BAB III
METODE PENELITIAN DAN PERANCANGAN

Pada bab metodologi penelitian ini akan dibahas penggunaan metode yang digunakan dalam pembuatan perangkat lunak untuk menganalisis data sertifikasi guru dengan menggunakan algoritma *FP-Growth*.

Tahapan pembuatannya sebagai berikut :

1. Melakukan studi literatur mengenai *association rule* dan algoritma *FP-Growth*.
2. Menganalisa dan merancang sistem.
3. Membuat perangkat lunak sesuai dengan analisa dan perancangan yang dilakukan.
4. Menguji coba perangkat lunak.
5. Mengevaluasi hasil analisa yang dilakukan oleh sistem.

3.1. Analisa Umum

3.1.1. Deskripsi Umum Sistem

Sistem yang akan dibuat merupakan sistem yang dikembangkan untuk melakukan analisa terhadap data NUPTK untuk mengetahui pola sertifikasi guru dan selanjutnya akan digunakan untuk menentukan kelayakan sertifikasi guru.

Metode yang digunakan dalam sistem yang akan dibuat adalah metode *association rule* untuk melakukan *frequent itemset mining*. Sedangkan untuk mencari *frequent itemset* dan menentukan nilai *support* dalam sistem yang akan dibuat digunakan algoritma *FP-Growth*.

Dalam sistem yang akan dibuat, *itemset* berupa himpunan data sertifikasi guru sehingga *frequent itemset*-nya berupa himpunan data sertifikasi guru yang memiliki frekuensi kemunculan tinggi yaitu diperoleh dari hasil pemindaian basis data dan yang memenuhi nilai *minimum support* yang menjadi masukan dalam sistem. Setelah didapatkan *frequent itemset* dengan menerapkan algoritma *FP-Growth*, selanjutnya dalam sistem akan dilakukan perhitungan nilai *confidence* untuk setiap *frequent itemset*. Pada tahap ini, dilakukan proses pembangkitan *frequent itemset* menjadi *rule* dengan mengimplementasilan metode *association*

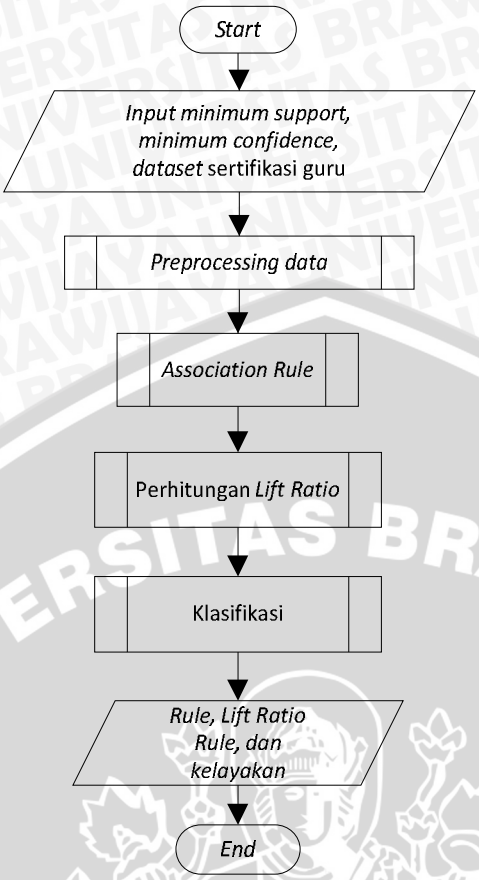
rule pada sistem. Setelah didapatkan *frequent itemset* dengan menerapkan algoritma *FP-Growth*, selanjutnya dalam sistem akan dilakukan perhitungan nilai *confidence* untuk setiap *frequent itemset*. Pada tahap ini, dilakukan proses pembangkitan *frequent itemset* menjadi *rule* dengan mengimplementasilan metode *association rule* pada sistem. Suatu f*requent itemset* akan dibangkitkan menjadi *rule* apabila memiliki nilai *confidence* yang memenuhi nilai *minimum confidence* (lebih besar atau sama dengan *minimum confidence*) yang juga menjadi masukan dalam sistem. *Frequent itemset* yang terbentuk menjadi *rule* yang kemudian digunakan untuk penentuan kelayakan sertifikasi guru dan akan ditampilkan sebagai hasil keluaran beserta nilai *confidence* dan nilai *lift ratio* tiap *rule* tersebut. *Rule* yang dihasilkan selanjutnya akan diproses dan membangun *classifier* yang digunakan untuk menentukan kelas-kelas yang digunakan dalam pengklasifikasian kelulusan sertifikasi guru. *Rule* yang telah dibangun menjadi *classifier* akan disimpan dalam *database* untuk mempermudah melakukan proses analisis hasil.

3.1.2. Data Penelitian

Data yang digunakan dalam tugas akhir ini adalah data PTK tahun 2009-2011. Data ini diperoleh dari Dinas Pendidikan Kabupaten Malang. Parameter yang digunakan dalam tugas akhir ini yaitu Nomor Induk Pegawai (NIP), kepemilikan NUPTK, tingkat pendidikan, mata pelajaran sertifikasi, usia, jenis sertifikasi dan masa kerja guru. Parameter-parameter ini dipilih sesuai dengan persyaratan peserta sertifikasi yang ditetapkan oleh Kementerian Pendidikan Nasional pada tahun 2009.

3.2. Perancangan Proses

Dalam proses penelitian ini akan diimplementasikan algoritma *FP-Growth*. Proses yang dilakukan digambarkan dengan *flowchart* pada Gambar 3.1. Secara umum, *input* yang dibutuhkan sistem adalah minimum *suport*, minimum *confidence* dan data sertifikasi guru yang ingin dilakukan *mining*. Proses awal dalam sistem adalah proses *preprocessing data* dan proses *association rule*. Selanjutnya dilakukan proses perhitungan *lift ratio* sehingga dihasilkan keluaran *rule*, nilai *lift ratio rule* dan kelayakan sertifikasi guru.



Gambar 3. 1. *Flowchart* Gambaran Umum Sistem

3.2.1. *Preprocessing Data*

Preprocessing data merupakan proses dimana data akan diinisialisasi sehingga memudahkan dalam penentuan kandidat *itemset*. Inisialisasi data ini menghasilkan id inisialisasi yaitu id 1, id 2, id 3, id 4, id 5, id 6, id 7, id 8, id 9, id 10, id 11, id 12, id 13 dan id 14 serta dapat dilihat pada tabel 3.1, 3.2, dan 3.3.

Tabel 3.1. Tabel Inisialisasi NIP, NUPTK dan Tingkat Pendidikan

NIP	NUPTK	Tingkat Pendidikan
1 : Tidak Ada NIP 2 : Ada NIP	3 : Tidak Ada NUPTK 4 : Ada NUPTK	5 : Tingkat Pendidikan SMA 6 : Tingkat Pendidikan lebih tinggi dari SMA

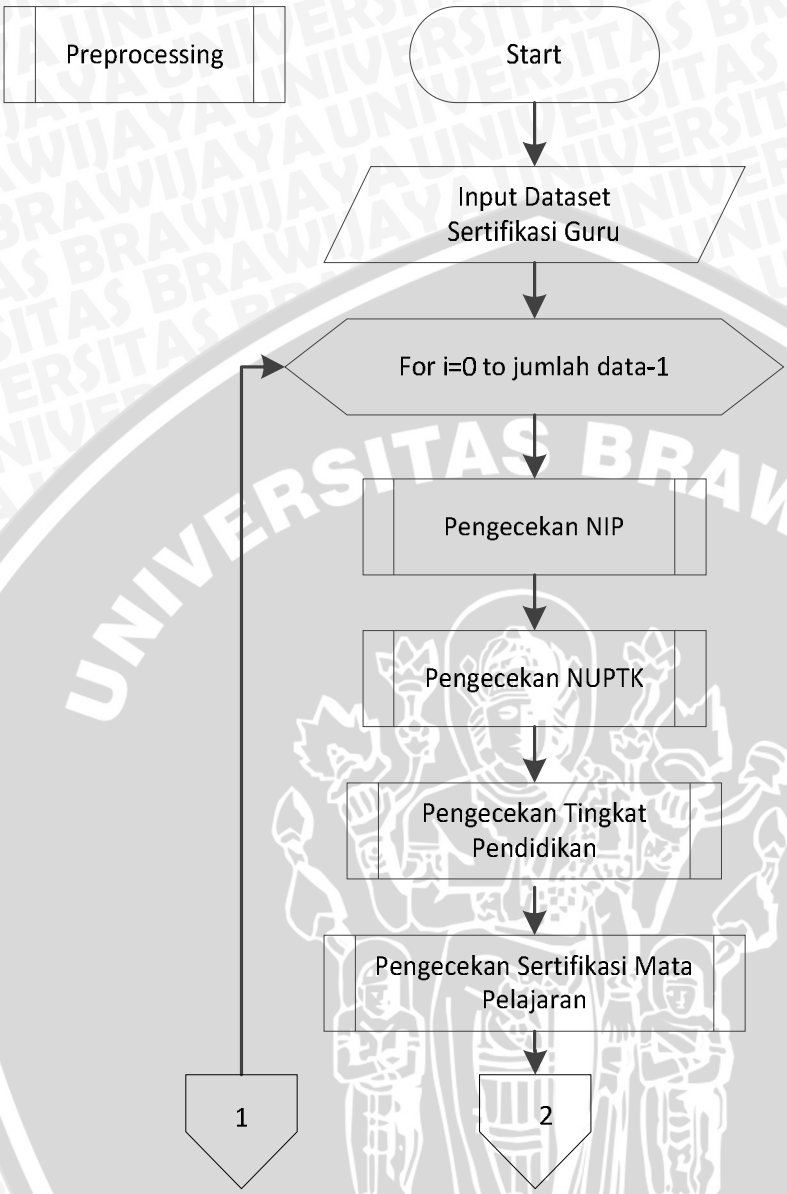
Tabel 3.2. Tabel Inisialisasi Sertifikasi Mata Pelajaran, Usia dan Jenis Sertifikasi

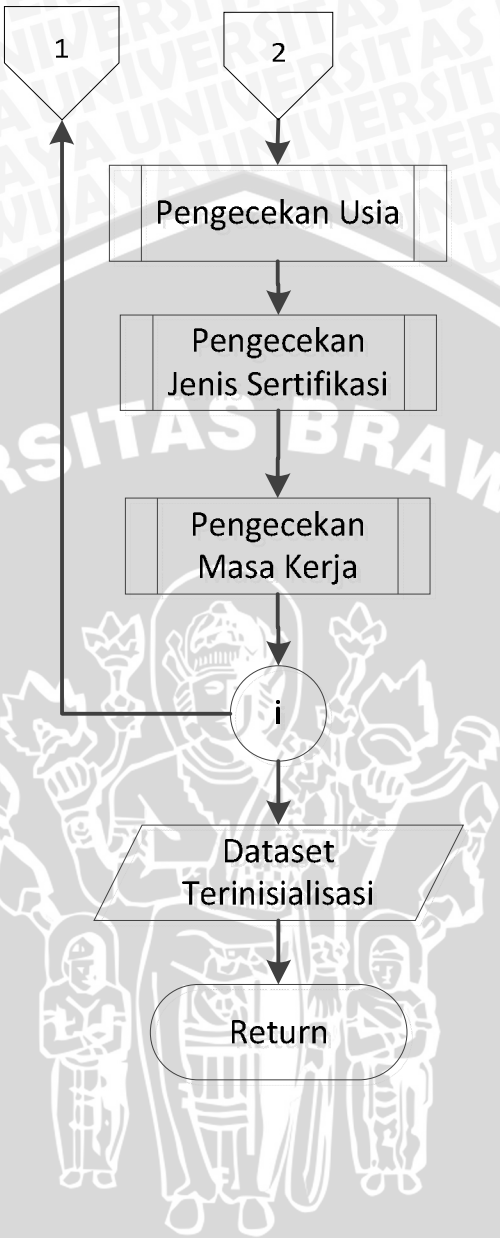
Sertifikasi Mata Pelajaran	Usia	Jenis Sertifikasi
7 : Tidak Ada Sertifikasi Mata Pelajaran 8 : Ada Sertifikasi Mata Pelajaran	9 : Usia <= 35 Tahun 10 : Usia > 35 Tahun	11 : Portofolio 12 : PLPG

Tabel 3.3. Tabel Inisialisasi Masa Kerja dan Status Sertifikasi

Masa Kerja	Status Sertifikasi
13 : Masa Kerja <= 20 Tahun 14 : Masa Kerja > 20 Tahun	15 : Lulus 16 : Tidak Lulus

Tahapan *preprocessing* data terdiri dari pengecekan NIP, pengecekan NUPTK, pengecekan tingkat pendidikan, pengecekan sertifikasi mata pelajaran, pengecekan usia, pengecekan jenis sertifikasi, pengecekan masa kerja dan status sertifikasi. *Flowchart preprocessing* dapat dilihat pada gambar 3.2.


Preprocessing
Start
Input Dataset Sertifikasi Guru
For i=0 to jumlah data-1
Pengecekan NIP
Pengecekan NUPTK
Pengecekan Tingkat Pendidikan
Pengecekan Sertifikasi Mata Pelajaran
1
2

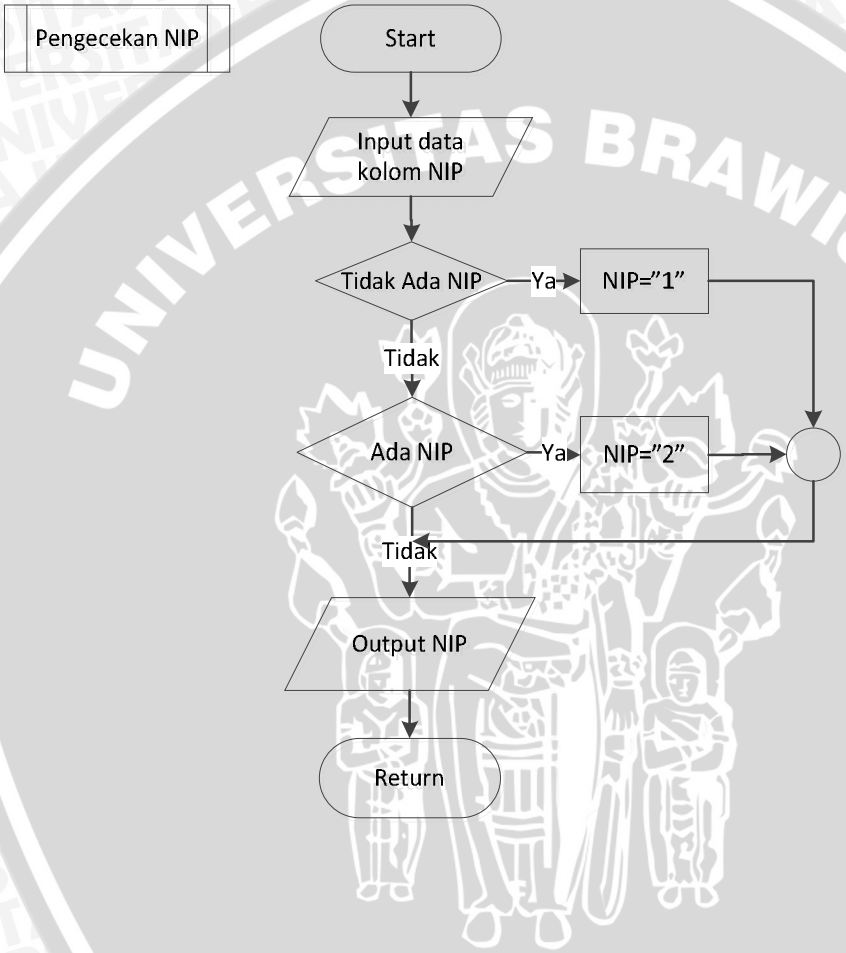


Gambar 3.2. ***Flowchart Preprocessing Data***

3.2.1.1. Pengecekan NIP

Pada tahap ini akan dilakukan inisialisasi terhadap data pada kolom NIP. NIP terdiri dari 4 bagian, yaitu 6 digit pertama menunjukkan tanggal lahir, empat

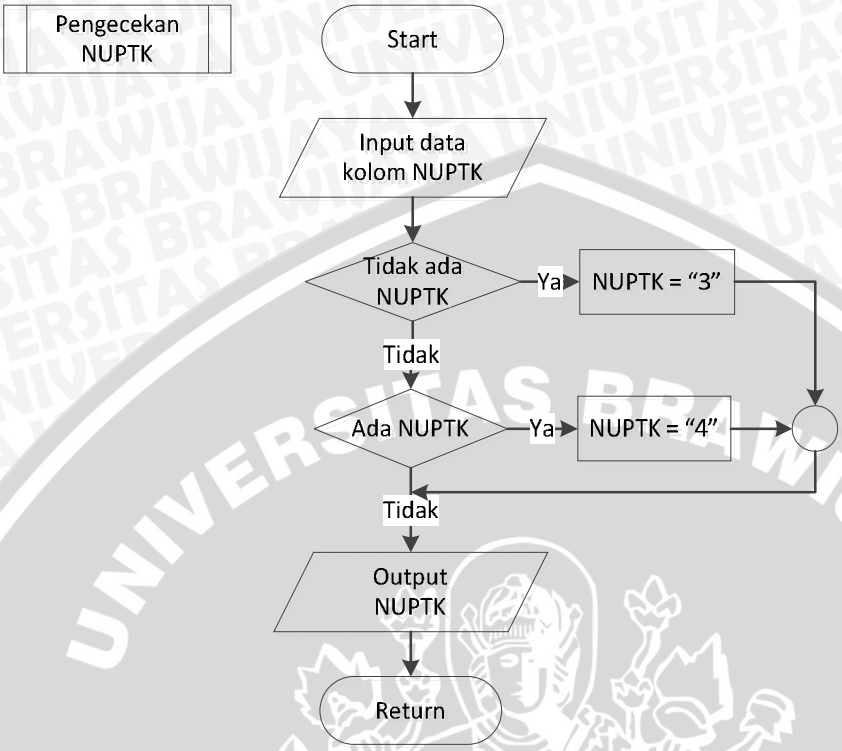
digit selanjutnya menunjukkan tahun pengangkatan, 1 digit menunjukkan jenis kelamin, dan tiga digit menunjukkan nomor urut. Jika data tidak ada nomornya, maka akan dianggap tidak memilki NIP dan akan diinisialisasi menggunakan interval yang ditunjukkan pada tabel 3.1. *Flowchart* pengecekan NIP dapat dilihat pada gambar 3.3.



