IMPLEMENTASI METODE *ASSOCIATION RULE* DENGAN ALGORITMA *FP GROWTH* PADA DATA HASIL TANGKAPAN IKAN LAUT

SKRIPSI

Untuk memenuhi sebagian persyaratan

mencapai gelar Sarjana Komputer

Disusun oleh :

NOVADYANA SETYANINGRUM

NIM. 0710963016

PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2014

LEMBAR PERSETUJUAN

IMPLEMENTASI METODE *ASSOCIATION RULE* DENGAN ALGORITMA *FP GROWTH* PADA DATA HASIL TANGKAPAN IKAN LAUT

SKRIPSI

Disusun oleh :

Novadyana Setyaningrum

NIM. 0710963016

Skripsi ini telah disetujui oleh dosen pembimbing pada tanggal 25 Agustus 2014

Pembimbing I,	Pembimbing II,
Candra Dewi, S.Kom., M.Sc	**Budi Darma S., S.Kom., M.Cs.**
NIP. 197711142003122001	**NIK. 84101506110090**

LEMBAR PENGESAHAN

IMPLEMENTASI METODE *ASSOCIATION RULE* DENGAN ALGORITMA *FP GROWTH* PADA DATA HASIL TANGKAPAN IKAN LAUT

SKRIPSI

LABORATORIUM KOMPUTASI CERDAS DAN VISUALISASI

Untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Disusun oleh :

Novadyana Setyaningrum

NIM. 0710963016

Setelah dipertahankan di depan Majenlis Penguji
pada tanggal 25 Agustus 2014
dan dinyatakan memenuhi syarat untuk memperoleh
gelar Sarjana dalam bidang Ilmu Komputer

Penguji I

Hurriyatul Fitriyah, ST., M.Sc

Penguji II

Ahmad Afif S., S.Si., M.Kom
NIK. 82062316110425

Penguji III

Lailil Muflikhah, S.Kom., M.Sc
NIP. 19741113 200501 2 001

Mengetahui
Ketua Program Studi Informatika/Ilmu Komputer

Drs. Marji, MT
NIP. 196708011992031001

PERNYATAAN ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata didalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, Agustus 2014

Mahasiswa,

NovadyanaSetyaningrum

NIM 0710963016

KATA PENGANTAR

Puji syukur Penulis panjatkan ke hadirat Allah SWT yang telah melimpahkan segala rahmat, karunia dan hidayah-Nya sehingga Penulis dapat menyelesaikan skripsi dengan judul "**Implementasi Metode *Association Rule* dengan Algoritma *FP Growth* pada Data Hasil Tangkapan Ikan Laut**". Tak lupa, mengucap shalawat dan salam kepada Nabi Muhammad *Shalallahu Alaihi Wasallam* sebagai obat penyejuk hati dan kunci segala kemudahan.

Skripsi ini diajukan sebagai syarat ujian seminar skripsi dalam rangka untuk memperoleh gelar Sarjana Komputer di Program Tekonologi Informasi dan Ilmu Komputer Universitas Brawijaya Malang. Atas terselesaikannya skripsi ini, Penulis mengucapkan terima kasih kepada:

1. Candra Dewi, S.Kom., M.Sc., selaku Dosen Pembimbing Skripsi I.
2. Budi Darma Setiawan, S.Kom., M.Cs., selaku Dosen Pembimbing Skripsi II.
3. Drs. Marji, MT., selaku Ketua Program Studi Informatika/Ilmu Komputer Program Teknologi Informasi & Ilmu Komputer Universitas Brawijaya.
4. Lailil Muflikhah, S.Kom., M.Sc., selaku Dosen Penasehat Akademik.
5. Ir. Sutrisno, MT., selaku Ketua Program Teknologi Informasi & Ilmu Komputer Universitas Brawijaya.
6. Issa Arwani, S.Kom., M.Sc., selaku Sekretaris Program Studi Informatika/Ilmu Komputer.
7. Wiwin Lukitohadi, S.H., S.Psi., CHRM., selaku Dosen Bimbingan Konseling Kelas Motivasi 2007.
8. Segenap Bapak dan Ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada Penulis selama menempuh pendidikan di Program Teknologi Informasi & Ilmu Komputer Universitas Brawijaya.
9. Orang tua Penulis dan kakak adikku tercinta atas segala dukungan materi dan doa restunya.
10. Che, Mas Anam, Pak Yudi, Pak Pardi dan rekan-rekan Ilmu Komputer Kelas Motivasi angkatan 2007 yang telah memberikan dukungannya kepada penulis.

