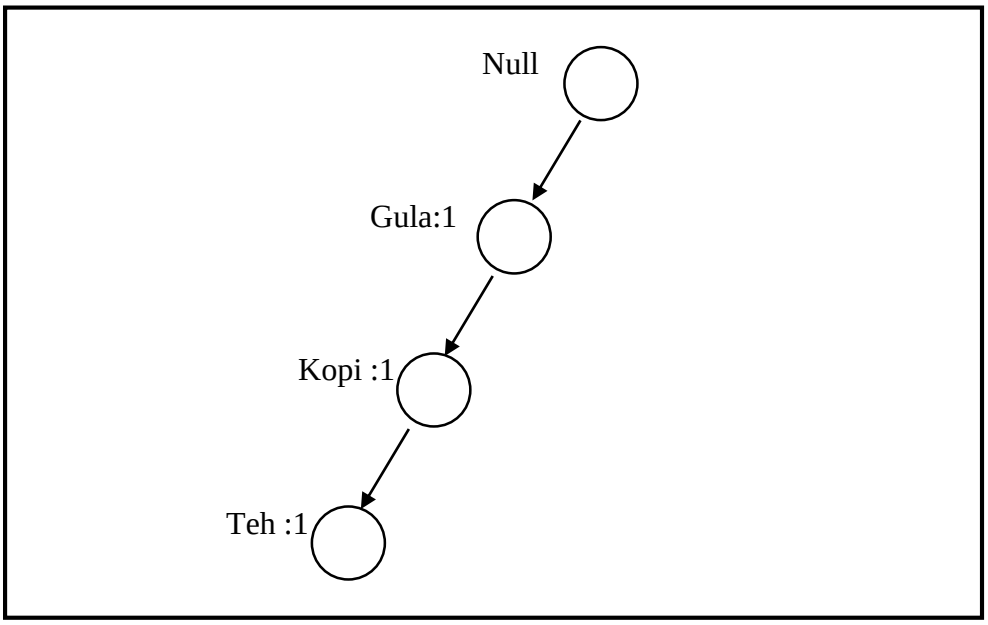
Nama Barang	Jumlah
Gula	5
Kopi	3
Roti	2
Susu	2
Teh	2

Item Air hilang karena frekuensinya tidak lebih dari sama dengan 2. Tahap selanjutnya adalah pembangunan *Tree* berdasarkan per transaksi dengan item yang telah dibatasi.

Tabel 2.5 Data Transaksi Setelah Proses *Filter*

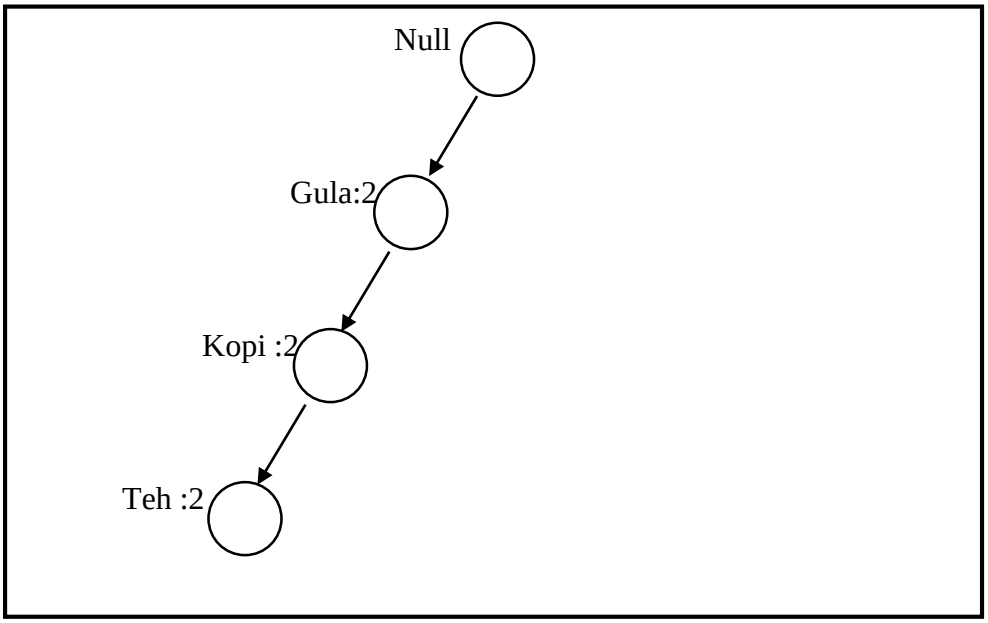
No. Transaksi	Nama Barang
21020130001	Gula, Kopi, Teh
21020130002	Gula, Kopi, Teh
21020130003	Gula, Roti, Susu
21020130004	Gula, Roti
21020130005	Gula, Kopi, Susu