Gambar 3.3. *Flowchart* Pengecekan NIP

3.2.1.2. Pengecekan NUPTK

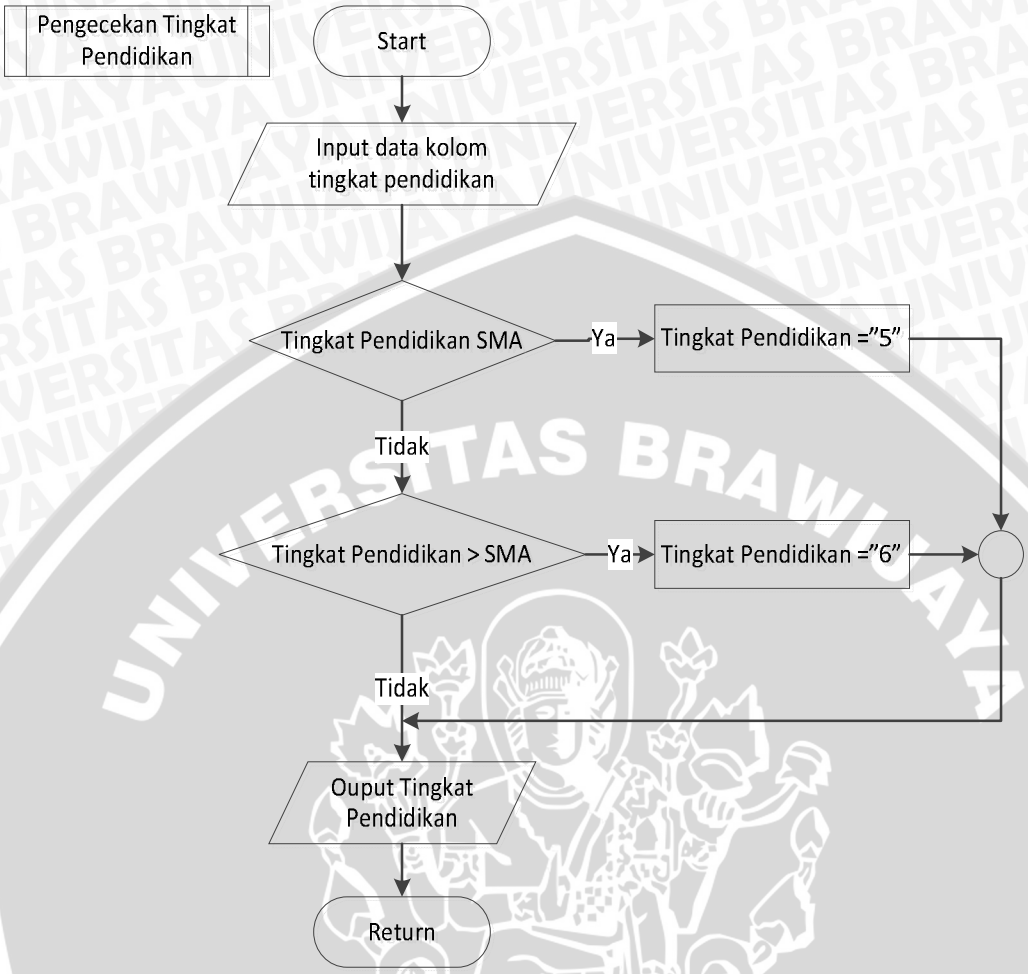
Pada tahap ini akan dilakukan inisialisasi terhadap data pada kolom NUPTK. Input data berupa nomor NUPTK. Jika data tidak ada nomornya, maka akan dianggap tidak memilki NUPTK. NUPTK diinisialisasi menggunakan tabel 3.1. *Flowchart* pengecekan NUPTK dapat dilihat pada gambar 3.4.



Gambar 3.4. *Flowchart* Pengecekan NUPTK

3.2.1.3. Pengecekan Tingkat Pendidikan

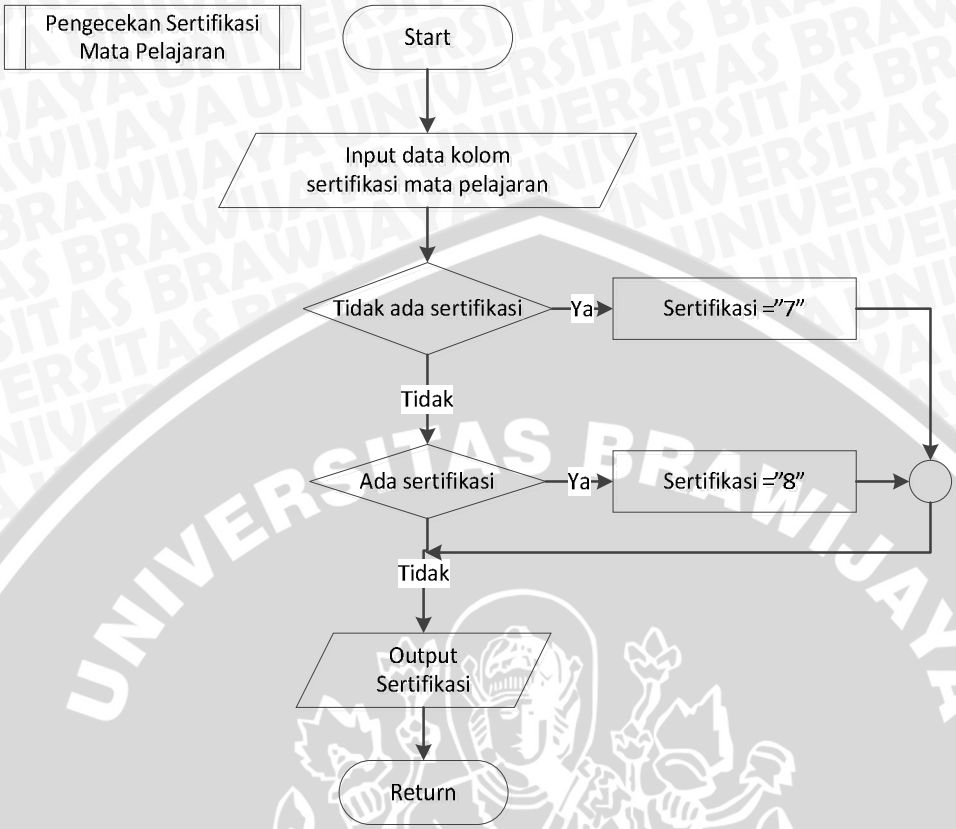
Pada tahap ini akan dilakukan inisialisasi terhadap data pada kolom Tingkat Pendidikan. Tingkat pendidikan merupakan jenjang akademik dari pendidik dan tenaga kependidikan. Tingkat pendidikan hanya digolongkan menjadi 2, yaitu pendidikan SMA dan pendidikan di atas SMA. Tingkat pendidikan diinisialisasi menggunakan tabel 3.1. *Flowchart* pengecekan tingkat pendidikan dapat dilihat pada gambar 3.5.



Gambar 3.5. *Flowchart* Pengecekan Tingkat Pendidikan

3.2.1.4. Pengecekan Sertifikasi Mata Pelajaran

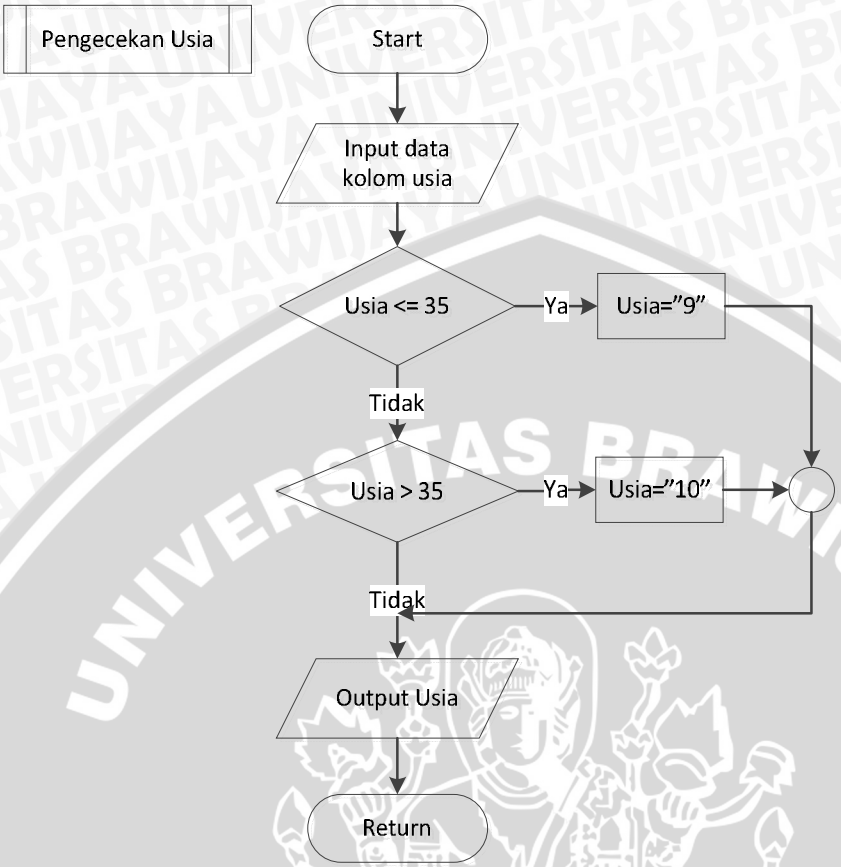
Pada tahap ini akan dilakukan inisialisasi terhadap data pada kolom sertifikasi mata pelajaran. Sertifikasi mata pelajaran merupakan pengecekan ada atau tidaknya sertifikasi mata pelajaran masing-masing Pendidik dan Tenaga Kependidikan. Sertifikasi mata pelajaran diinisialisasi menggunakan tabel 3.2. *Flowchart* pengecekan sertifikasi mata pelajaran dapat dilihat pada gambar 3.6.



Gambar 3.6. *Flowchart* Pengecekan Sertifikasi Mata Pelajaran

3.2.1.5. Pengecekan Usia

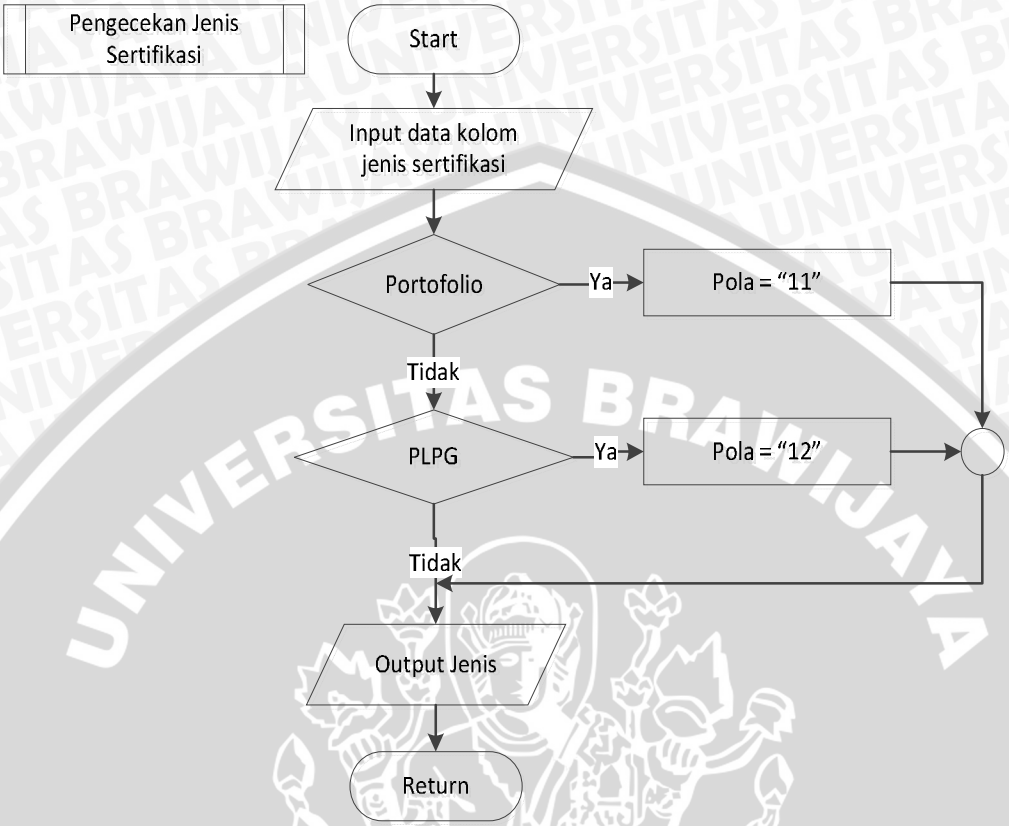
Pada tahap ini akan dilakukan inisialisasi terhadap data pada kolom usia. Usia dibedakan menjadi 2 interval, yaitu usia kurang dari atau sama dengan 35 tahun dan usia lebih dari 35 tahun. Penginisialisasian usia menggunakan interval yang ditunjukkan pada tabel 3.2. *Flowchart* pengecekan usia dapat dilihat pada gambar 3.7.



Gambar 3.7. *Flowchart* Pengecekan Usia

3.2.1.6. Pengecekan Jenis Sertifikasi

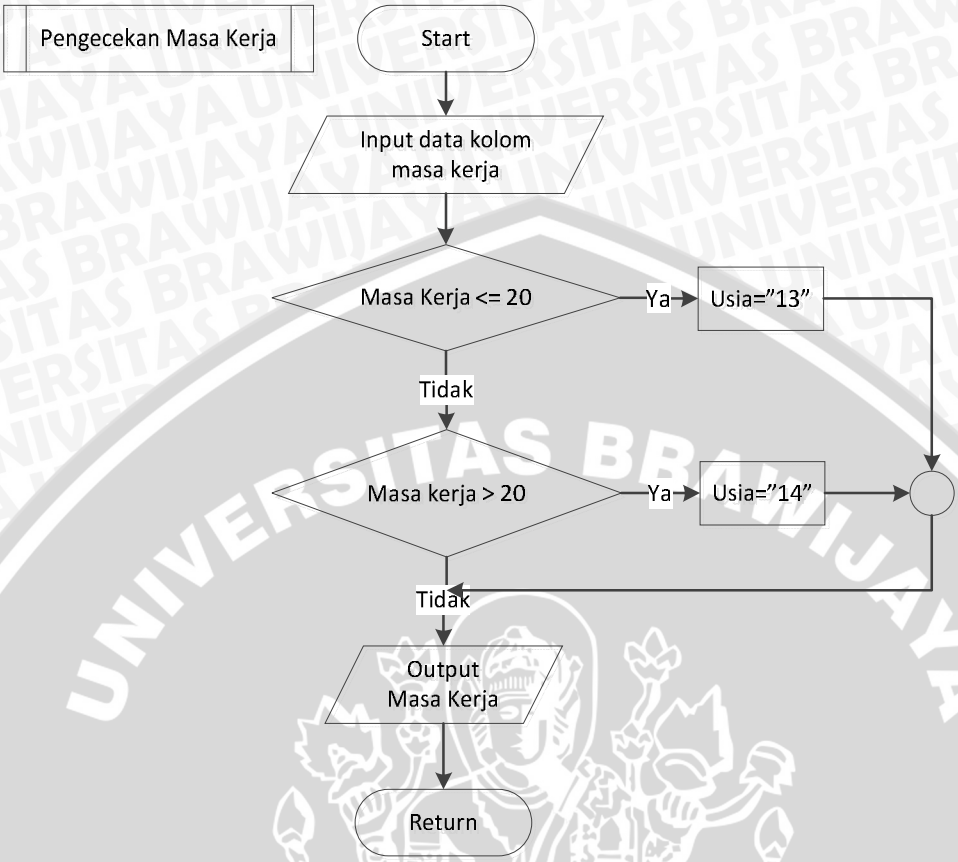
Pada tahap ini akan dilakukan inisialisasi terhadap data pada kolom jenis sertifikasi. Terdapat dua jenis sertifikasi, yaitu portofolio dan PLPG. Portofolio adalah dokumen yang berisi sekumpulan informasi pribadi yang merupakan catatan dan dokumentasi atas pencapaian prestasi seseorang dalam pendidikannya. Sedangkan PLPG singkatan dari Pendidikan dan Pelatihan Profesi Guru, yaitu pelatihan yang diadakan bagi guru yang sudah memenuhi syarat untuk menerima tunjangan profesi (sertifikasi). Penginisialisasian jenis sertifikasi berdasarkan pada tabel 3.2. *Flowchart* pengecekan pola sertifikasi dapat dilihat pada gambar 3.8.



Gambar 3.8. *Flowchart* Pengecekan Jenis Sertifikasi

3.2.1.7. Pengecekan Masa Kerja

Pada tahap ini akan dilakukan inisialisasi terhadap data pada kolom masa kerja. Pada penginisialisasian ini menggunakan 2 interval, yaitu masa kerja kurang dari atau sama dengan 20 tahun dan masa kerja lebih dari 20 tahun. Penginisialisasian masa kerja menggunakan interval yang ditunjukkan pada tabel 3.3. *Flowchart* pengecekan masa kerja dapat dilihat pada gambar 3.9.

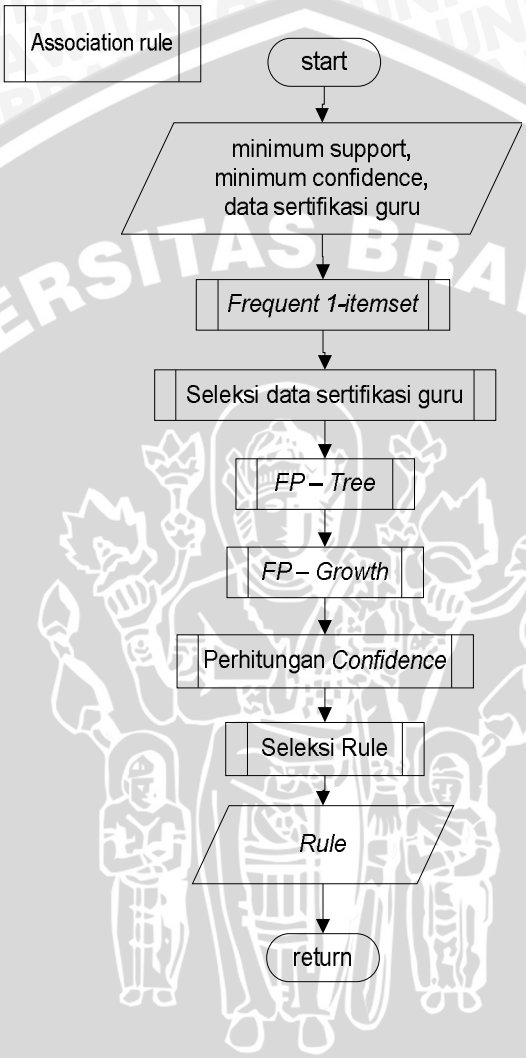


Gambar 3.9. *Flowchart* Pengecekan Masa Kerja

3.2.2. Proses *Association Rule*

Proses *association rule* merupakan proses membangkitkan *rule* dari himpunan *frequent itemset* yang didapat. Himpunan *frequent itemset* yang dibangkitkan menjadi *rule* adalah *frequent itemset* yang memiliki nilai *confidence* lebih dari atau sama dengan nilai minimum *confidence*. Pada proses *association rule* sendiri membutuhkan data input berupa minimum *support*, minimum *comfidence*, dan data sertifikasi guru yang telah diinisialisasi dan akan diproses melalui proses – proses yang terdiri dari proses *frequent 1-itemset*, proses seleksi data sertifikasi guru, proses *FP-Tree*, proses *FP-Growth*, proses perhitungan

confidence dan proses seleksi *rule*. Keluaran dari proses ini adalah *rule* yang akan digunakan sebagai aturan dalam klasifikasi kelulusan sertifikasi guru. Keseluruhan proses *association rule* ini dapat dilihat pada Gambar 3.10.