11. I Fata Sagedistira, atas dukungan, kritik, saran, doa, kebersamaan dan kekompakan yang membantu membangkitkan semangatku selama di bangku perkuliahan dan penyusunan skripsi ini.
12. Momo, Oreo dan Chitato tercinta yang telah menjadi teman setia Penulis.
13. Mas Rhond yang telah membantu terselesaikannya skripsi ini.
14. Semua pihak yang telah membantu terselesaikannya skripsi ini yang tidak dapat kami sebutkan satu per satu.

Semoga penulisan laporan skripsi ini bermanfaat bagi pembaca sekalian. Sebagai manusia, penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan dan banyak kekurangan, dengan kerendahan hati penulis mengharapkan kritik dan saran yang membangun dari pembaca.

Malang, Agustus 2014

Novadyana Setyaningrum

Penulis

ABSTRAK

Novadyana Setyaningrum. 2014. : Implementasi Metode *Association Rule* dengan Algoritma *FP Growth* pada Data Hasil Tangkapan Ikan Laut
Dosen Pembimbing : Candra Dewi, S.Kom., M.Sc., dan Budi Darma Setiawan, S.Kom., M.Cs.

Saat ini sumber daya alam yaitu hasil perikanan laut di Indonesia perlu menjadi perhatian khusus para pihak yang terkait. Salah satu hal yang perlu diperhatikan adalah mengenai informasi daerah penangkapan ikan untuk para nelayan. Setiap daerah mempunyai potensi jenis ikan laut yang jumlahnya berbeda-beda, hal ini dapat menimbulkan terjadinya permasalahan yaitu terdapat daerah tertentu yang kekurangan stok ikan, sedangkan pada daerah lain mengalami kelebihan stok. Tentu saja untuk mengatasi permasalahan ini harus diimbangi dengan adanya peningkatan teknologi dan sistem yang mampu memberikan informasi dan data yang akurat mengenai daerah penangkapan ikan beserta potensinya. Teknik khusus *data mining* yang digunakan dalam penelitian ini yaitu metode *association rule* dengan algoritma *FP Growth*. Metode tersebut akan digunakan untuk menganalisa data sehingga didapatkan pola data berupa pola ikan yang muncul di suatu daerah. Hasil pola ikan yang muncul tersebut akan dilakukan perhitungan nilai *lift ratio* (tingkat kekuatan) untuk mengetahui seberapa kuat peluang munculnya pola data yang dihasilkan. Dengan demikian, berdasarkan analisa pola data ikan diharapkan dapat membantu para nelayan dalam menangkap ikan dan mendistribusikannya.

Penelitian ini menggunakan data dari Dinas Perikanan dan Keluatan Jawa Timur. Setelah dilakukan uji coba didapatkan bahwa metode *association rule* dengan algoritma *FP Growth* dapat diimplementasikan pada sebuah perangkat lunak dengan menghasilkan *rule* dengan nilai *lift ratio rule*. Pada penelitian ini didapatkan nilai *lift ratio* dari *rule* yang dihasilkan dengan *minimum confidence* 50% adalah nilai *lift ratio* tertinggi sebesar 2 dan nilai *lift ratio* terendah sebesar 1. Nilai rata-rata *lift ratio* tertinggi sebesar 1.86 yang terdapat pada Pamekasan.