Berikut ini memberikan ilustrasi mengenai pembentukan FP-tree setelah pembacaan no transaksi yang pertama 30001, yaitu (gula, kopi, teh) dengan support count awal bernilai 1 sehingga membentuk lintasan transakasi seperti gambar 2.3 :



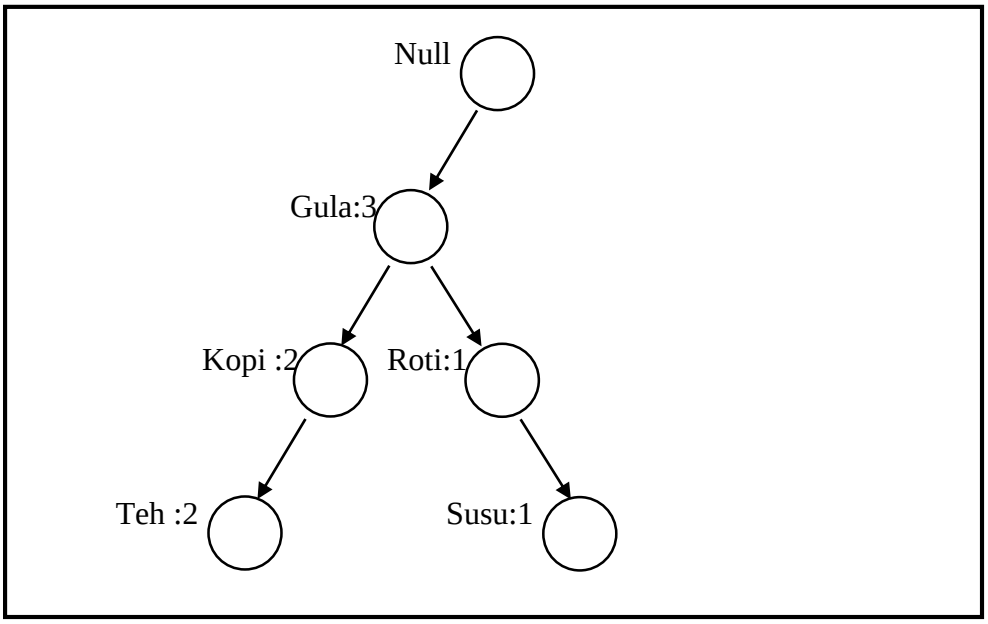
Gambar 2.3 Pembacaan No transaksi 30001

Setelah pembacaan no transaksi yang kedua yaitu (gula, kopi, teh), karna transaksi yang kedua memiliki prefix transaksi yang sama dengan transaksi yang pertama yaitu (gula, kopi, teh) maka lintasan yang kedua dapat ditimpahkan, sehinnga menambah nilai *support count* dari (gula,kopi,teh). Ilustrasi dapat dilihat seperti gambar 2.4:



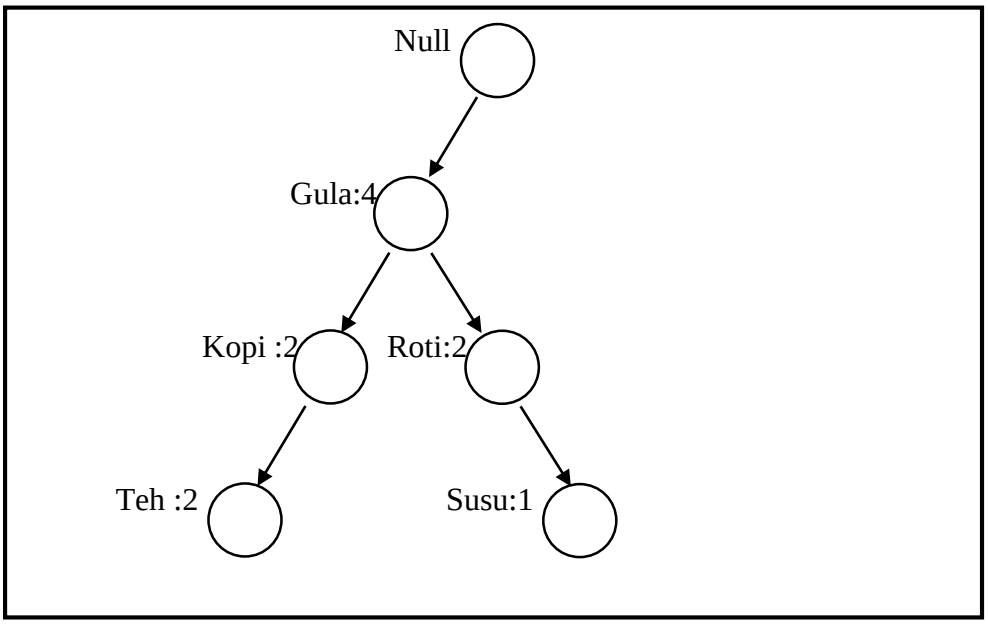
Gambar 2.4 Pembacaan No transaksi 30002

Setelah pembacaan transaksi yang ketiga yaitu (gula, roti. Susu), karna memiliki prifix yang sama yaitu (gula) maka lintasan transakasi ketiga dapat ditimpakan di (gula) sehinnga menambah nilai suppor count (gula), dan selanjutnta membuat lintasan baru dari (gula) -(roti, susu) seperti gambar 2.5 :



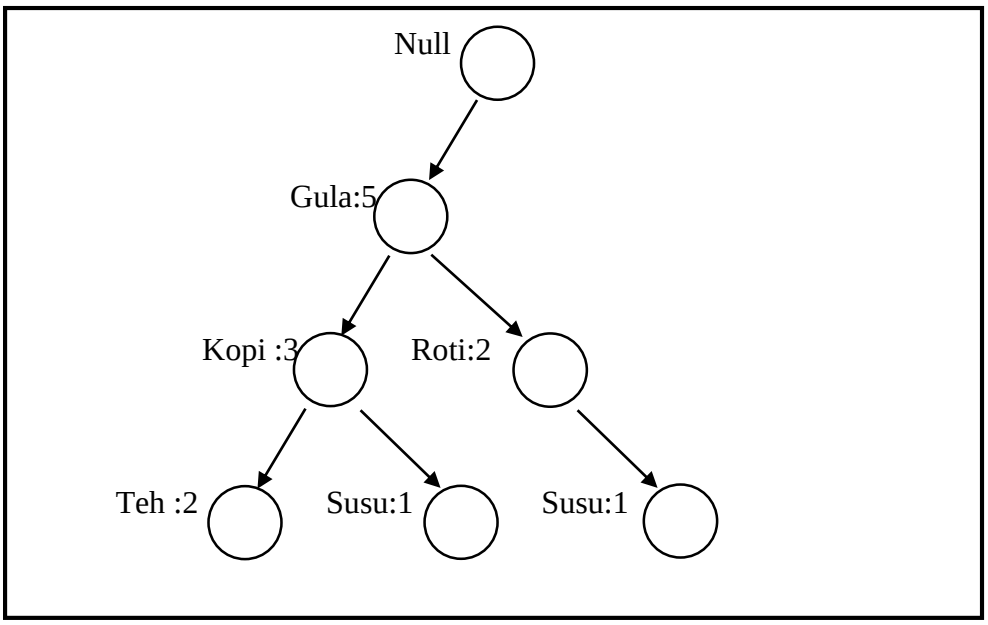
Gambar 2.5 Pembacaan No transaksi 30003

Setelah pembacaan no transaksi yang keempat yaitu (gula, roti) karna transaksi keempat memiliki lintasan awal yang sama yaitu (gula, roti), maka lintasan keempat dapat ditimpahakn di (gula, roti) sehingga menambah nilai support count (gula, roti) seperti gambar 2.6 :