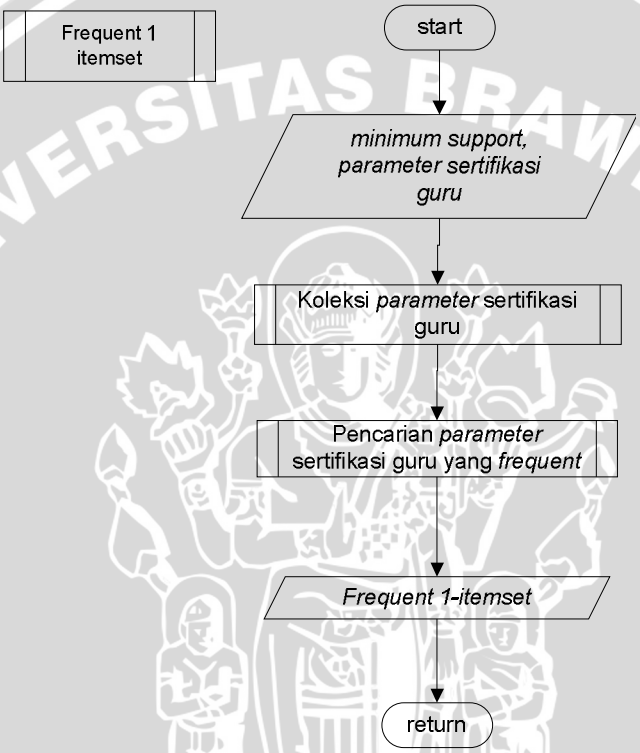


Gambar 3.10. *Flowchart* Proses *Association Rule*

3.2.3. Proses *Frequent 1-itemset*

Proses *frequent 1-itemset* merupakan proses untuk mendapatkan himpunan *frequent 1-itemset*. Himpunan *frequent 1-itemset* didapatkan dari *item* yang memiliki nilai *support* lebih dari atau sama dengan nilai minimum *support. Item* disini merupakan *parameter-parameter* sertifikasi guru yang telah diinisialisasi pada proses *preprocessing data*. Pada proses *frequent 1-itemset* sendiri

membutuhkan data *input* berupa minimum *support* dan *parameter* sertifikasi guru terinisialisasi dan akan diproses melalui proses–proses yang terdiri dari proses pengkoleksian *parameter* sertifikasi guru dan frekuensinya serta proses pencarian *parameter* sertifikasi guru yang *frequent*. Keluaran dari proses ini adalah *frequent 1-itemset*. Keseluruhan proses *frequent 1-itemset* ini dapat dilihat pada Gambar 3.11.

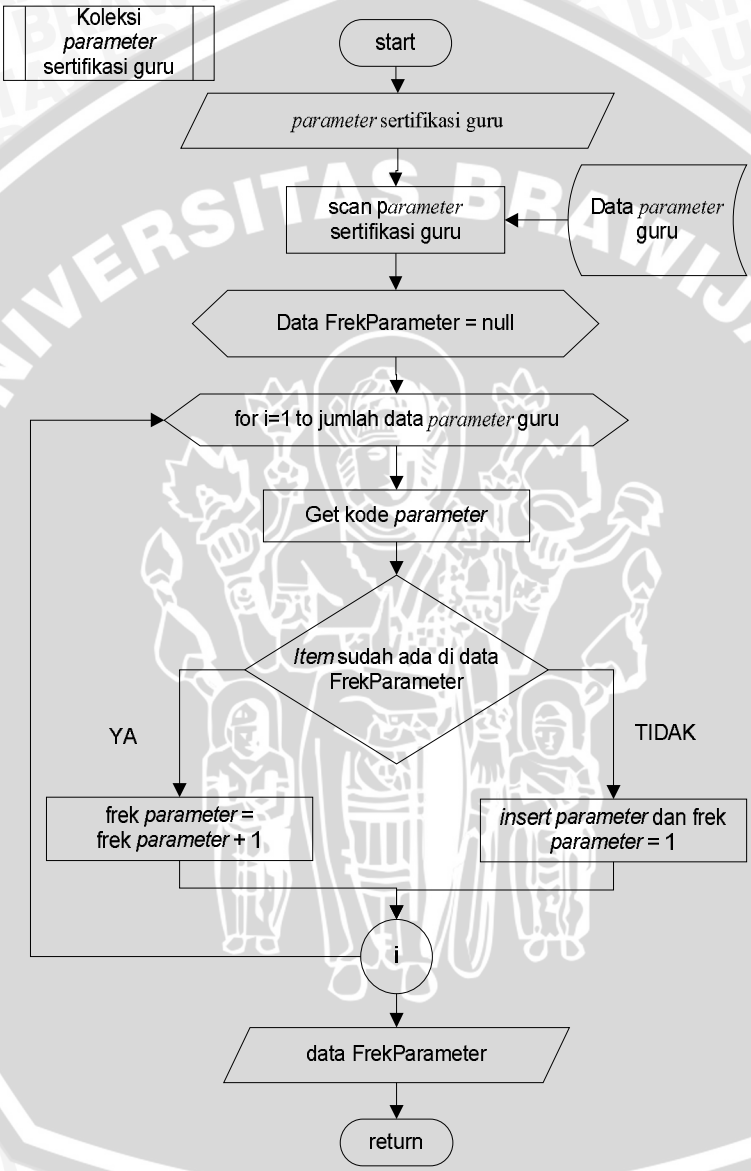


Gambar 3.11. *Flowchart* Proses *Frequent* 1-*itemset*

3.2.4. Proses Koleksi *Parameter* Sertifikasi Guru

Proses koleksi *parameter* sertifikasi guru merupakan proses untuk mendapatkan frekuensi kemunculan tiap *item* pada *parameter* sertifikasi guru. Pada proses ini dibutuhkan data *input* berupa *parameter* sertifikasi guru terinisialisasi. Proses dilakukan dengan melakukan *scan* data sertifikasi guru terinisialisasi. Selanjutnya dilakukan inisialisasi FrekParameter yang akan menyimpan *parameter* sertifikasi guru terinisialisasi dan frekuensinya. Setiap pembacaan data akan dilakukan pengecekan apakah data tersebut sudah ada pada

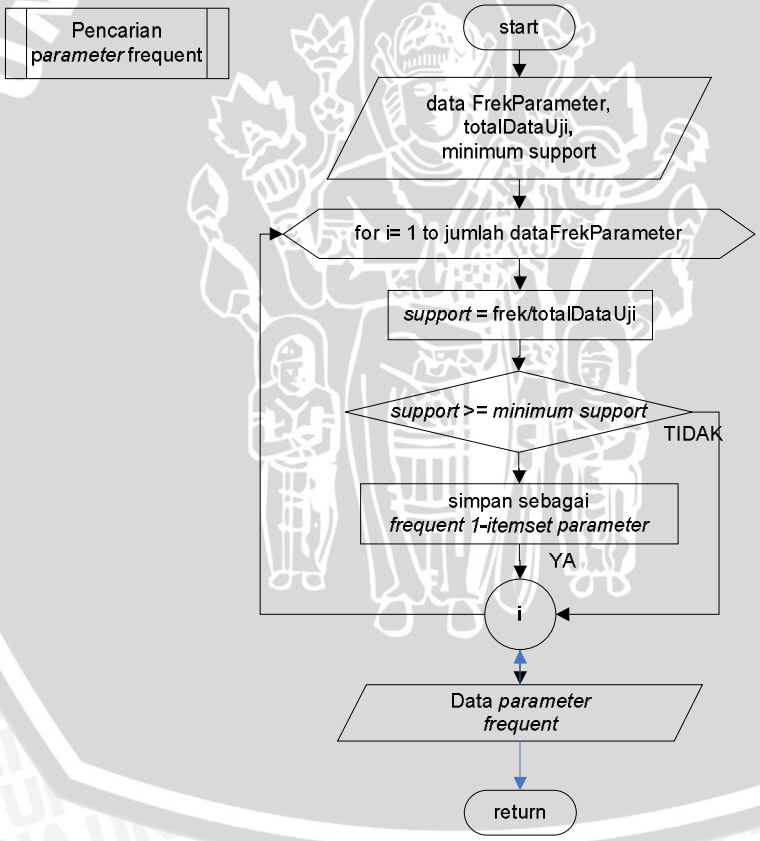
data FrekParameter, jika ada maka frekuensi ditambahkan 1 nilainya , sedangkan jika belum ada maka disimpan kode *parameter* tersebut dan frekuensi diberi nilai 1. Keluaran dari proses ini adalah data FrekParameter. Keseluruhan proses koleksi parameter sertifikasi guru ini dapat dilihat pada Gambar 3.12.



Gambar 3.12. *Flowchart* Proses Koleksi *Parameter* Sertifikasi Guru

3.2.5. Proses Pencarian *Parameter* Sertifikasi Guru yang *Frequent*

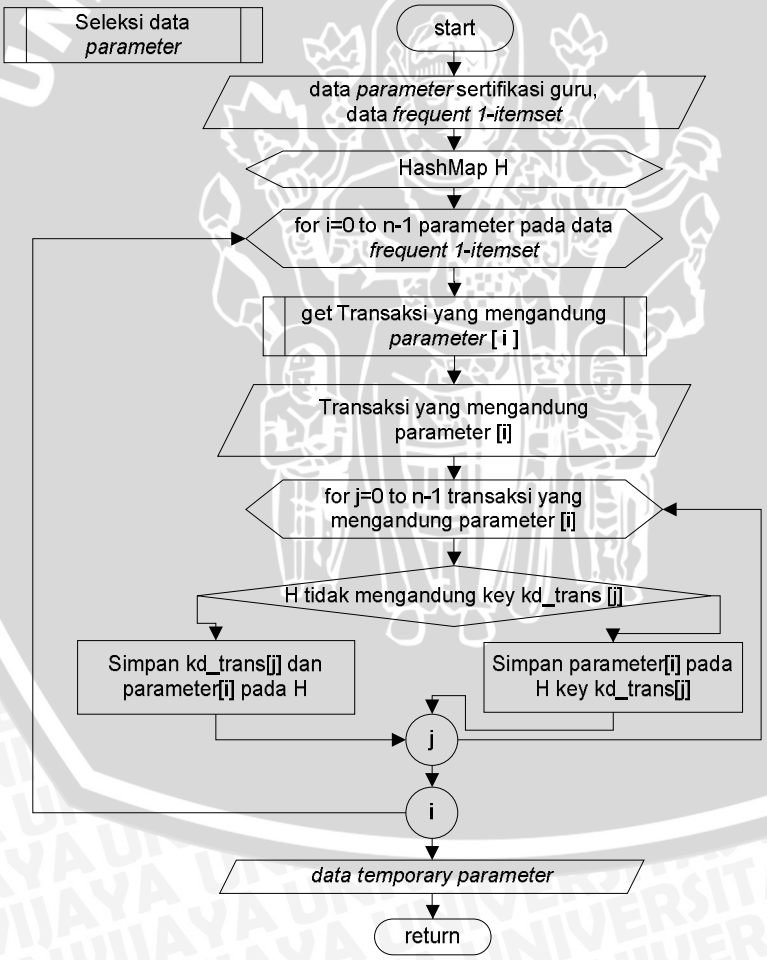
Proses pencarian *parameter* sertifikasi guru yang *frequent* merupakan proses untuk menyeleksi data *parameter* sertifikasi guru yang memenuhi nilai minimum *support*. Pada proses ini dibutuhkan data *input* berupa data FrekParameter, total data sertifikasi guru yang digunakan sebagai data uji dan minimum *support*. Proses dilakukan dengan melakukan pembacaan setiap data FrekParameter. Setiap pembacaan data FrekParameter akan dilakukan perhitungan nilai *support* yang didapat dari nilai frekuensi dibagi dengan total data sertifikasi guru yang digunakan sebagai data uji. Kemudian dilakukan pengecekan apakah nilai *support* lebih dari atau sama dengan nilai minimum *support*, jika lebih dari atau sama dengan maka disimpan sebagai *frequent 1-itemset* parameter sertifikasi guru. Keseluruhan proses pencarian parameter sertifikasi guru yang *frequent* ini dapat dilihat pada Gambar 3.13.



Gambar 3.13. *Flowchart* Proses Pencarian *Parameter* Sertifikasi Guru yang *Frequent*

3.2.6. Proses Seleksi Data *Parameter* Sertifikasi Guru

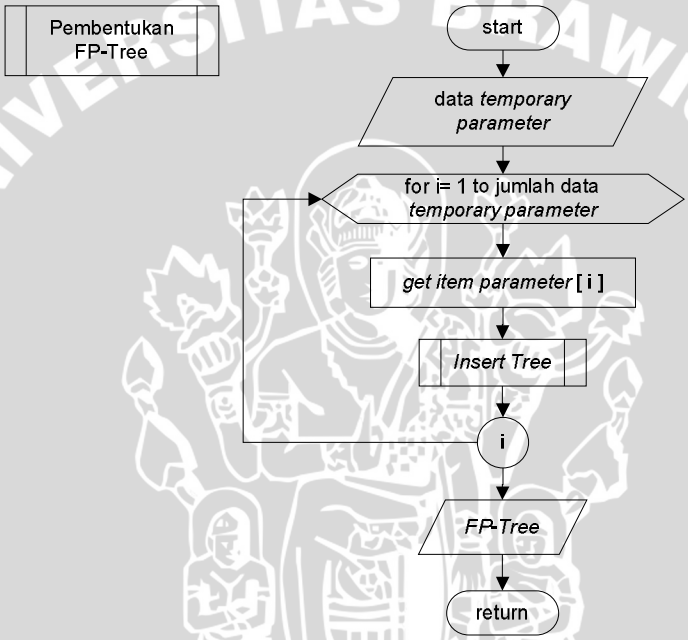
Proses seleksi data *parameter* sertifikasi guru merupakan proses untuk menyeleksi data yang memiliki *parameter* dan kombinasi *parameter* sertifikasi guru yang menjadi *frequent 1-itemset*. Pada proses ini dibutuhkan data input berupa data *parameter* sertifikasi guru dan *frequent 1-itemset parameter* sertifikasi. Proses dilakukan dengan melakukan pembacaan setiap data *frequent 1-itemset parameter* sertifikasi guru, kemudian dilakukan pencarian pada data *parameter* sertifikasi guru yang *memiliki item – item* yamg dicari. *Parameter* sertifikasi guru yang memilki kedua *item* tersebut disimpan sebagai *temporary* data *parameter* sertifikasi guru. Keluaran yang dihasilkan dari proses ini adalah *temporary* data *parameter* sertifikasi guru. Keseluruhan proses seleksi data *parameter* sertifikasi guru ini dapat dilihat pada Gambar 3.14.



Gambar 3.14. *Flowchart* Proses Seleksi Data Sertifikasi Guru

3.2.7. Proses Pembentukan FP-*Tree*

Proses pembentukan *FP-Tree* membutuhkan data *input* berupa data *temporary* dari data *parameter* sertifikasi guru. Proses dilakukan dengan cara *item* yang berupa kombinasi antar *parameter* serta data *parameter* sertifikasi guru pada data *temporary* dari data *parameter* sertifikasi guru yang dibaca dimasukkan ke dalam *FP-Tree* melalui proses *insert tree*. Keluaran yang dihasilkan dari proses ini adalah *FP-Tree*. Keseluruhan proses pembentukan *FP-Tree* ini digambarkan pada Gambar 3.15.

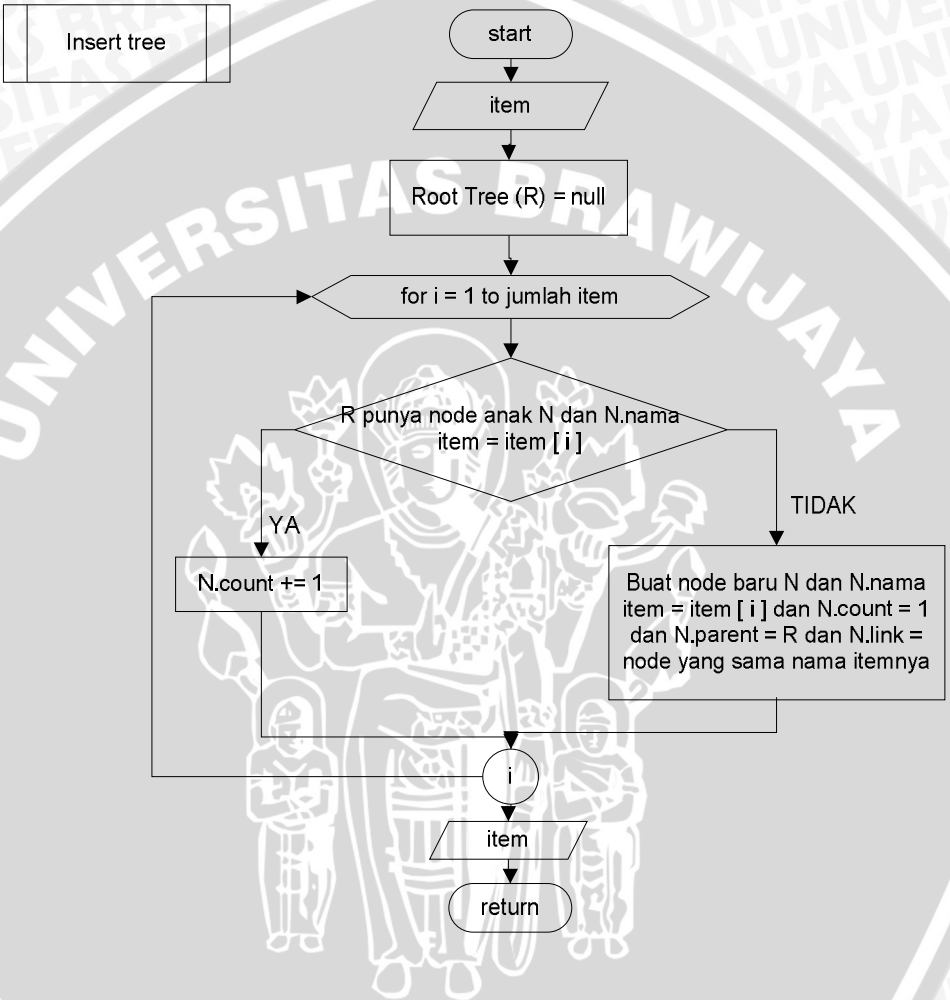


Gambar 3.15. *Flowchart* Proses Pembentukan FP-*Tree*

3.2.8. Proses *Insert Tree*

Proses *insert tree* merupakan proses yang digunakan untuk memasukkan *item* berupa *parameter* sertifikasi ke dalam *FP-Tree*. Pada proses ini dibutuhkan data input berupa *item* yang akan dimasukkan. Proses dilakukan dengan melakukan inisialisasi awal *root tree* adalah *null*. Setiap pembacaan data *item* dilakukan pengecekan apakah *root* mempunyai *node* anak dengan nama *item* yang sama dengan *item* yang sedang dibaca. Jika ada maka *count* pada *node* tersebut ditambahkan 1 nilainya, sedangkan jika tidak maka dibuat *node* baru dengan nama

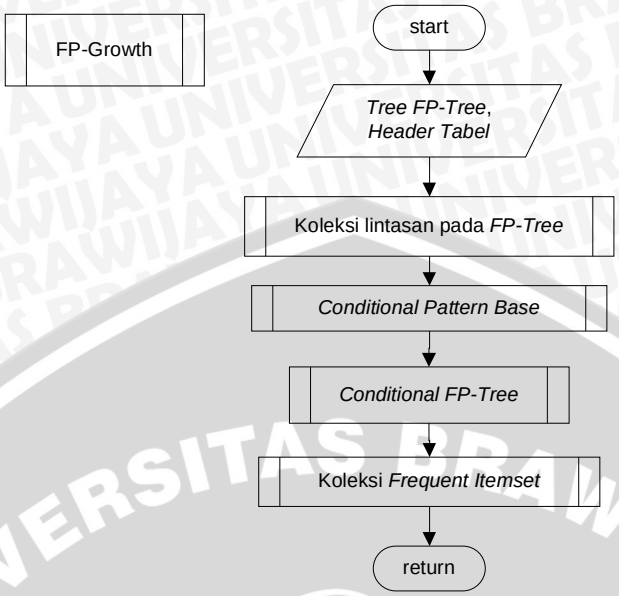
item adalah *item* yang dibaca dan *parent node* tersebut adalah *root* saat ini, sedangkan *link node* adalah *node* lain yang sama nama *item*nya. Keseluruhan proses *insert tree* ini dapat dilihat pada Gambar 3.16.



Gambar 3.16. *Flowchart* Proses *Insert Tree*

3.2.9. Proses FP-*Growth*

Proses ini berupa *FP-Tree* dan *Header Tabel*. Keseluruhan proses *FP-Growth* ini dapat dilihat pada Gambar 3.17.