Kata Kunci : *Association rule*, Algoritma *FP Growth*, *Lift Ratio*

ABSTRACT

Novadyana Setyaningrum. 2014. : Implementation of Association Rule Method with FP Growth Algorithm on Marine Fish Catch Data
Advisor : Candra Dewi, S.Kom., M.Sc., and Budi Darma Setiawan, S.Kom., M.Cs.

Current natural resource that is the result of marine fisheries in Indonesia needs to be special attention of the parties concerned. One thing to note is the information regarding the fishing area for fishermen. Each region has a potential marine fish species whose numbers vary, this can lead to a problem that there are certain areas that lack fish stocks, whereas in other areas having excess stock. Of course, to solve this problem must be balanced by an increase in technology and systems that are capable of providing accurate information and data regarding fishing areas and opportunities. Special techniques of data mining that is used in this study is the association rule method with FP Growth algorithm. Methods will be used to analyze the data so obtained data patterns that emerge in the form of a pattern of fish in an area. The results of the emerging patterns of fish that will be the calculation of the lift ratio value (power level) to determine how strong probability that the data pattern generated. Thus, based on the analysis of the data pattern of fish is expected to help the fishermen in catching fish and distribute them.

This study uses data from the Department of Fisheries and Marine East Java. After testing found that the association rule method with FP Growth algorithm can be implemented in software by generating a rule with lift ratio rule value. In this study, the lift ratio value generated by the minimum confidence value 50% is the highest lift ratio value at 2 and the lowest lift ratio value at 1. This research have resulted 1.86 in average value of lift ratio found in Pamekasan.

Keywords: *Association rule, FP Growth Algorithm, Lift Ratio*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR PERSAMAAN

DAFTAR *PSEUDOCODE*

DAFTAR *SOURCE CODE*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Saat ini sumber daya alam yaitu hasil perikanan laut di Indonesia perlu menjadi perhatian khusus para pihak yang terkait. Salah satu hal yang perlu diperhatikan adalah mengenai informasi daerah penangkapan ikan untuk para nelayan. Setiap daerah mempunyai potensi jenis ikan laut yang jumlahnya berbeda-beda, hal ini dapat menimbulkan terjadinya permasalahan yaitu terdapat daerah tertentu yang kekurangan stok ikan, sedangkan pada daerah lain mengalami kelebihan stok. Tentu saja untuk mengatasi permasalahan ini harus diimbangi dengan adanya peningkatan teknologi dan sistem yang mampu memberikan informasi dan data yang akurat mengenai daerah penangkapan ikan beserta potensinya.

Solusi yang dapat dilakukan adalah dengan memanfaatkan laporan data statistik dari Dinas Perikanan dan Kelautan Provinsi Jawa Timur dan kemudian dilakukan analisis data hasil tangkapan ikan laut. Ketika telah dilakukan suatu analisis data, sehingga dapat diketahui informasi bahwa pada masing-masing daerah penangkapan tertentu terdapat jenis ikan apa saja yang muncul. Dengan demikian, para nelayan atau instansi terkait dapat menentukan arah ke daerah mana yang akan dituju sebagai daerah penangkapan.

Menangani suatu data dengan jumlah ratusan hingga ribuan data akan terjadi kesulitan jika hanya menggunakan teknik yang sederhana ketika menganalisis data tersebut, sehingga diperlukan suatu teknik khusus yang mempunyai kemampuan untuk menganalisis data berskala besar. Teknik khusus yang tepat dan dapat membantu pencarian pola atau hubungan asosiatif dari data berskala besar adalah teknik *data mining*.