Gambar 2.6 Pembacaan No transaksi 30004

Setelah pembacaan transaksi kelima yaitu (gula, kopi, susu) karna lintasan transaksi kelima memiliki lintasan awal yang sama yaitu (gula, kopi) maka lintasan dapat ditimpahkan di (gula, kopi) sehingga menambah jumlah nilai support count (gula, kopi) dan selanjutnya membentuk lintasan baru seperti gambar 2.7 :



Gambar 2.7 Pembacaan No transaksi 30005

3. FP-Growth

Setelah FP-Tree terbentuk, maka langkah selanjutnya adalah tahap pembangkitan *conditional pattern base*, tahap pembangkitan *conditional FP-Tree*, dan tahap pencarian *frequent itemset*. Pada tahap ini dapat dilakukan dengan melihat kembali FP-Tree yang sudah dibuat sebelumnya. (Achmad Fendi Suprastyo, 2012)

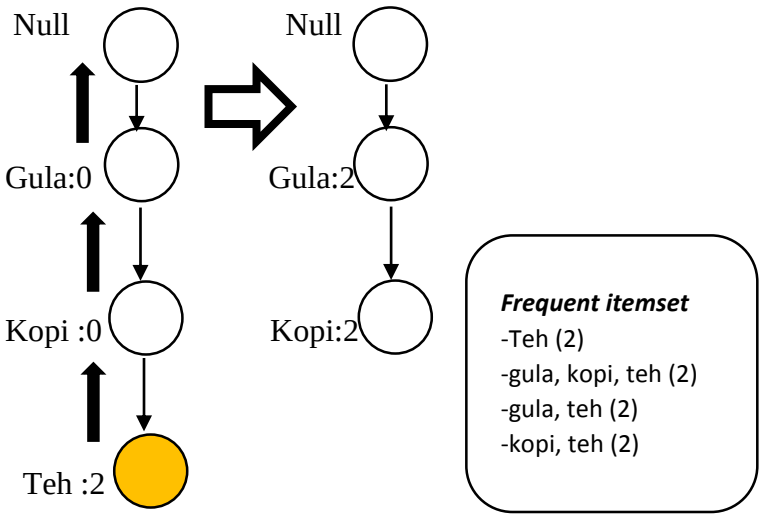
Berikut tiga langkah utama Algoritma FP-Growth yaitu :

1. Pembangkitan ***conditional pattern base***, Algoritma *FP-Growth* menemukan frequent itemset yang berakhiran *suffix* menggunakan metode *divide and conquere* untuk memecah problem menjadi subproblem yang lebih kecil. Misalkan Jika kita ingin menemukan semua frekuensi itemset yang berakhiran ke salah satu item, hal yang pertama dilakukan adalah pembangkitan conditional pattern base pada sub *database* yang berisikan *prefix path* (himpunan item terurut yang mengawali k-itemsets), dan *suffix pattern* (k-itemsets).

2. Pembangkitan ***conditional FP-Tree***, dan mencari *frequent itemset*, dengan menerapkan metode *Divide and Conquer* sesuai urutan *Frequent List* dari yang paling kecil jumlah kemunculannya.