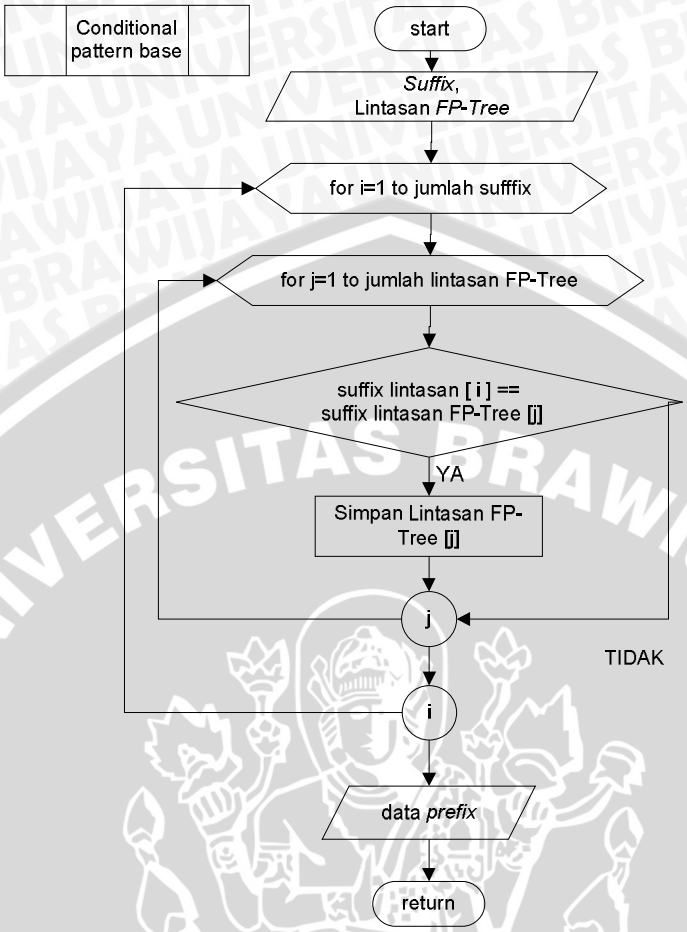


Gambar 3.17. *Flowchart* Proses Algoritma FP-*Growth*

Proses *FP-Growth* merupakan proses yang digunakan untuk *mining frequent itemset*. Pada proses ini akan dilakukan proses – proses yang terdiri dari proses koleksi lintasan pada *FP-Tree,* proses *conditional pattern base,* proses *conditional FP-Tree,* dan proses koleksi *frequent itemset*.

3.2.10. Proses *Conditional Pattern Base*

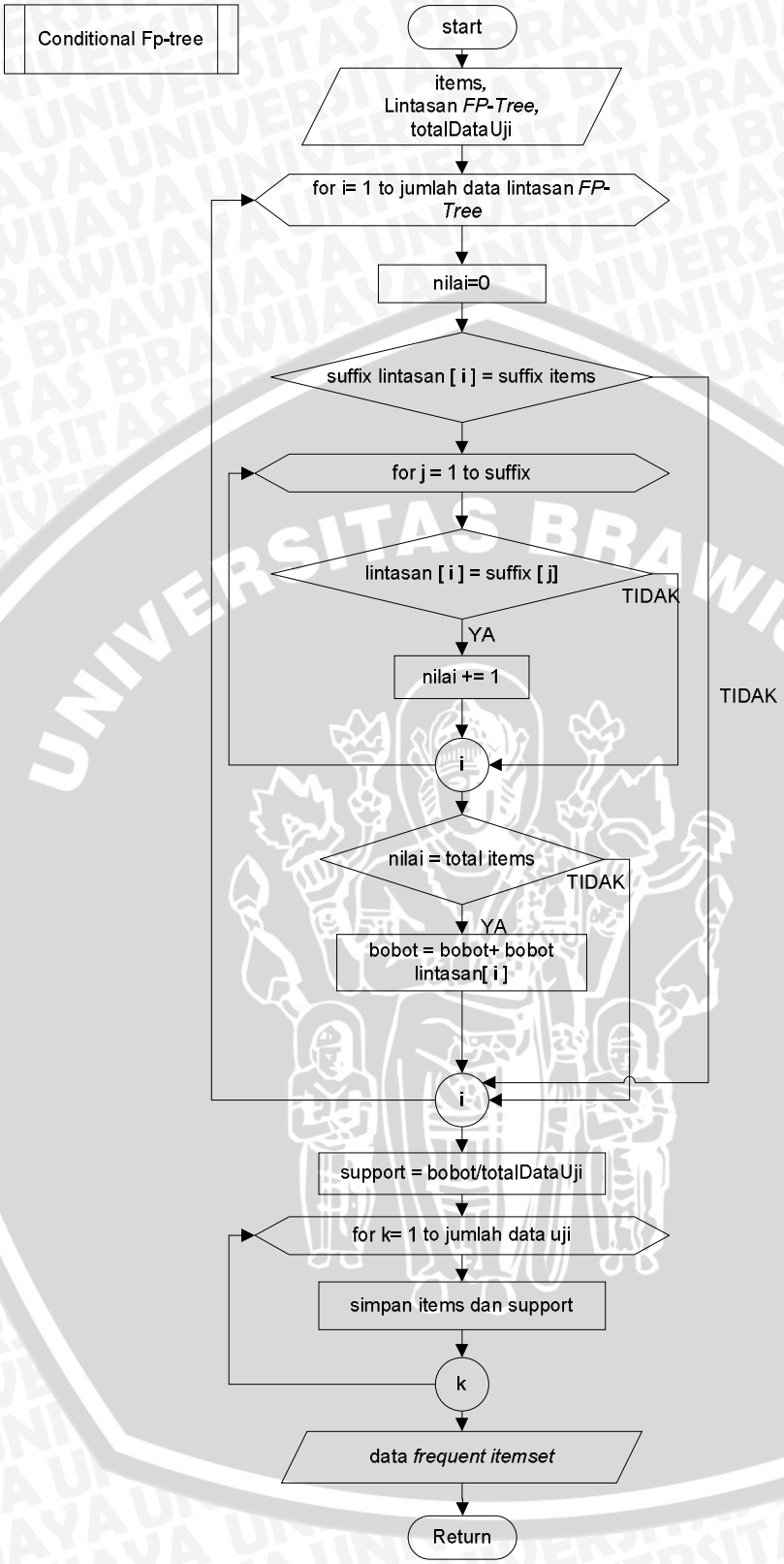
Proses *conditional pattern base* merupakan proses yang digunakan untuk koleksi lintasan tiap *suffix* dimana *item* yang menjadi *suffix* adalah salah satu *parameter* data sertifikasi guru. Pada proses ini dibutuhkan data input berupa *suffix* dan lintasan *FP-Tree*. Proses dilakukan dengan melakukan pembacaan pada setiap data lintasan *FP-Tree* yang selanjutnya dilakukan pengecekan apakah *suffix* pada lintasan yang dibaca sama dengan *suffix* [i]. Jika sama maka *item* dan bobot pada lintasan *FP-Tree* tersebut akan disimpan sebagai data *prefix*. Keseluruhan proses *conditional pattern base* ini digambarkan pada Gambar 3.18.



Gambar 3.18. ***Flowchart Proses Conditional Pattern Base***

3.2.11. Proses *Conditional FP-Tree*

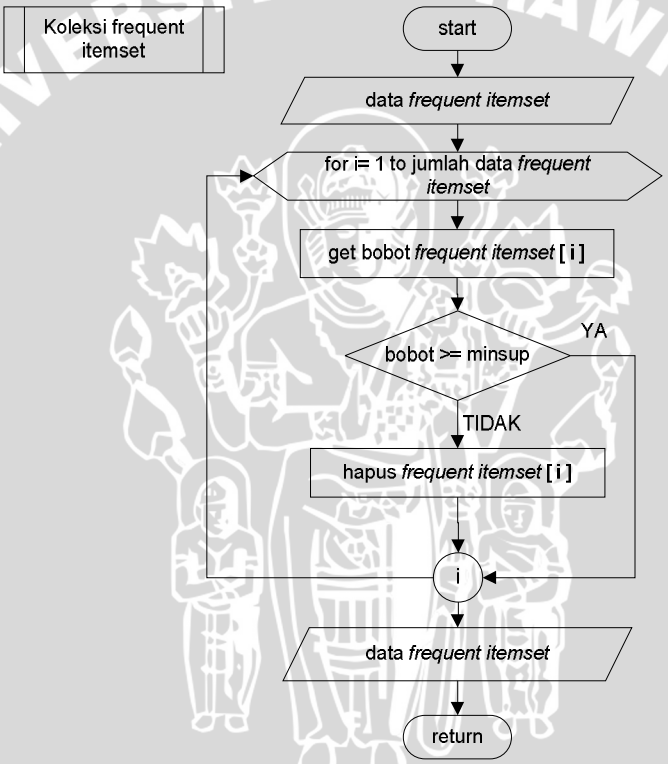
Proses *conditional FP-Tree* merupakan proses untuk mengetahui *support* tiap *item* dengan lintasan yang sama. Pada proses ini dibutuhkan data input berupa data *parameter* sertifikasi guru atau *item*, lintasan *FP-Tree*, dan total data *parameter* sertifikasi guru yang digunakan sebagai data uji. Keluaran dari proses ini adalah data *frequent itemset*. Keseluruhan proses *conditional FP-Tree* dijelaskan pada Gambar 3.19.



Gambar 3.19. *Flowchart* Proses *Conditional FP-tree*

3.2.12. Proses Koleksi *Frequent Itemset*

Proses koleksi *frequent itemset* merupakan proses yang digunakan untuk mengkoleksi *freuqent itemset* yang memenuhi nilai minimum *support*. Pada proses ini dibutuhkan data input berupa data *frequent itemset*. Proses dilakukan dengan melakukan pembacaan pada data *frequent itemset*. Setiap pembacaan data *frequent itemset* dilakukan pengecekan apakah bobot *frequent itemset* lebih dari atau sama dengan minimum *support*. Jika tidak maka dihapus sebagai *frequent itemset*. Keseluruhan proses ini dapat dilihat pada Gambar 3.20.

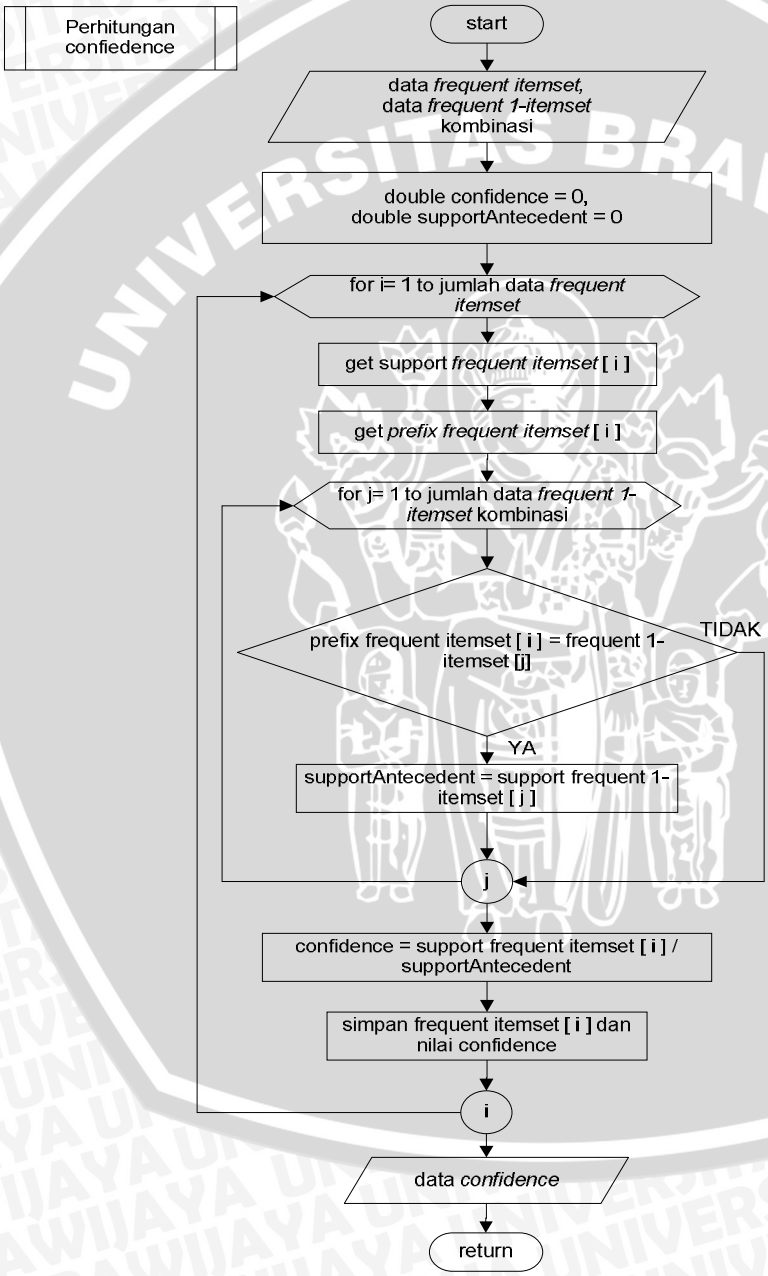


Gambar 3.20. *Flowchart* Proses Koleksi F*requent Itemset*

3.2.13. Proses Perhitungan *Confidence*

Proses perhitungan *confidence* merupakan proses yang digunakan untuk menghitung *confidence frequent itemset*. Pada proses ini dibutuhkan data input data *frequent itemset* dan data *frequent 1-itemset* kombinasi. Proses dilakukan

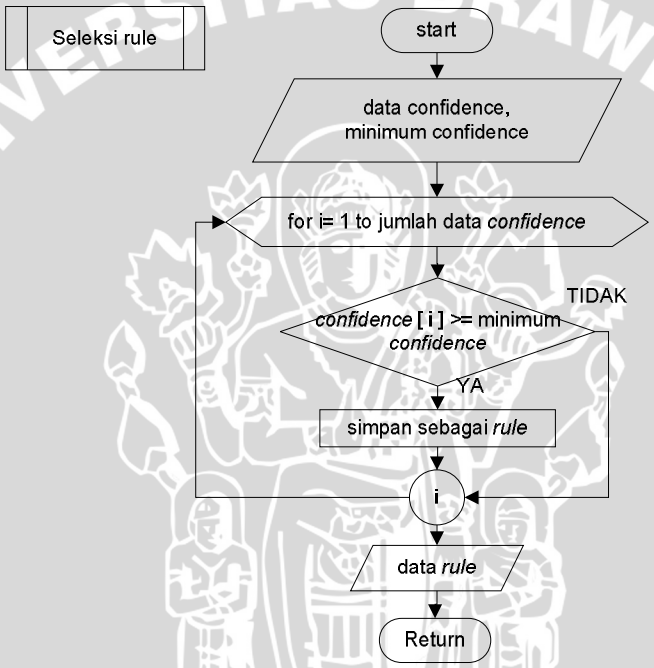
dengan mendapatkan terlebih dahulu nilai *support antecedent. Support antecedent* merupakan *support frequent 1-itemset* kombinasi yang sama dengan *prefix frequent* itemset yang dibaca. Kemudian nilai *confidence* didapatkan dari nilai *support frequent itemset* tersebut dibagi dengan nilai *support antecedent.* Keluaran yang dihasilkan dari proses ini adalah data *confidence.* Keseluruhan proses perhitungan *confidence* ini dapat dilihat pada gambar 3.21.



Gambar 3.21. **Flowchart *Proses Perhitungan* Confidence**

3.2.14. Proses Seleksi *Rule*

Proses seleksi *rule* merupakan proses yang digunakan untuk menyeleksi *rule* yang memenuhi minimum *confidence*. Pada proses ini dibutuhkan data *input* berupa data *confidence* dan data minimum *confidence*. Proses dilakukan dengan melakukan pembacaan setiap data *confidence* yang selanjutnya dilakukan pengecekan apakah data *confidence* yang dibaca lebih dari atau sama dengan nilai minimum *confidence*. Jika ya maka disimpan sebagai *rule*. Keseluruhan proses seleksi *rule* ini dapat dilihat pada Gambar 3.22.

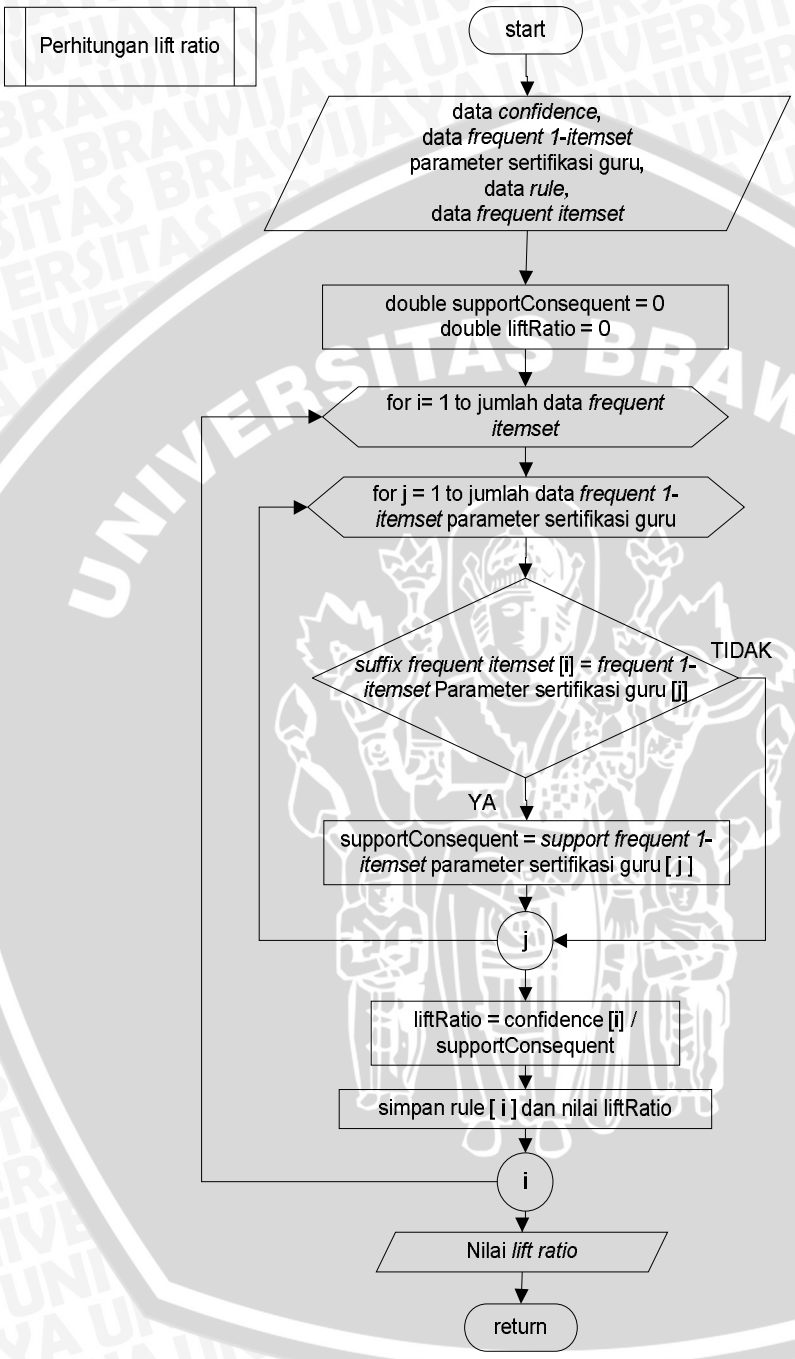


Gambar 3.22. *Flowchart* Proses Seleksi *Rule*

3.2.15. Proses Perhitungan *Lift Ratio*

Proses perhitungan *lift ratio* merupakan proses yang digunakan untuk menghitung nilai *lift ratio* tiap *rule*. Pada proses ini dibutuhkan data *input* berupa data *confidence*, data *frequent itemset*, data *rule*, dan data *frequent 1-itemset* jenis mobil. Proses dilakukan dengan mendapatkan nilai *support consequent* terlebih dahulu. *Support consequent* didapatkan dari *frequent 1-itemset parameter* sertifikasi guru yang sama dengan *suffix frequent itemset* yang sedang dibaca. Kemudian dilakukan perhitungan nilai *lift ratio* yang didapatkan dari nilai

confidence frequent itemset dibagi dengan nilai *support consequent*. Keseluruhan proses perhitungan *lift ratio* ini dapat dilihat pada Gambar 3.23.