Data mining adalah serangkaian proses untuk menggali nilai tambah dari suatu kumpulan data berupa pengetahuan yang selama ini tidak diketahui secara manual. Ada beberapa metode atau teknik yang dikenal di dalam *data mining* salah satunya adalah *association rule* atau aturan asosiasi [KUS-07]. *Association rule* digunakan untuk menemukan hubungan di antara data atau bagaimana suatu

kelompok data mempengaruhi suatu keberadaan data yang lain. Pencarian *association rule* dilakukan melalui dua tahap yaitu pencarian *frequent itemset* (himpunan data yang sering muncul) dan penyusunan *rule* (aturan). *Frequent itemset* adalah bagian yang terpenting dari *association rule* [RUL-08].

Ada banyak algoritma pada metode *association rule* yang sudah dikembangkan, yaitu *AIS, SETM, Apriori, Apriori-TID, Apriori-Hybrid, Off-line Candidate Determination (OCD), Partitioning, Sampling, Dynamic Itemset Counting (DIC), Continous Association Rule Mining Algorithm (CARMA), FP-Growth, Charm, MagnumOpus*, dan lain-lain [SEL-08].

Algoritma *FP Growth* merupakan pengembangan dari algoritma *Apriori*. Sehingga kekurangan dari algoritma *Apriori* diperbaiki oleh algoritma *FP-Growth. Frequent Pattern Growth* (*FP Growth*) adalah salah satu alternatif algoritma yang dapat digunakan untuk menentukan himpunan data yang paling sering muncul (*frequent itemset*) dalam sebuah kumpulan data [SAM-08].

Kelebihan dari algoritma *FP Growth* juga dijelaskan antara lain hanya membutuhkan dua kali proses pembacaan *dataset*, tidak ada pembangkitan kandidat (*candidate generation*), lebih cepat daripada *Apriori*, dan memampatkan *dataset* [VER-08].

Penggunaan salah satu algoritma *association rule* yaitu algoritma *Apriori*, algoritma tersebut pernah digunakan pada suatu penelitian yang menganalisis pola data hasil tangkapan ikan laut. Seiring perkembangan algoritma baru yang telah dikembangkan para ahli, dengan algoritma *FP Growth* akan digunakan dalam penelitian yang sama yaitu penelitian untuk menganalisis pola data hasil tangkapan ikan laut. Pemilihan metode *association rule* dengan algoritma *FP Growth* dalam penelitian ini merupakan bentuk kelanjutan dari penelitian yang sudah ada.

Berdasarkan latar belakang yang telah dijelaskan tersebut, maka teknik yang digunakan pada penelitian ini adalah menggunakan metode *association rule* dengan algoritma *FP Growth* untuk mengetahui pola data hasil tangkapan ikan laut dengan judul penelitian "**Implementasi Metode *Association Rule* dengan Algoritma *FP Growth* pada Data Hasil Tangkapan Ikan Laut**".

1.2 Rumusan Masalah

Rumusan masalah pada skripsi ini adalah sebagai berikut:

1. Bagaimana mengimplementasikan metode *association rule* dengan algoritma *FP Growth* pada data hasil tangkapan ikan laut.
2. Berapa tingkat kekuatan (*lift ratio*) *rule* yang terbentuk dari hasil penerapan algoritma *FP Growth*.

1.3 Batasan Masalah

Adapun batasan masalah pada skripsi ini adalah sebagai berikut :

1. Data yang digunakan pada penelitian ini adalah data nama daerah dan data jenis ikan yang diperoleh dari laporan statistik Dinas Perikanan dan Kelautan Provinsi Jawa Timur, sedangkan untuk data waktu penangkapan (kuartal) dan jumlah ikan yang ditangkap adalah data simulasi.
2. Pembahasan pada data hanya menggunakan data penangkapan ikan di laut saja, tidak membahas data penangkapan ikan di air tawar dan ikan hasil budidaya.
3. Variabel penentu yang digunakan adalah nama jenis ikan, nama daerah penangkapan (kabupaten atau kota), waktu penangkapan (kuartal) dan jumlah ikan yang ditangkap.