1. Condisi *FP-Tree* untuk item teh

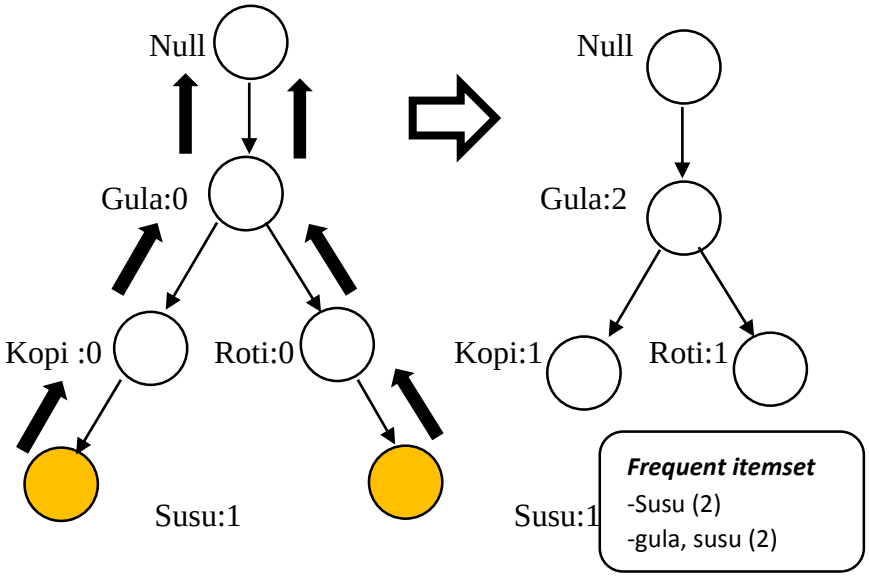
Langkah awal, ekstrak semua lintasan yang berakhiran teh. Selain lintasan teh diberi nilai 0, hal ini dilakukan agar dapat mengetahui informasi berapa kali *item* yang lain dibeli bersamaan dengan *item* teh dan bisa mengetahui *frequent itemsets* mana yang memenuhi syarat minimum *support*. Untuk lebih memperjelas, dapat dilihat contoh menemukan *frequent itemset* yang berakhiran dengan *item* teh dapat dilihat pada gambar 2.8



Gambar 2.8 Kondisi *FP-Tree* untuk item teh

Setelah itu, naikkan satu persatu lintasan teh sampai ke *null* dan nilai lintasan teh dimasukan ke setiap lintasan yang dilintasi sampai ke *null*. Pada kasus ini, *item* gula, kopi nilai kemunculan bersamaan dengan *item* teh sebayk 2 kali, maka *subsets* yang terbentuk dari *item* teh adalah{teh} {kopi,teh},{gula,kopi,teh},{gula,teh}

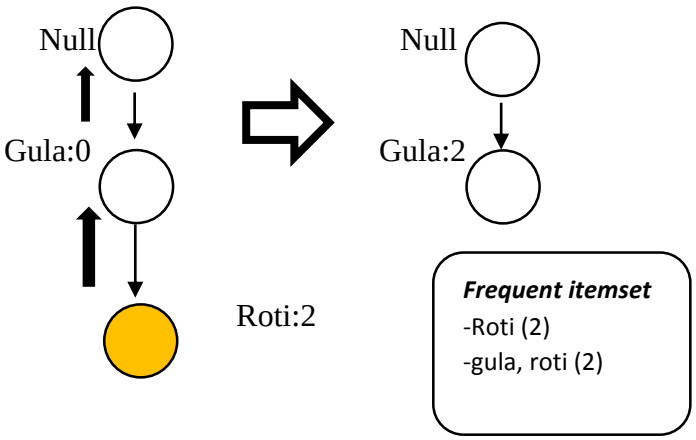
2 Kondisi *FP-Tree* untuk item susu



Gambar 2.9 Kondisi *FP-Tree* untuk item susu

Pada kasus ini, *item* roti dan kopi nilai kemunculan bersamaan dengan *item* susu hanya 1 kali sehingga *item* roti, kopi dibuang. sedangkan *item* gula nilai kemunculannya 2 maka *subsets* yang terbentuk adalah {susu}{susu,gula}.