Gambar 3.23. *Flowchart* Proses Perhitungan *Lift Ratio*

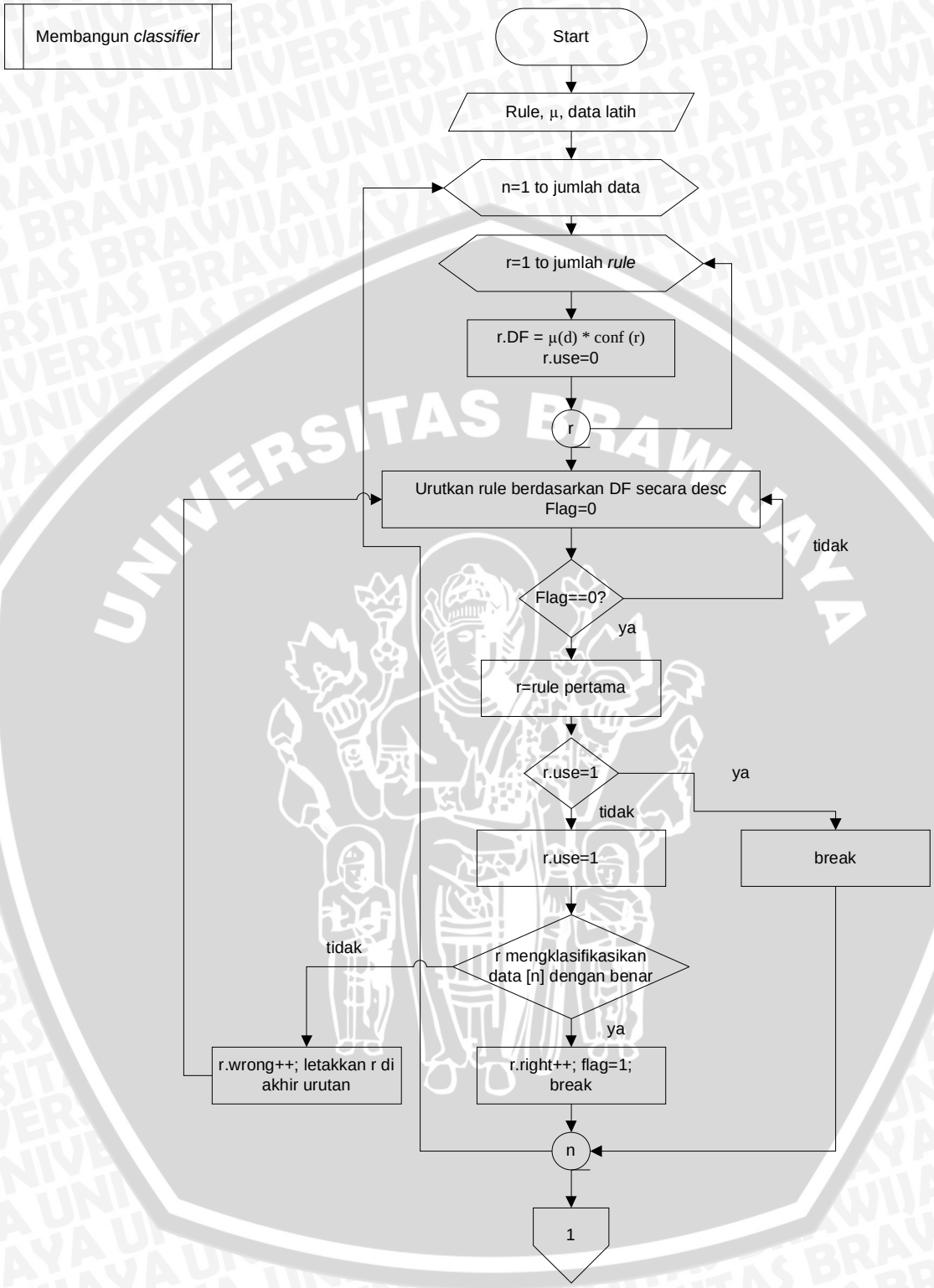
3.2.16. Proses Penentuan Kelayakan Sertifikasi Guru

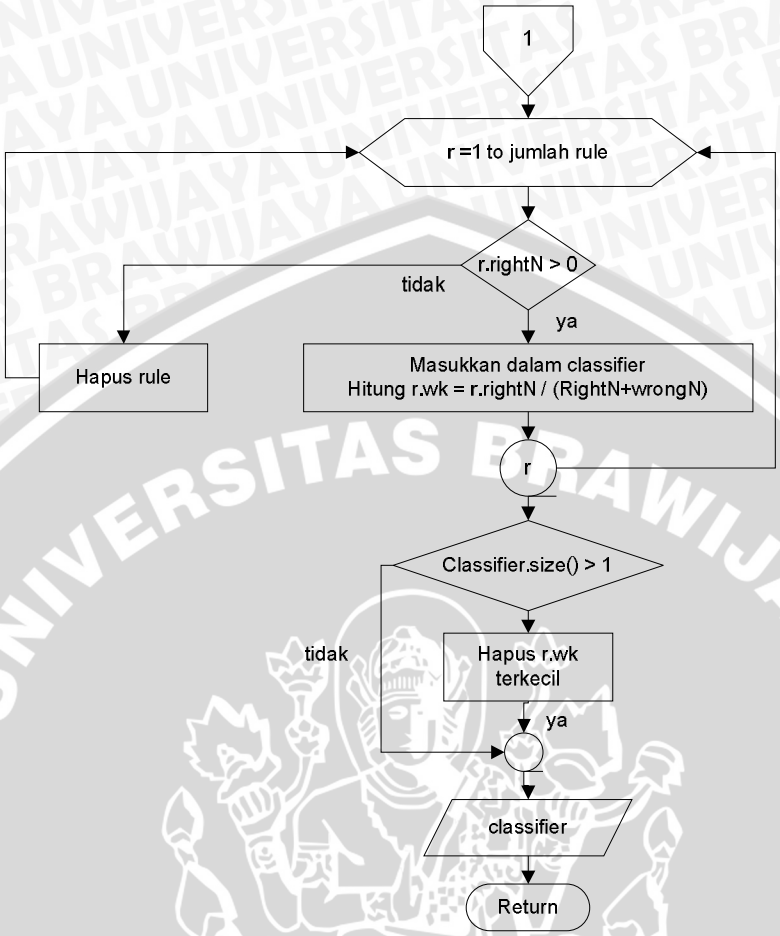
Proses penentuan kelulusan sertifikasi guru merupakan proses yang digunakan untuk mengklasifikasi *itemset* yang mengandung *rule* yang memenuhi minimum *confidence*. Proses yang akan dilakukan adalah membangun *classifier* dan proses klasifikasi.

3.2.16. 1. Proses Membangun *Classifier*

Rule yang didapat pada proses sebelumnya dilatih dengan menggunakan data latih untuk membangun *classifier*. Pelatihan dilakukan dengan menghitung nilai DF setiap *rule*. DF merupakan perkalian derajat keanggotaan *items* pada data ke-n yang bersesuaian dengan *rule* ke-r seperti pada rumus 2.10. Nilai DF kemudian diurutkan dari nilai terbesar ke terkecil. Jika *rule* mengklasifikasikan data latih dengan benar maka nilai *rightN*, *rule* tersebut akan bertambah. Sebaliknya, jika tidak maka nilai *wrongN* yang bertambah. Pada skripsi ini, *rule* dengan nilai *rightN* lebih dari 0 akan dimasukkan dalam *classifier*. Kemudian *rule* dengan nilai *wk* terkecil akan dihapus dari *classifier* selama *rule* yang ada di dalam *classifier* lebih dari 1. Proses membangun *classifier* dapat dilihat pada Gambar 3.24.

Membangun *classifier*





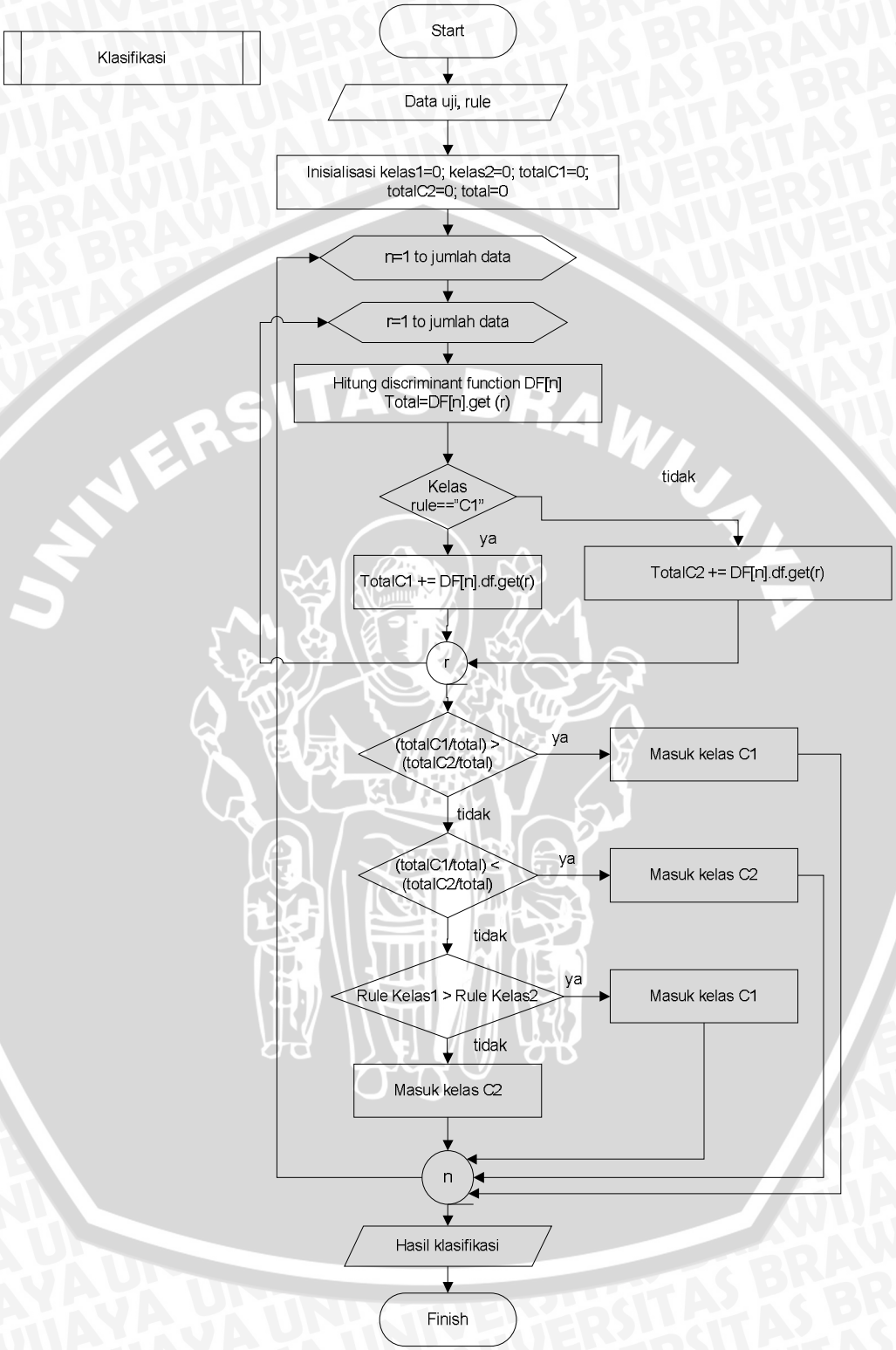
Gambar 3.24. *Flowchart* Proses Membangun *Classifier*

3.2.16.2. Proses Klasifikasi

Setelah *classifier* terbentuk, dilakukan proses klasifikasi yang merupakan proses untuk mengklasifikasikan data uji menggunakan *rule* yang ada di dalam *classifier*. Dalam proses klasifikasi ini, dilakukan proses menghitung *discriminant function* untuk setiap *rule*. *Discriminant function* merupakan hasil perkalian derajat keanggotaan data uji dengan semua *rule* yang dihasilkan.

Hasil perhitungan *discriminant function* digunakan dalam proses klasifikasi data uji. Kelas diklasifikasikan menjadi dua, yaitu kelas C1 atau kelas Lulus dan kelas C2 atau kelas Tidak Lulus. Semua *discriminant function* pada aturan kelas C1 kemudian dibagi dengan total *discriminant function* untuk semua *rule*. Jika

nilai *discriminant function* kelas C1 lebih besar daripada kelas C2, maka data uji akan diklasifikasikan ke kelas C1, begitu juga sebaliknya. Jika *discriminant function* kelas C1 dan kelas C2 sama, maka akan dilihat jumlah aturan kelas mana yang terbesar. Jika jumlah aturan kelas C1 lebih besar maka dimasukkan ke dalam kelas C1. Jika sebaliknya maka akan diklasifikasikan ke dalam kelas C2. Proses perhitungan *discriminant function* dan proses klasifikasi dapat dilihat pada gambar 3.26.



Gambar 3.26. **Flowchart *Proses Perhitungan* Discriminant Function *dan Klasifikasi***

3.3. Perancangan Database

Pada sistem ini, digunakan database sebagai tempat penyimpanan data. Tabel yang digunakan berisi data dari tiap guru, terdiri dari ID, NIP, NUPTK, jenjang_pend, sertifikasi_matapel, tingkat_sekolah, usia, pola_sertifikasi dan masa_kerja yang ditunjukkan pada tabel 3.4.

Tabel 3.4. Atribut tabel Data PTK

Field	Type	Deskripsi	Keterangan
ID	Varchar(4)	Kode data	PK
NIP	Char(16)	Kode NIP	-
NUPTK	Char(16)	NUPTK	-
Tingkat_Pend	Varchar(3)	jenjang pendidikan guru	-
Matapel_Sertifikasi	Varchar(100)	sertifikasi mata pelajaran	-
Usia	Char(2)	usia	-
Jenis_Sertifikasi	Varchar(10)	jenis sertifikasi	-
Masa_Kerja	Varchar(2)	masa kerja guru	-

Database yang digunakan pada skripsi ini terdiri atas 1 tabel yaitu tabel data_latih. Tabel ini masing-masing terdiri atas *field* yang ditunjukkan pada tabel 3.4.

3.4. Perhitungan Manual

Pada perhitungan manual ini digunakan data yang terdiri dari 10 *record* dan 8 atribut. Atribut yang digunakan ID, NIP, NUPTK, tingkat pendidikan, sertifikasi mata pelajaran, tingkat sekolah, usia, jenis sertifikasi dan masa kerja. Nilai minimum *support* yang digunakan pada proses perhitungan manual adalah 50% dan nilai minimum *confidence* yang digunakan adalah 70%.

Dalam melakukan perhitungan manual ini, sebelum dilakukan penerapan *FP-Growth*, sampel data diproses terlebih dahulu sampai bentuk data sesuai dengan yang dibutuhkan. Gambaran data yang digunakan dapat dilihat pada tabel 3.5.

Tabel 3.5. Data Sertifikasi Guru

No	NIP	NUPTK	Tingkat Pendidi-kan	Sertifikasi Mata Pelajaran	Usia	Pola Sertifikasi	Masa Kerja	Kete-rangan
1	198206242 010012019	29567606 61300072	SMA	Tidak Ada	30	Portofolio	7	Tidak Lulus
2	198408072 011012009	41397626 63300013	S1	Tidak Ada	28	Portofolio	6	Lulus
3	197906112 011011002	19437576 59200012	D2	Tidak Ada	33	Portofolio	7	Tidak Lulus
4	197005062 007012009	88387486 51300012	S1	Ada	42	Portofolio	9	Lulus
5	198206122 010012019	09537606 61300012	S1	Tidak Ada	30	Porotfolio	2	Tidak Lulus

No	NIP	NUPTK	Tingkat Pendidi-kan	Sertifikasi Mata Pelajaran	Usia	Jenis Sertifikasi	Masa Kerja	Kete-rangan
6	196911101 992012002	44427476 49300033	S1	Ada	43	PLPG	20	Lulus
7	197804122 005012017	17447566 56300002	S1	Ada	34	PLPG	7	Lulus
8	198204272 006041014	77597606 61200002	S1	Tdak Ada	30	Portofolio	8	Tidak Lulus
9	197106272 006041015	39597496 51200012	S1	Ada	41	PLPG	6	Lulus
10	198103222 006041014	56547596 60200002	S1	Tidak Ada	31	Portofolio	6	Tidak Lulus

Data tersebut kemudian diinisialisasi sesuai tabel 3.1, tabel 3.2, dan tabel 3.3. Data yang telah terinisialisasi dan akan digunakan pada perhitungan dapat dilihat pada tabel 3.6.

Tabel 3.6. Data Sertifikasi Guru Terinisialisasi

No	NIP	NUP TK	Tingkat Pendidikan	Sertifikasi Mata Pelajaran	Usia	Jenis Sertifikasi	Masa Kerja	Kelas
1	2	4	5	7	10	12	13	Lulus
2	2	4	6	8	10	11	13	Tidak
3	1	4	6	8	10	11	13	Tidak
4	2	4	6	8	9	12	14	Lulus
5	2	4	6	8	10	12	14	Lulus
6	2	4	6	8	10	12	13	Lulus
7	1	3	5	7	10	11	14	Tidak
8	2	4	5	8	10	12	14	Lulus
9	1	4	6	8	9	11	13	Tidak
10	2	3	6	8	10	11	13	Tidak

Langkah kedua, dilakukan pembacaan pada data sertifikasi guru yang telah dilakukan *preprocessing* untuk menghitung frekuensi kemunculan tiap kombinasi *parameter* sertifikasi guru dan perhitungan *support* pada kandidat *frequent* 1-*itemset* seperti pada tabel 3.7.

Tabel 3.7. Nilai *Support* Kandidat 1-*Itemset*

Item (id)	*Support* Count(s)	*Support*	Item (id)	*Support* Count(s)	*Support*
1	3	0.3	8	8	0.8
2	7	0.7	9	2	0.2
3	2	0.2	10	8	0.8
4	8	0.8	11	5	0.5
5	3	0.3	12	5	0.5
6	7	0.7	13	6	0.6
7	2	0.2	14	4	0.4

Langkah ketiga adalah pencarian *frequent 1-itemset* dari kandidat *frequent 1-itemset*. Pada kandidat *frequent 1-itemset* yang telah dihitung nilai *support*nya, *itemset* {{1}, {3}, {5}, {7}, {9}, {14}} menjadi *itemset* yang tidak *frequent*, sehingga *itemset* {{2}, {4}, {6}, {8}, {10}, {11}, {12}, {13}} merupakan kandidat *item* yang *frequent* atau terpilih menjadi *frequent 1-itemset* dan dapat dilihat pada Tabel 3.8.

Tabel 3. 8. Nilai *Support Frequent* 1-*Itemset*

Frequent 1-Itemset	*Support*
2	0.7
4	0.8
6	0.7
8	0.8
10	0.8
11	0.5
12	0.5
13	0.6

Langkah keempat adalah tahap transformasi data dimana data hasil *preprocessing* data diubah bentuk sesuai dengan kebutuhan. Tahap transformasi yang dilakukan adalah mengubah bentuk tabel data *parameter* sertifikasi guru sehingga memudahkan proses *data mining*. Pada tahap transformasi ini, *parameter* yang memenuhi *minimum support* akan diberi kode id 1, sedangkan yang tidak memenuhi *minimum support* akan diberi kode id 0. Untuk kode klasifikasi kelas, untuk kelas Lulus disebut C1 dan diberi kode O dan kelas Tidak Lulus disebut kelas C2 akan diberi kode P. Jika termasuk kelas Lulus maka diberi kode id 1, sedangkan jika termasuk kelas Tidak Lulus maka diberi kode id 0. Tabel hasil transformasi dapat dilihat pada tabel 3.9 dan 3.10.