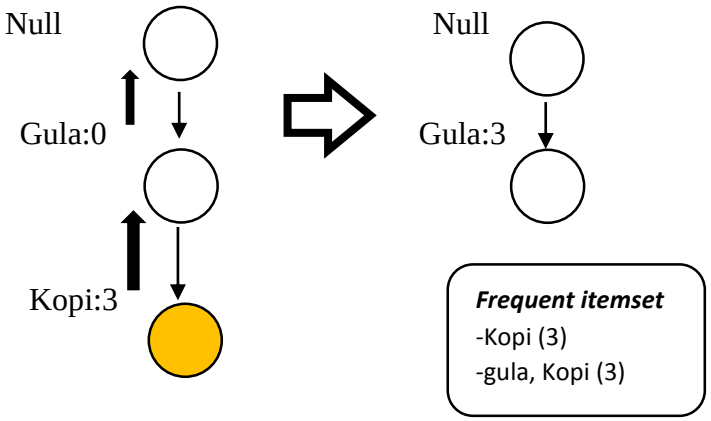
3 Kondisi *FP-Tree* untuk item roti.



Gambar 2.10 Kondisi FP-Tree untuk item Roti

Pada kasus ini, *item* gula mempunyai 2 nilai kemunculan bersamaan dengan item, maka *subsets* yang terbentuk adalah {roti}{roti,gula}.

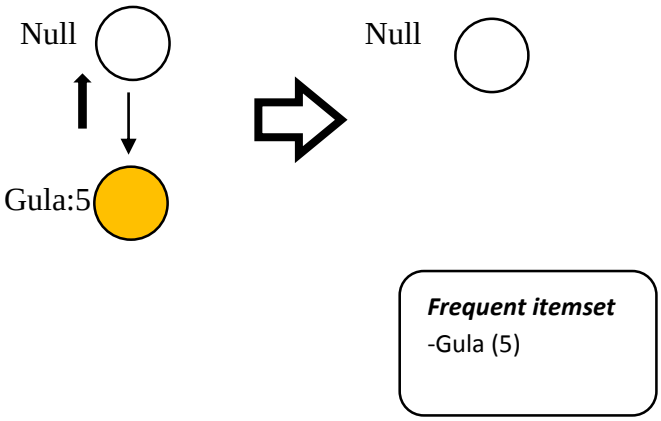
4 Kondisi *FP-Tree* untuk item kopi



Gambar 2.11 Kondisi *FP-Tree* untuk item Kopi

Pada kasus ini, *item* gula mempunyai 3 kali kemunculan bersamaan dengan item kopi, maka *subsets* yang terbentuk adalah {kopi}{gula,kopi}.

5 Kondisi *FP-Tree* untuk item Gula



Gambar 2.12 Kondisi FP-Tree untuk item Gula

Pada kasus ini, lintasan yang berakhiran *item* Gula merupakan lintasan tunggal yang bersiri sendiri dan memiliki nilai frekuensi 5 sehingga *frequent itemsets*-nya hanya {Gula}.

4. *Frequent itemset* yang dihasilkan

Setelah memeriksa kondisi *FP-Tree*, didapat 11 *Frequent itemsets* yang hasilnya di rangkum dalam tabel 2.7, dan menghasilkan rule antar item dengna minimum support count dua yang dirangkum pada tabel 2.6

Tabel 2.6 Hasil rule kombinasi item

Rule	Support count
{gula→ kopi}	3
{gula→ teh}	2
{gula→susu}	2
{gula→roti}	2
{kopi→ teh}	2

Hasil dari 5 transaksi mendapatkan rule kombinasi antar item tertinggi yaitu {gula→kopi} dengan nilai support count 3, yang artinya item gula dan kopi lebih sering dibeli bersamaan sebayak 3 kali dari 5 transaksi.