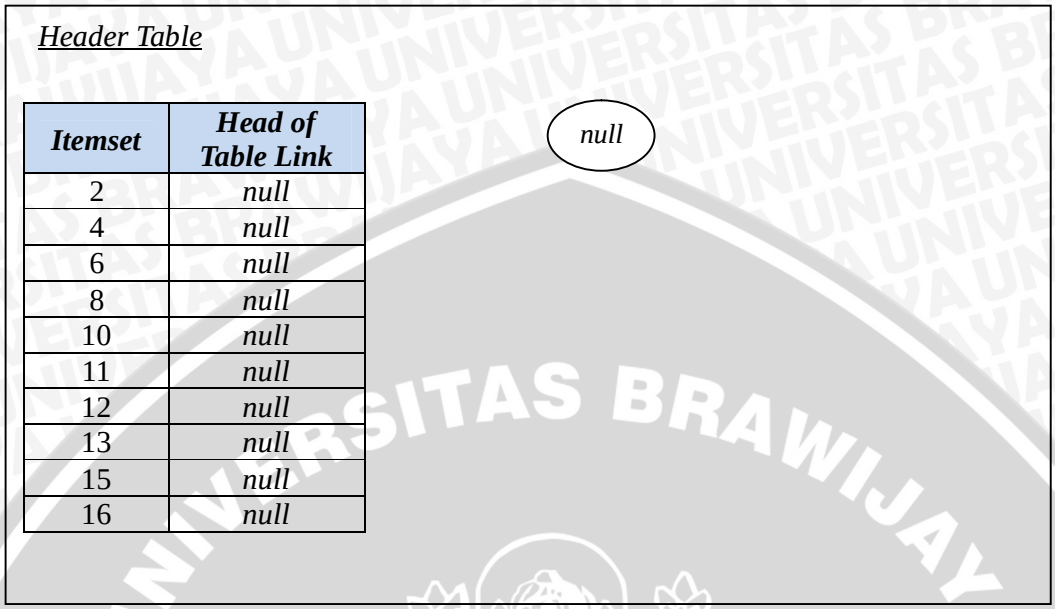
Tabel 3. 9. Tabel *Transformasi* Data *Parameter* Sertifikasi Guru

No	NIP		NUPTK		Tingkat Pendidikan		Sertifikasi Mata Pelajaran		Usia		Jenis Sertifikasi		Masa Kerja		Kelas C1	Kelas C2
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)
1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	1	0
2	0	1	0	1	0	1	0	1	0	1	1	0	1	0	0	1
3	0	0	0	1	0	1	0	1	0	1	1	0	1	0	0	1
4	0	1	0	1	0	1	0	1	0	0	0	1	0	0	1	0
5	0	1	0	1	0	1	0	1	0	1	0	1	0	0	1	0
6	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0
7	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1
8	0	1	0	1	0	0	0	1	0	1	0	1	0	0	1	0
9	0	0	0	1	0	1	0	1	0	0	1	0	1	0	0	1
10	0	1	0	0	0	1	0	1	0	1	1	0	1	0	0	1

Tabel 3. 10. Tabel *Itemset Parameter* Sertifikasi Guru

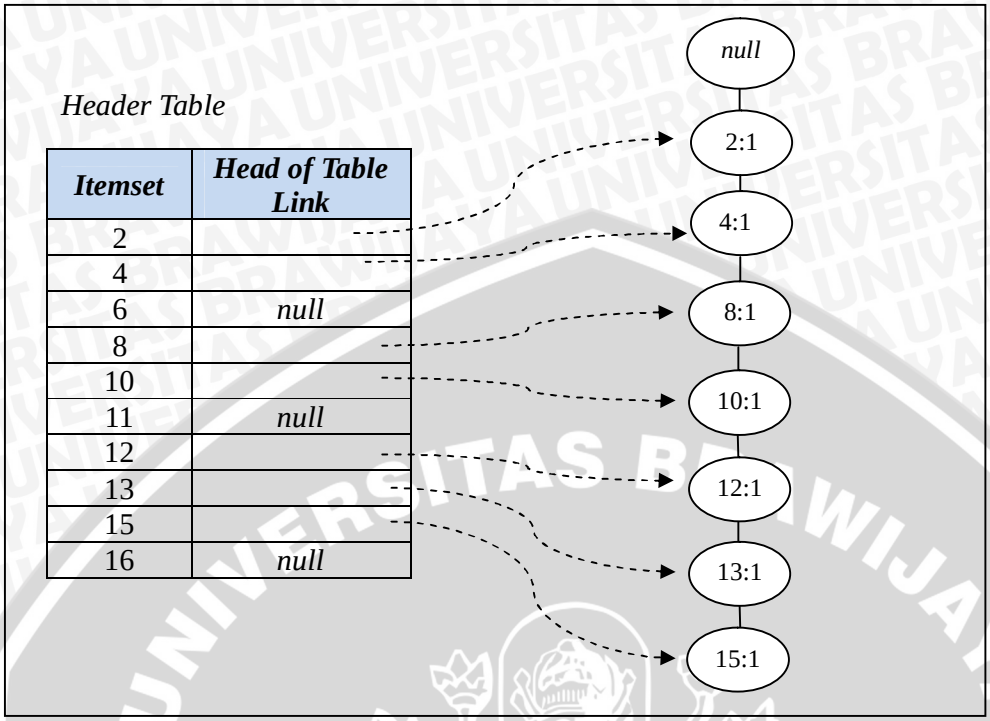
Id *Item*	*Itemset*
1	2, 4, 8, 10, 12, 13, 15
2	2, 4, 6, 8, 10, 11, 13, 16
3	4, 6, 8, 10, 11, 13, 16
4	2, 4, 6, 8, 12, 15
5	2, 4, 6, 8, 10, 12, 15
6	2, 4, 6, 8, 10, 12, 13, 15
7	10, 11, 16
8	2, 4, 8, 10, 12, 15
9	4, 6, 8, 11, 13, 16
10	2, 6, 8, 10, 11, 13, 16

Langkah kelima adalah setiap *itemset* dari sampel data *parameter* sertifikasi guru dibuat *FP-tree* yang disajikan pada Gambar 3.27 sampai Gambar 3.39. Pada pembuatan awal *root* diberikan nilai *null.* Pada *header table, head of table link* untuk semua kode *item* bernilai *null*.



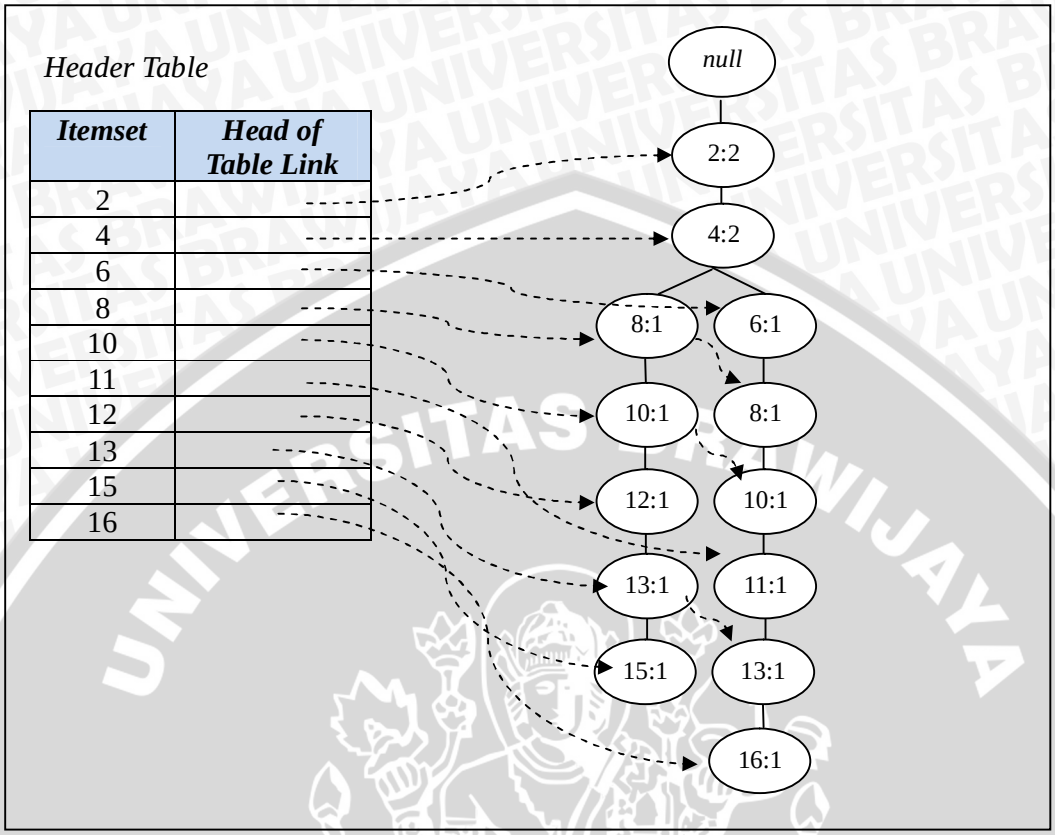
Gambar 3.28. FP-*Tree* Awal

Pembacaan pertama yaitu untuk pembacaan id *item* 1 adalah { 2, 4, 8, 10, 12, 13, 15}. *Itemset* 2 masuk terlebih dahulu ke dalam *FP-Tree* sehingga dibuat *node* baru dengan label 2 yang merupakan *node* anak dari *root* dan *count support* diberi nilai 1. Selain itu *itemset* 2 pada *header table* dibuat *link* yang menunjukkan *node* dimana informasi mengenai 2 disimpan. Selanjutnya *item* transaksi 4 masuk ke dalam *FP-Tree*, karena *node* 2 belum memiliki *node* anak berlabel 2 maka dibuat *node* baru dengan label 4 dan *count support node* 4 diberi nilai 1. Begitu pula dengan *itemset* 4, pada *header table* dibuat *link* yang merujuk ke *node* 4 tersebut. Proses ini berlanjut hingga *itemset* 15 dimasukkan ke dalam FP-*Tree*. *FP-Tree* yang terbentuk untuk pembacaan data id *item* 1 dapat dilihat pada Gambar 3.29.



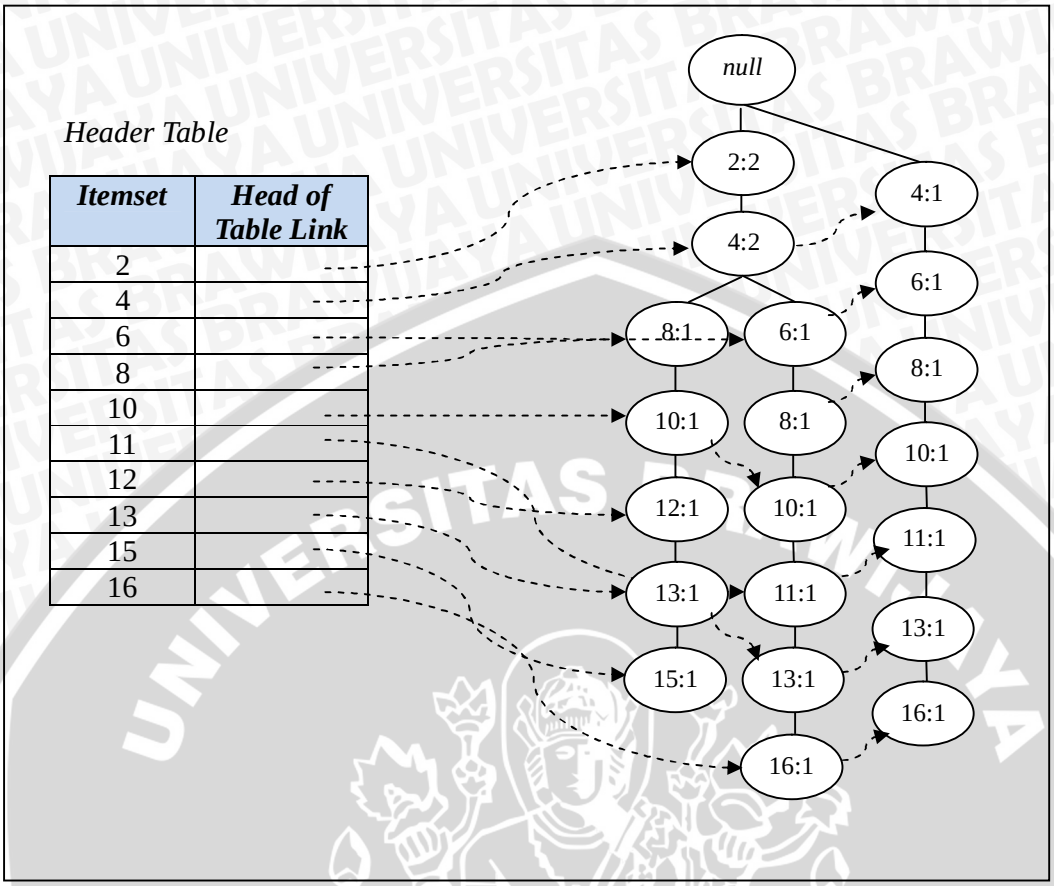
Gambar 3. 31. FP-*Tree* Pembacaan Id *Item* 1

Pembacaan kedua yaitu untuk id *item* 2 adalah { 2, 4, 6, 8, 10, 11, 13, 16 }. *Itemset* 2 sama dengan transaksi sebelumnya, sehingga tidak dibuat simpul anak baru dari root. *Itemset* 4 dimasukkan kedalam *node* yang sama dengan *count support* bertambah 1 (*count support* = 2). Selanjutnya *itemset* 6 masuk ke dalam *FP-Tree,* karena *node* 4 belum memiliki *node* anak berlabel 6 maka dibuat *node* baru dengan label 6 dan *count support node* 6 diberi nilai 1. Proses ini berlanjut hingga *itemset* 16 dimasukkan ke dalam FP-*Tree*. *FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 2 dapat dilihat pada Gambar 3.32.



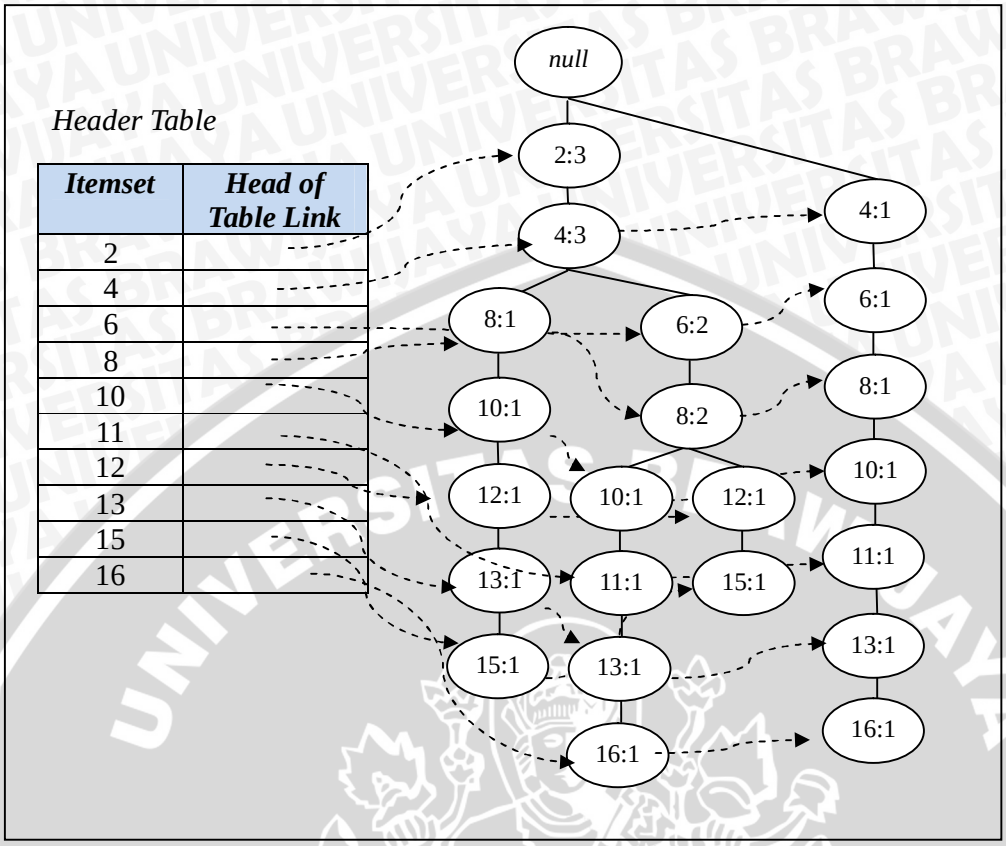
Gambar 3. 32. FP-*Tree* Pembacaan Id *Item* 2

Pembacaan ketiga yaitu untuk transaksi dengan id *item* 3 adalah { 4, 6, 8, 10, 11, 13, 16 }. *Item* 4 masuk ke dalam *FP-Tree* sehingga dibuat *node* baru dengan label D yang merupakan *node* anak dari *root* dan *count support* diberi nilai 1. Selanjutnya *item* transaksi F masuk ke dalam *FP-Tree*, karena *node* 4 belum memiliki *node* anak berlabel 6 maka dibuat *node* baru dengan label 6 dan *count support node* 4 diberi nilai 1. Proses ini berlanjut hingga *itemset* 16 dimasukkan ke dalam FP-*Tree. FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 3 dapat dilihat pada Gambar 3.33.



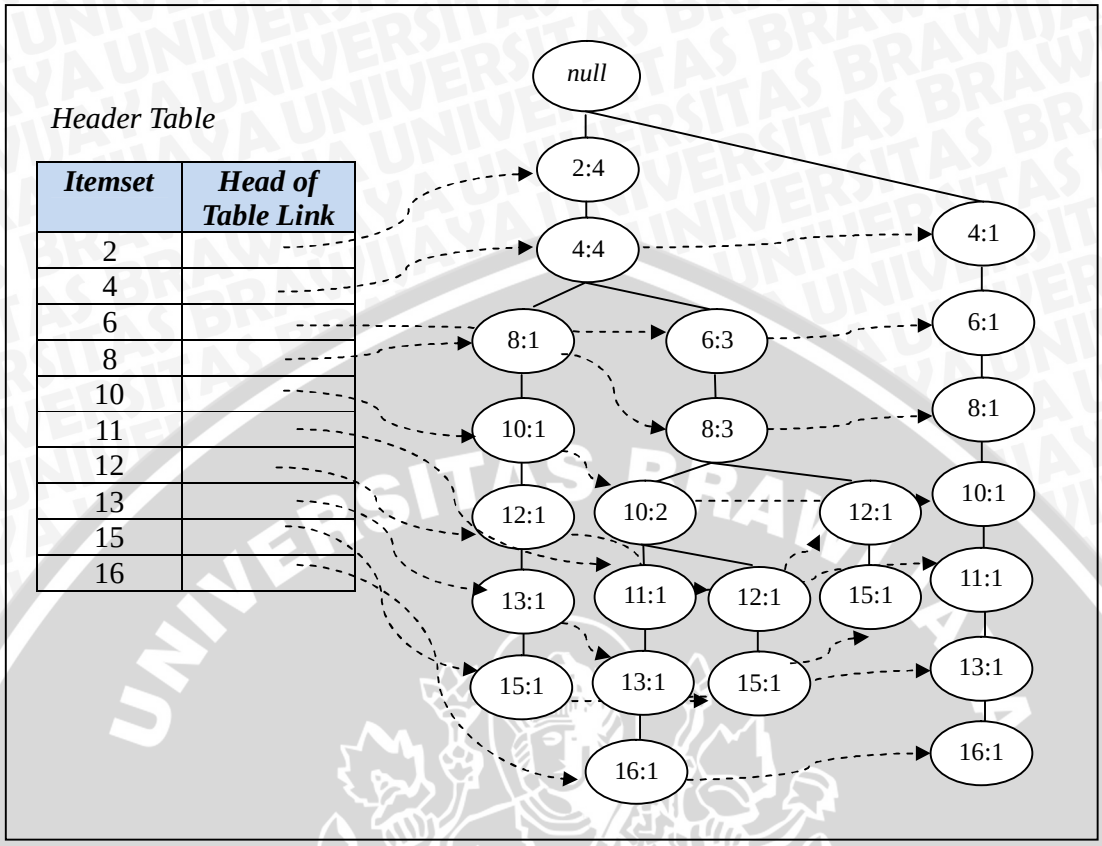
Gambar 3. 33. FP-*Tree* Pembacaan Id *Item* 3

Pembacaan keempat yaitu untuk transaksi dengan id *item* 4 adalah { 2, 4, 6, 8, 12, 15 }. *Itemset* 2 sama dengan transaksi sebelumnya, sehingga tidak dibuat simpul anak baru dari root. Kemudian *itemset* 4 dimasukkan kedalam *node* yang sama dengan *count support* bertambah 1 (*count support* = 3). Proses ini berlanjut hingga *itemset* 15 dimasukkan ke dalam FP-*Tree*. *FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 4 dapat dilihat pada Gambar 3.34.