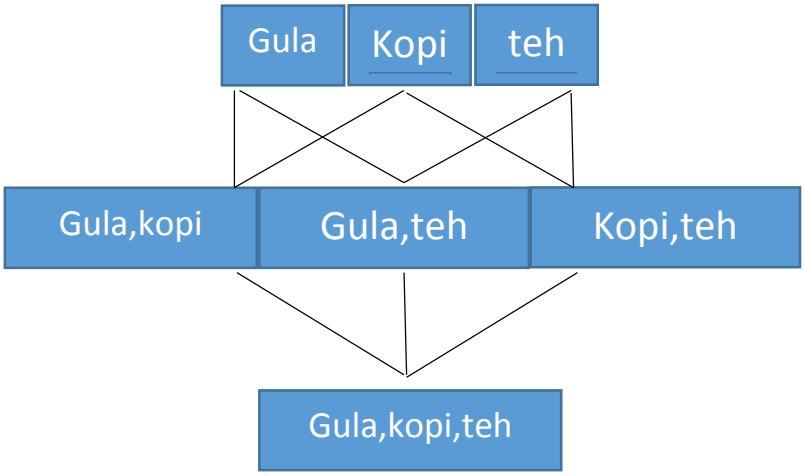
Tabel 2.7 *Frequent itemset*

Suffix	***Frequent itemset***
Teh	{kopi,teh}, {Gula,kopi,teh}, {gula,teh}, {teh}
Susu	{gula,susu}, {susu}
Roti	{gula,roti}, {roti}
Kopi	{gula,kopi}, {kopi}
Gula	{gula}

Dari 11 *Frequent itemsets* tersebut, tidak semua dihitung. Karena dalam menghasilkan *Association Rule*, minimal *Frequent itemsets* yang dihitung

terdapat 2 *item* dimana jika membeli *item* A maka akan membeli *item* B. Sehingga yang layak dihitung *confidence*-nya adalah 6 *subsets*, diantaranya : **{kopi,teh}, {gula, kopi, teh}, {gula,teh}, {gula,susu}, {gula,roti}, {gula,kopi}**

Setelah mendapatkan *frequent itemsets* yang akan dihitung, selanjutnya adalah membuat *rule* dengan menghitung *minumum support* dan *minmum confidence*-nya. Hanya pola yang nilai *support*-nya ≥0,4 dan *confidence*-nya ≥ 1 yang akan diambil. Rumus menghitung *confidence* dapat dilihat pada rumus yang sebelumnya sudah dijelaskan. Karena perhitungannya sangat banyak, maka penulis mengambil contoh dari *frequent itemsets* {gula, kopi,teh} untuk dicari kombinasinya dan dihitung nilai *confidence*-nya seperti terlihat pada gambar 2.13



Gambar 2.13 Proses pencarian kombinasi untuk frequent itemset

Dari proses pencarian kombinasi untuk *frequent itemsets* {gula,kopi,teh}, didapat 12 pola, dan perhitungan *confidence*-nya dengan rumus sebagai berikut:

$$Confidence = (A \rightarrow B) = \frac{Jumlah\ Transaksi\ Mengandung\ A\ dan\ B}{Jumlah\ Transaksi\ mengandung\ A}$$

Rule frequent itemset untuk (Gula. Kopi. Teh)

- Gula -> kopi = 3/5= 0,6
- Kopi -> gula = 3/3= 1
- Gula -> teh = 2/5= 0,4
- Teh -> gula = 2/2= 1
- Kopi -> teh =2/3= 0,67
- Teh -> kopi =2/2= 1

- Teh ^kopi -> gula = 2/2= 1
- Gula -> teh^kopi = 2/5= 0,4
- Kopi -> teh^gula = 2/3= 0,67
- Teh -> gula ^ kopi = 2/2= 1
- Teh -> kopi^gula = 2/2=1
- Kopi^gula -> teh = 2/3=0,67

Rule frequent itemset untuk (gula, susu)

- Gula -> susu =2/5=0,4
- Susu -> gula =2/2= 1

Rule frequent itemset untuk (gula, roti)

- Gula -> roti = 2/5= 0,4
- Roti -> gula = 2/2= 1

Dari perhitungan *confidence* terhadap pola diatas maka *Association Rule* yang memenuhi syarat *confidence* ≥ 1,0 adalah :

1) kopi → gula (jika konsumen membeli **kopi** maka membeli **gula** dengan nilai *confidence* 1,0)

2) teh → gula (jika konsumen membeli **teh** maka membeli **gula** dengan nilai *confidence* 1.0)