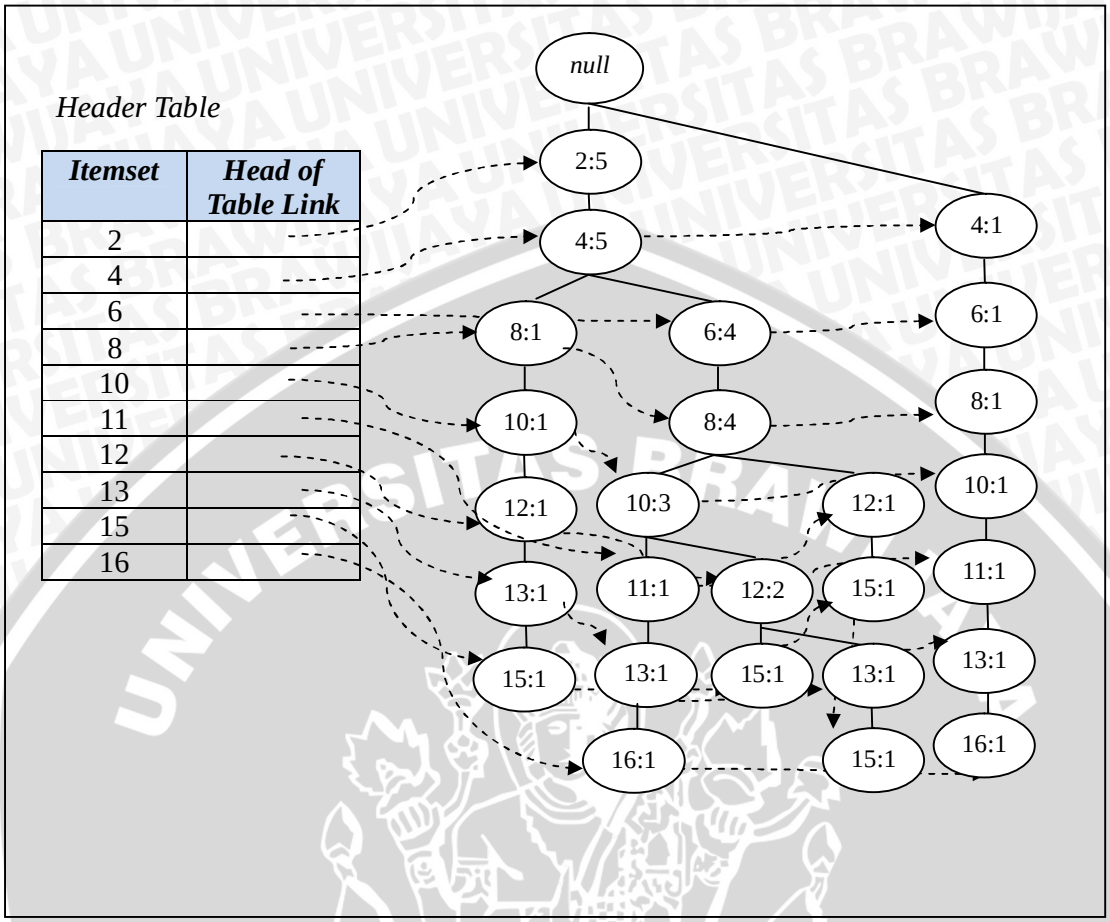
Gambar 3.34. FP-*Tree* Pembacaan Id *Item* 4

Pembacaan kelima yaitu untuk transaksi dengan id *item* 5 adalah { 2, 4, 6, 8, 10, 12, 15 }. *Itemset* 2 sama dengan transaksi sebelumnya, sehingga tidak dibuat simpul anak baru dari root. Kemudian *itemset* 4 dimasukkan kedalam *node* yang sama dengan *count support* bertambah 1 (*count support* = 4). Proses ini berlanjut hingga *itemset* 15 dimasukkan ke dalam FP-*Tree*. *FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 5 dapat dilihat pada Gambar 3.35.



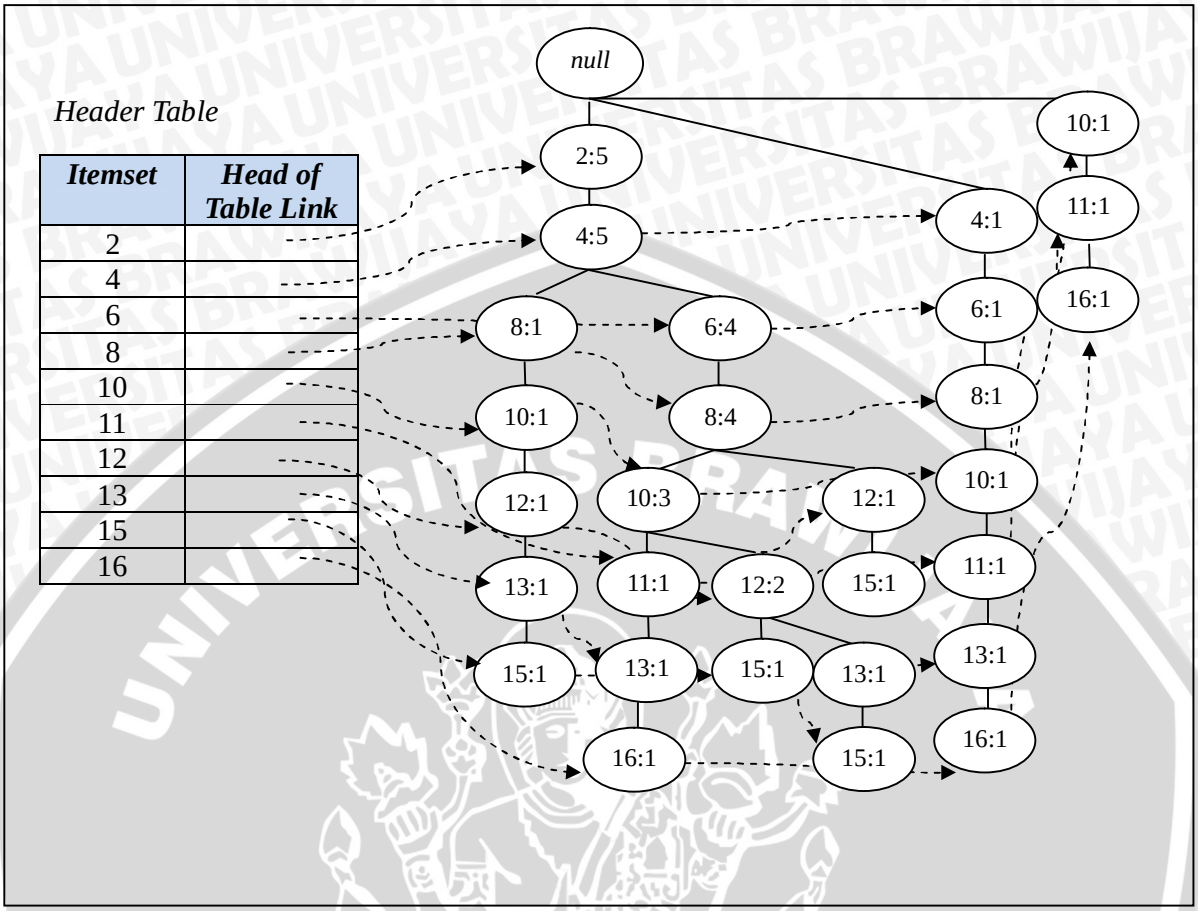
Gambar 3.35. FP-*Tree* Pembacaan Id *Item* 5

Pembacaan keenam yaitu untuk transaksi dengan id *item* 6 adalah { 2, 4, 6, 8, 10, 12, 13, 15 }. *Itemset* 2 sama dengan transaksi sebelumnya, sehingga tidak dibuat simpul anak baru dari root. Kemudian *itemset* 4 dimasukkan kedalam *node* yang sama dengan *count support* bertambah 1 (*count support* = 5). Proses ini berlanjut hingga *itemset* 15 dimasukkan ke dalam FP-*Tree*. *FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 6 dapat dilihat pada Gambar 3.36.



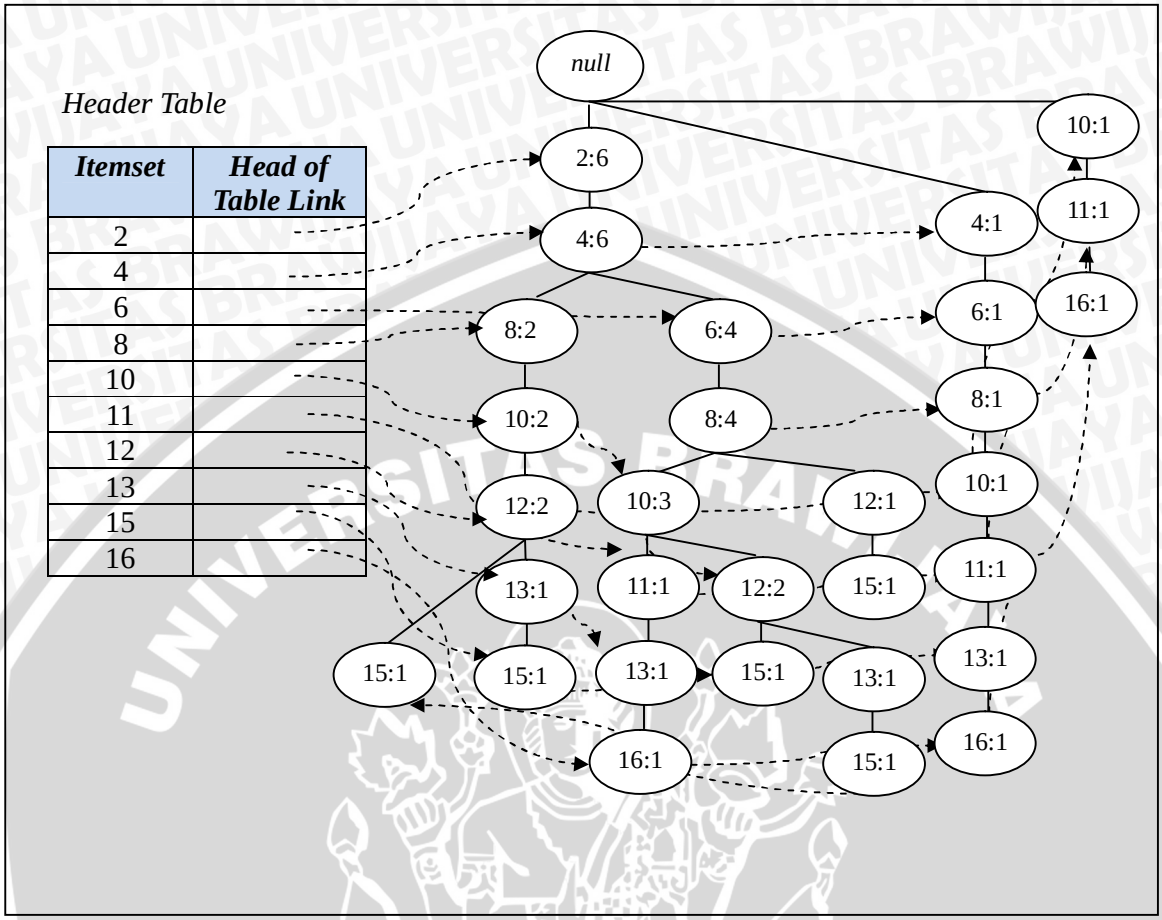
Gambar 3.36. FP-*Tree* Pembacaan Id *Item* 6

Pembacaan ketujuh yaitu untuk transaksi dengan id *item* 7 adalah { 10, 11, 16 }. *Item* 10 masuk ke dalam *FP-Tree* sehingga dibuat *node* baru dengan label 10 yang merupakan *node* anak dari *root* dan *count support* diberi nilai 1. Selanjutnya *item* transaksi 11 masuk ke dalam *FP-Tree,* karena *node* 10 belum memiliki *node* anak berlabel 11 maka dibuat *node* baru dengan label 11 dan *count support node* 11 diberi nilai 1. Proses ini berlanjut hingga *itemset* 16 dimasukkan ke dalam FP-*Tree. FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 7 dapat dilihat pada Gambar 3.37.



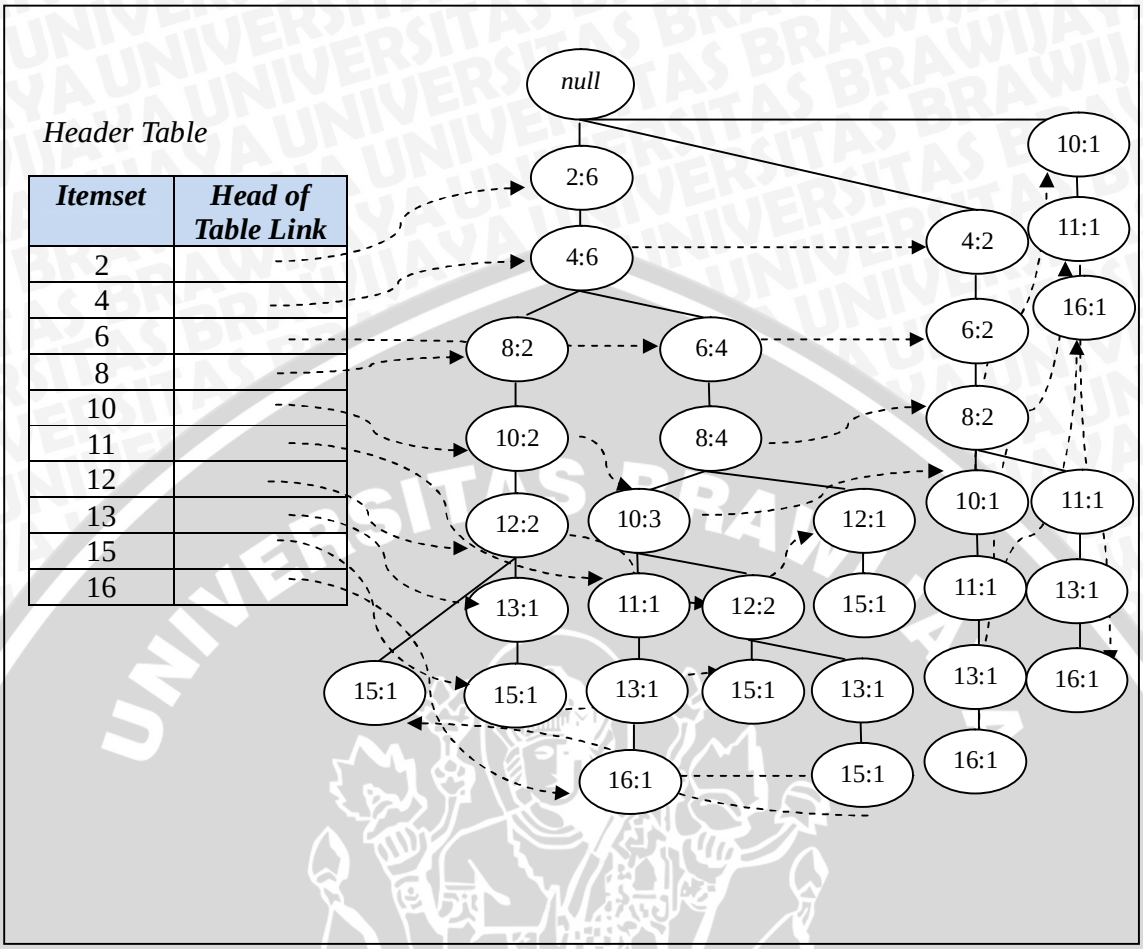
Gambar 3.37. FP-*Tree* Pembacaan Id *Item* 7

Pembacaan kedelapan yaitu untuk transaksi dengan id *item* 8 adalah { 2, 4, 8, 10, 12, 15 }. *Itemset* 2 sama dengan transaksi sebelumnya, sehingga tidak dibuat simpul anak baru dari *root*. Kemudian *itemset* 4 dimasukkan kedalam *node* yang sama dengan *count support* bertambah 1 (*count support* = 6). Proses ini berlanjut hingga *itemset* 15 dimasukkan ke dalam FP-*Tree*. *FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 8 dapat dilihat pada Gambar 3.38.



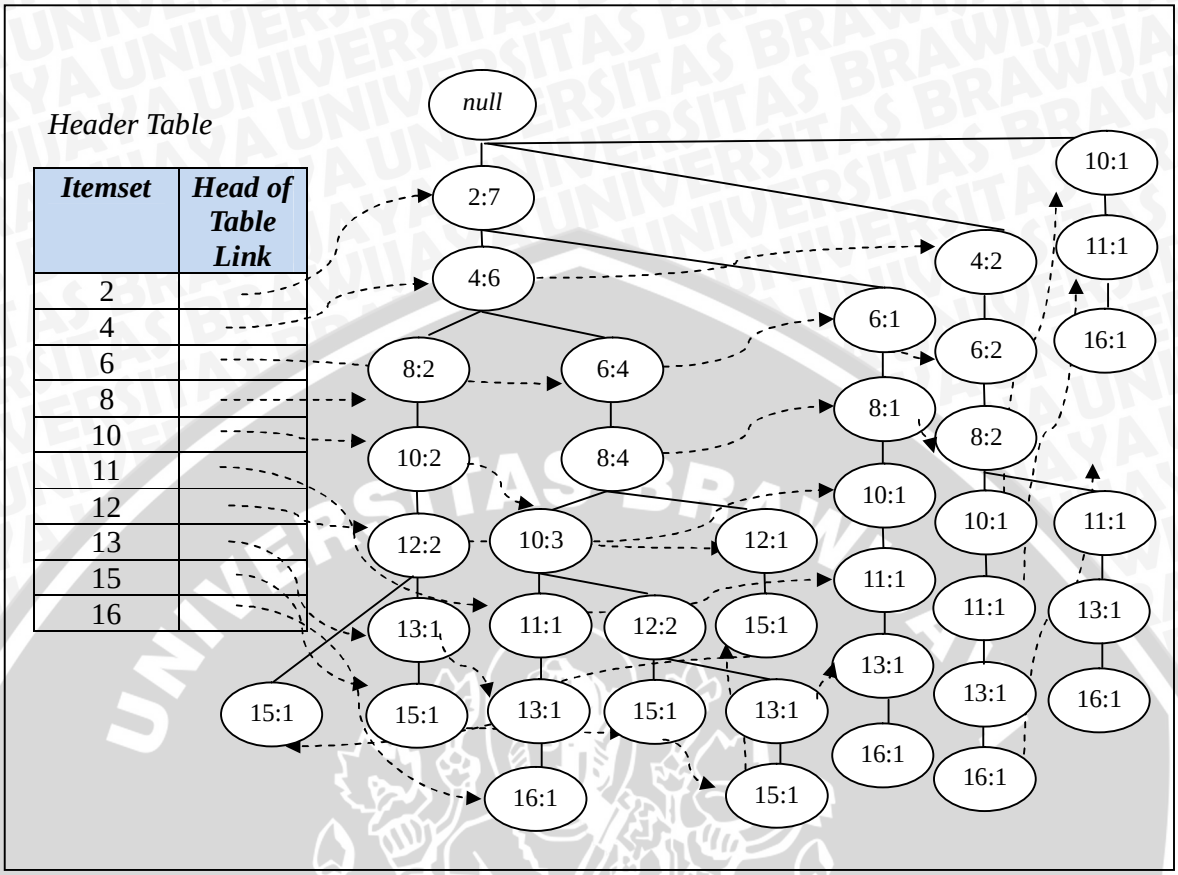
Gambar 3.38. FP-*Tree* Pembacaan Id *Item* 8

Pembacaan kesembilan yaitu untuk transaksi dengan id *item* 9 adalah { 4, 6, 8, 11, 13, 16 }. *Itemset* 4 sama dengan transaksi sebelumnya, sehingga tidak dibuat simpul anak baru dari root. Kemudian *itemset* 6 dimasukkan kedalam *node* yang sama dengan *count support* bertambah 1 (*count support* = 2). Proses ini berlanjut hingga *itemset* 16 dimasukkan ke dalam FP-*Tree*. *FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 9 dapat dilihat pada Gambar 3.39.



Gambar 3.39. FP-*Tree* Pembacaan Id *Item* 9

Pembacaan kesepuluh yaitu untuk transaksi dengan id *item* 10 adalah { 2, 6, 8, 10, 11, 13, 16 }. *Itemset* 2 sama dengan transaksi sebelumnya, sehingga tidak dibuat simpul anak baru dari root. Kemudian *itemset* 6 masuk ke dalam *FP-Tree,* karena *node* 2 belum memiliki *node* anak berlabel 6 maka dibuat *node* baru dengan label 6 dan *count support node* 2 diberi nilai 1. Proses ini berlanjut hingga *itemset* 16 dimasukkan ke dalam FP-*Tree*. *FP-Tree* yang terbentuk untuk pembacaan data transaksi dengan id *item* 10 dapat dilihat pada Gambar 3.40.



Gambar 3.40. FP-*Tree* Pembacaan Id *Item* 10

Langkah selanjutnya adalah *conditional pattern base* yaitu mencari *suffix pattern* dan *prefix path* pada *FP-tree* yang telah terbentuk. Pada tahap ini juga dibentuk *conditional FP-tree* yaitu *support count* setiap itemset *prefix path* setiap *suffix* dijumlahkan. Item yang memiliki jumlah *support count* lebih besar atau sama dengan *minimum support* akan dibangkitkan dengan *conditional FP-tree*. Proses klasifikasi memerlukan *conditional pattern base* yang memiliki *suffix pattern* 15 dan 16.

- *Suffix pattern* 16 : 5

  Conditional pattern base : {(2, 4, 6, 8, 10, 11, 13, 16) : 1, (2, 6, 8, 10, 11, 13, 16) : 1, (4, 6, 8, 10, 11, 13, 16) : 1, (4, 6, 8, 11, 13, 16) : 1, (10, 11, 16) : 1}

Dari *conditional pattern base* tersebut akan dibentuk *conditional FP-Tree* untuk mendapatkan *frequent itemset* yang memenuhi *minimum support*. Berdasarkan *conditional FP-Tree* yang terbentuk, tidak ada *frequent itemset* dengan *suffix pattern* 16 yang memenuhi *minimum support*.

- *Suffix pattern* 15 : 5

  Conditional pattern base : {(2, 4, 8, 10, 12, 13, 15) : 1, (2, 3, 8, 10, 12, 15) : 1, (2, 4, 6, 8, 10, 12, 15) : 1, (2, 4, 6, 8, 10, 12, 13, 15) : 1, (2, 4, 6, 8, 12, 15) : 1}

  Dari *Conditional pattern base* tersebut akan dibentuk *conditional FP-Tree* untuk mendapatkan *frequent itemset* seperti pada tabel 3.11 berikut.

Tabel 3.11. Tabel *Frequent Itemset Suffix Pattern* 15

Frequent Pattern	*Support*
2, 8, 15	0.5
2, 15	0.5
8, 15	0.5

Langkah selanjutnya adalah *frequent itemset* yang didapat akan dilakukan perhitungan nilai *confidence*. Nilai *confidence* ini akan menentukan apakah *frequent itemset* tersebut dapat dibangkitkan menjadi *rule*. Bila nilai *confidence* lebih besar atau sama dengan nilai *minimum confidence* yang telah ditentukan maka *frequent itemset* tersebut dapat dibangkitkan untuk menjadi *rule*, sedangkan bila nilai *confidence* lebih kecil dari nilai minimum *confidence* maka *frequent itemset* tersebut tidak dapat dibangkitkan menjadi *rule*. Nilai *confidence* dari *frequent itemset* yang telah didapat disajikan dalam Tabel 3.12.

Tabel 3.12. *Confidence Frequent Itemset* Sampel Data

No	*Frequent Itemset*	*Support (A,B)*	*Support (A)*	*Confidence*
1	2, 8, 15	0.5	0.5	1
2	2, 15	0.5	0.5	1
3	8, 15	0.5	0.5	1

Berdasarkan Tabel 3.12, semua *frequent itemset* terpilih untuk dibangkitkan menjadi *rule* karena nilai *confidence* masing – masing *frequent itemset* tersebut lebih dari nilai *minimum confidence* yang telah ditentukan, sehingga *rule* yang dibentuk adalah 3 *rule.*

Selanjutnya *rule* yang terbentuk dilakukan uji kekuatan *rule* dengan menggunakan *lift ratio*. *Lift ratio* didapatkan dari hasil perbandingan antara *confidence* dengan *benchmark confidence*. Hasil perhitungan nilai *lift ratio* dapat dilihat pada Tabel 3.13.

Tabel 3.13. *Lift Ratio* dari *Rule*

No	***Rule***	***Confidence***	***Benchmark Confidence***	***Lift Ratio***
1	2, 8, 15	1	0.5	2
2	2, 15	1	0.5	2
3	8, 15	1	0.5	2

Berdasarkan hasil perhitungan *lift ratio* pada Tabel 3.13, dapat dilihat bahwa semua *rule* memiliki nilai *lift ratio* lebih dari 1 sehingga menunjukkan adanya manfaat karena memiliki kekuatan asosiasi yang tinggi.

Langkah selanjutnya adalah proses penentuan kelayakan kelulusan sertifikasi guru. R*ule-rule* yang digunakan disajikan dalam Tabel 3.14.

Tabel 3.14. *Rule* yang Digunakan dalam Proses Klasifikasi

No	***Rule***	***Confidence***	***Lift Ratio***
1	2, 8, 15	1	2
2	2, 15	1	2

3	8, 15	1	2

Dari 3 *rule* yang akan digunakan ini akan dilatih menggunakan data latih dengan cara menghitung nilai DF setiap *rule* menggunakan rumus 2.13 untuk setiap data latih. Nilai DF *rule* kemudian diurutkan dari nilai terbesar ke nilai terkecil. Jika *rule* pertama mengklasifikasikan data latih dengan benar, maka nilai *rightN*-nya bertambah. Jika tidak maka *rule* diletakkan di urutan terakhir dan nilai *wrongN*-nya bertambah. Contoh hasil perhitungan nilai DF data ke-1 dapat dilihat pada tabel 3.15.

Tabel 3.15. Nilai DF Tiap *Rule*

No	*Rule*	*Confidence*	Nilai DF
1	2, 8, 15	1	0
2	2, 15	1	1
3	8, 15	1	0

Nilai DF yang telah didapat kemudian diurutkan secara menurun dan dapat dilihat pada Tabel 3.16.

Tabel 3.16. Nilai DF Terurut Data ke-1

No.	*Rule*	DF
1	**2**	1
2	**1**	0
3	**3**	0

Cek apakah kelas *rule* ke-2 sama dengan kelas data ke-1. Karena kelas *rule* ke-2 sama dengan kelas data ke-1 yaitu kelas C1 atau kelas Lulus, maka nilai R2.*rightN* menjadi 1 dan proses berpindah ke data ke-2.

Cek apakah kelas *rule* ke-2 sama dengan kelas data ke-2. Karena kelas *rule* ke-2 adalah kelas C1 sedangkan kelas data ke-2 adalah kelas C2 atau kelas Tidak Lulus, maka nilai R2.*wrongN* menjadi 1 dan *rule* ke-2 diletakkan di akhhir urutan. Proses berpindah ke *rule* ke-1.

Cek apakah kelas *rule* ke-1 sama dengan kelas data ke-1. Karena kelas *rule* ke-1 sama dengan kelas data ke-1 yaitu kelas C1 atau kelas Lulus, maka

nilai R1.*rightN* menjadi 1 dan proses berpindah ke data ke-2.

Jika semua *rule* telah dilatih pada semua data latih, maka dilakukan pengecekan pada setiap *rule*. Jika nilai *rightN rule* = 0 maka *rule* akan dibuang. Jika lebih dari 0 maka akan dimasukkan dalam *classifier* dan menghitung nilai *r.wk*. Nilai *r.wk* adalah hasil bagi *rightN* dengan *rightN* + *wrong N*. Selama jumlah *rule* dalam *classifier* lebih dari 1 maka *rule* dengan nilai *r.wk* terkecil akan dibuang. Jumlah *rightN* dan *wrongN* hasil pelatihan *rule* ditunjukkan pada Tabel 3.17.

Tabel 3.17. Hasil Pelatihan *Rule*

Rule	***rightN***	***wrongN***	***r.wk***
1	5	5	1
2	5	5	1
3	5	5	1

Hasil akhir *rule* yang masuk ke dalam *classifier* dapat dilihat pada Tabel 3.18.

Tabel 3.18. *Rule* dalam *Classifier*

No *Rule*	**Keterangan**	***Confidence***
1	JIKA ada NIP DAN ada sertifikasi mata pelajaran MAKA lulus (2, 8 → 15)	1
2	JIKA ada NIP MAKA lulus (2 → 15)	1
3	JIKA ada sertifikasi mata pelajaran MAKA lulus (8 → 15)	1

Untuk mengklasifikasikan data baru dilakukan perhitungan *discriminant function* (DF) menggunakan persamaan 2.13 untuk setiap kelas. Hasil perhitungan DF pada *record* ke-n lalu dijumlahkan sesuai dengan masing-masing kelas. Jumlah DF masing-masing kelas ini kemudian dibagi dengan jumlah seluruh nilai DF untuk menentukan hasil klasifikasi data uji.

Jika jumlah nilai DF pada kelas C1 lebih besar daripada kelas C2, maka data uji tersebut diklasifikasikan sebagai kelas C1. Sebaliknya, jika jumlah nilai DF pada kelas C2 lebih besar daripada kelas C1, maka data uji diklasifikasikan sebagai kelas C2. Jika jumlah nilai DF kelas C1 sama dengan kelas C2, maka dilihat jumlah *rule* kelas mana yang paling banyak dibangkitkan.

Misalnya diberikan 2 data uji yang diambil secara acak sebagai berikut :

No	NIP	NUPTK	Tingkat Pendidikan	Sertifikasi Mata Pelajaran	Usia	Jenis Sertifikasi	Masa Kerja	Kete-rangan	Kelas
1	195908171 980102007	31497376 39300083	S2	Ada	54	Portfolio	33	Lulus	?
2	-	31617326 35200003	S1	Ada	59	PLPG	46	Lulus	?

Setelah ditransformasi menjadi :

No	NIP		NUPTK		Tingkat Pendidikan		Sertifikasi Mata Pelajaran		Usia		Jenis Sertifikasi		Masa Kerja		Kelas C1	Kelas C2
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)
1	0	1	0	1	0	1	0	1	0	1	1	0	0	1	1	0
2	0	0	0	1	0	1	0	1	0	1	0	1	0	1	1	0

1. Menghitung *DF* untuk setiap *rule* dengan data ke-1:

Rule-1	= 1*1*1	= 1
Rule-2	= 1*1	= 1
Rule-3	= 1*1	= 1

Discriminant function pada kelas 1 :

$$g_1(y) = \frac{rule\ 1 + rule\ 2 + rule\ 3}{rule\ 1 + rule\ 2 + rule\ 3}$$

$$g_1(y) = \frac{1+1+1}{1+1+1} = 1$$

Discriminant function pada kelas 2:

$$g_2(y) = \frac{0}{rule\ 1 + rule\ 2 + rule\ 3}$$

$$g_2(y) = \frac{0}{1 + 1 + 1} = 0$$

Karena *discriminant function* untuk kelas C1 lebih besar dari pada kelas 2 maka data uji diklasifikasikan dalam kelas C1 yaitu kelas Lulus.

2. Menghitung *DF* untuk setiap *rule* dengan data ke-2 :

Rule-1	= 1*0*1	= 0
Rule-2	= 1*0	= 0
Rule-3	= 1*1	= 1

Discriminant function pada kelas 1 :

$$g_1(y) = \frac{rule\ 1 + rule\ 2 + rule\ 3}{rule\ 1 + rule\ 2 + rule\ 3}$$

$$g_1(y) = \frac{0 + 0 + 1}{0 + 0 + 1} = 1$$

Discriminant function pada kelas 2:

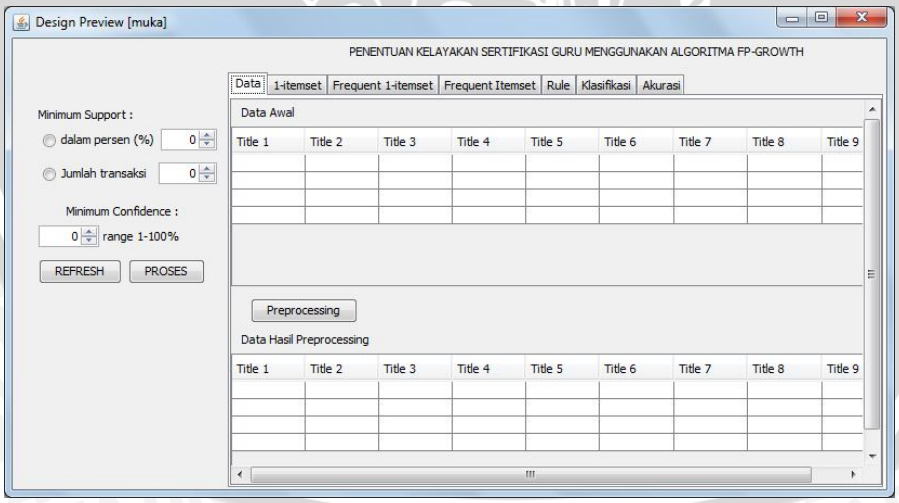
$$g_2(y) = \frac{0}{rule\ 1 + rule\ 2 + rule\ 3}$$

$$g_2(y) = \frac{0}{1} = 0$$

Karena *discriminant function* untuk kelas C1 lebih besar dari pada kelas C2 maka data uji diklasifikasikan dalam kelas C1 yaitu kelas Lulus.

3.5. Perancangan Antarmuka

Rancangan antarmuka secara umum terdiri dari bagian *input*, bagian *output*, tombol proses dan tombol *refresh*. Bagian *input* terdiri dari dua data *input*, yaitu data *input minimum support* dan data *input* minimum *confidence*. Bagian *output* terdiri dari empat *submenu* yaitu *submenu* Data Transaksi, *submenu 1-Itemset*, *submenu Frequent 1-Itemset*, *submenu Frequent Itemset*, *submenu Rule*, dan *submenu* Akurasi. Tampilan rancangan antarmuka secara umum dapat dilihat pada Gambar 3.48.



Gambar 3.48. Rancangan Antar Muka Secara Umum

3.6. Rancangan Uji Coba

Uji coba sistem untuk kelayakan sertifikasi guru ini adalah dilakukan

evaluasi terhadap hasil analisa yang dihasilkan oleh sistem. Tujuan dari uji coba ini adalah untuk mengetahui tingkat kekuatan (*lift ratio*) dari *rule* yang dihasilkan dan menentukan kelayakan sertifikasi guru. Selain itu, uji coba ini dilakukan juga untuk mengetahui akurasi *rule* yang dihasilkan terhadap data uji.

Sistem memiliki tabel *frequent itemset* dan tabel kandidat rule. Tabel *frequent itemset* ini menampilkan *itemset* yang lolos minimum *support*, yaitu yang memiliki nilai *support* lebih besar dari atau sama dengan minimum *support*. Tabel kandidat *rule* menampilkan hasil perhitungan nilai *confidence* setiap *frequent itemset* dan ditunjukkan pada Tabel 3.19.

Tabel 3.19. Tabel Rancangan *Frequent Itemset*

Frequent Itemset	***Conditional Pattern Base***	***Support***	***Confidence***

Tabel rancangan akurasi *rule* yang ditunjukkan pada Tabel 3.20 adalah *frequent itemset* yang lolos minimum *confidence*, yaitu yang memiliki nilai *confidence* lebih besar dari atau sama dengan minimum *confidence.*

Tabel 3.20. Tabel Rancangan *Rule*

Rule	***Conditional Pattern Base***	***Confidence***	***Lift Ratio***

Pada perancangan uji coba akan diuji berdasarkan jumlah *rule*, rata-rata *confidence*, rata-rata akurasi kalsifikasi, *lift ratio* dan pengujian terhadap data uji. Pengujian pertama yaitu pengujian jumlah *rule*, rata-rata *confidence*, dan rata-rata akurasi klasifikasi. Pengujian ini menggunakan parameter jumlah data, minimum *support* dan minimum *confidence.* Jumlah data yang digunakan adalah 250, 500, 750, dan 1000. Nilai minimum *support* yang

digunakan adalah 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90% dan 100%. Nilai minimum *confidence* yang digunakan adalah 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90% dan 100%. Perancangan uji coba berdasarkan jumlah *rule,* rata-rata *confidence,* dan rata-rata akurasi klasifikasi dapat dilihat pada tabel 3.21.

Tabel 3.21. Tabel Rancangan Uji Coba Jumlah *Rule,* Rata-Rata *Confidence* dan Akurasi Klasifikasi

Jumlah Data	Minimum *Support*	Minimum *Confidence*	Jumlah *Rule*	Rata-rata *Confidence*	Rata-rata *Lift Ratio*	Rata-rata Akurasi (%)

Pengujian kedua adalah pengujian *lift ratio*. Tabel uji coba berdasarkan *lift-ratio* dapat dilihat pada tabel 3.22.

Tabel. 3.22. Tabel Uji Coba *Lift Ratio*

Rule	Nilai *confidence*	Lift Ratio

Pengujian ketiga yaitu pengujian terhadap data uji. Pengujian ini

menggunakan parameter dengan jumlah *rule* terbanyak untuk lift ratio lebih dari 1. Data yang digunakan untuk pengujian ini adalah data sertifikasi guru tahun terakhir untuk mengecek kebenaran dari *rule-rule* yang telah dihasilkan. Tabel pengujian akurasi terhadap data uji dapat dilihat pada tabel 3.23.

Tabel. 3.23. Tabel Pengujian Akurasi Terhadap Data Uji

Rule	Akurasi (%)