3) teh → kopi (jika konsumen membeli **teh** maka membeli **kopi** dengan nilai *confidence* 1.0)

4) teh^kopi → gula (jika konsumen membeli **teh** dan **kopi** maka membeli **gula** dengan nilai *confidence* 1,0)

5) teh^gula → kopi (jika konsumen membeli **teh** dan **gula** maka membeli **kopi** dengan nilai *confidence* 1.0)

6) teh → kopi^gula (jika konsumen membeli **teh** maka membeli **kopi** dan **gula** dengan nilai *confidence* 1.0)

7) teh → kopi^gula (jika konsumen membeli **susu** maka membeli **gula** dengan nilai *confidence* 1.0)

8) teh → kopi^gula (jika konsumen membeli **roti** maka membeli **gula** dengan nilai *confidence* 1.0)

Dari hasil perhitungan confidace rule yang memenuhi syarat c*onfidence* ≥ 1,0 di rangkum pada tabel 7

Tabel 2.8 Hasil *Association rules*

rule	***confidence***
Kopi → gula	1,0
Teh → gula	1.0
Teh → kopi	1,0
Teh → gula, kopi	1.0
Susu → gula	1,0
Roti → gula	1,0

2.9 Penelitian sebelumnya

Adapun papper yang digunakan sebagai refrensi pembelajaran sebagai berikut :

I. Penelitian yang pertama dengan judul“Implementasi Algoritma Frquent Pattern Growth (FP-Growth) untuk menentukan assosiasi antar produk” oleh (Rizka Nurul Arifin, 2014) Dari hasil analisis dan pengujian yang telah dilakukan, penulis menjelaskan bahwa metode data mining yaitu *market basket analysis* dengan Algoritma *FP-Growth* dapat diterapkan pada data transaksi untuk menentukan promosi di Minimarket Nadiamart dengan aturan asosiasi yang dihasilkan adalah :

   a) Hasil dari pengolahan 2020 data transaksi melalui aplikasi *Market Basket Analysis* dengan batasan minimum nilai *support* sebesar 7% dan *confidence* sebesar 30%, terdapat 1 pola asosiasi yang memenuhi syarat. Salah dua pola asosiasi yaitu jika membeli snack maka membeli susu instant dengan nilai *support* = 8.01% dan nilai *confidence* = 33. 89% yang merupakan pola dengan nilai *support* dan *confidence* tertinggi. Hasil ini juga menjelaskan bahwa, semakin banyak jenis kriteria *item* yang diteliti maka semakin kecil nilai *support*-nya.
   b) Pola yang didapat bisa digunakan membantu swalayan untuk membantu dalam menetukan keputusan memberikan paket diskon atau *bundling* terhadap pola pembelian item yang memiliki nilai *confidence* tinggi namun memiliki *support* yang kecil.

II. Penellitian yang kedua dengan judul “Penerapan Data Mining Dengan Algoritma FP-Growth untuk mendukung strategi promosi pendidikan” oleh Ali Ikhwan, Dicy Nofriansyah, sriani, penulis menjelaskan bahwa Kampus STMIK Triguna Dharma dalam

menentukan strategi promosi pendidikan yang tepat dengan menggunkan Algoritma FP-growth dan membangun FP-Tree

dalam menentukan *frequent itemset* bekerja sangat baik, dengan cara mengelolah data asal sekolah, daerah, jurusan asal, jurusan yang diambil dan menghasilkan Informasi yang berkaitan dengan pelaksanaan promosi dapat tersedia dengan cepat.

Contoh jika siswa asal sekolahnya suwasta, Alamatnya Medan dan Asal Jurusanya IPS maka dia memilih Prodi Sistem Informasi dengan tingkat kepercayaan 100% dan didukung 6% dari data

keseluruhan. Hasil dari rule tersebut yang akan dijadikan target dalam mempromosikan pendidikan dalam pemilihan tempat yang akan dijadikan promosi pendidikan, sehingga pihak manajemen dari kampus sendiri dapat melakukan pengambilan keputusan dengan cepat. *(*Ali Ikhwan *, DKKr 2015).*