1.4 Tujuan

Tujuan yang ingin dicapai dari skripsi ini adalah:

1. Mengimplementasikan metode *association rule* dengan algoritma *FP Growth* pada data hasil tangkapan ikan laut.
2. Mengetahui tingkat kekuatan (*lift ratio*) *rule* yang dihasilkan dari penerapan algoritma *FP Growth*.

1.5 Manfaat

Dengan diperolehnya pola data hasil tangkapan ikan laut yang telah digali menggunakan *association rule* ini, diharapkan mampu memberikan informasi mengenai daerah penangkapan ikan dan potensinya sehingga dapat membantu nelayan atau instansi terkait dalam menangkap ikan.

1.6 Sistematika Pembahasan

Sistematika penulisan skripsi terdiri dari 6 bab yaitu sebagai berikut :

BAB I PENDAHULUAN

Bab ini menguraikan latar belakang masalah yang akan dibahas, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika pembahasan.

BAB II KAJIAN PUSTAKA DAN DASAR TEORI

Bab ini menjelaskan teori-teori *data mining*, *association rule*, algoritma *FP Growth*, *lift ratio* dan teori lain yang merupakan konsep dasar penelitian.

BAB III METODE PENELITIAN DAN PERANCANGAN

Bab ini menjelaskan metode yang digunakan dan rancangan sistem yang akan dibangun dalam menyelesaikan penelitian ini.

BAB IV IMPLEMENTASI

Bab ini membahas proses implementasi dari sistem yang telah dibuat.

BAB V PENGUJIAN DAN ANALISIS

Bab ini menjelaskan tentang uji coba yang dilakukan beserta pembahasannya.

BAB VI PENUTUP

Pada bab ini akan menujukkan kesimpulan dan saran yang dapat digunakan untuk pengembangan berikutnya.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

Berdasarkan judul penelitian Implementasi Metode *Association Rule* dengan Algoritma *FP-Growth* pada Data Hasil Tangkapan Ikan Laut, penulis menemukan beberapa pustaka yang relevan untuk mendukung penelitian ini antara lain :

2.1 Data

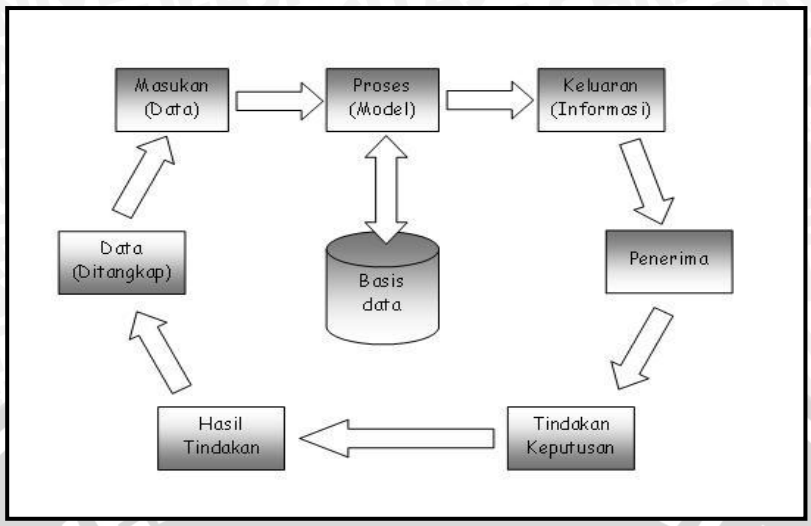
Keterangan atau bukti mengenai suatu kenyataan yang masih mentah, masih berdiri sendiri, belum diorganisasikan, dan belum diolah disebut data. Contoh data yang didapatkan dari hasil sumber daya alam misalnya, dapat berupa sekumpulan keterangan seperti jenis sumber daya, daerah penghasil sumber daya, dan jumlah yang dihasilkan pada tiap tahunnya. Data dapat dinyatakan dalam bentuk angka, simbol, maupun huruf yang apabila disusun sedemikian rupa akan menjadi informasi [AMS-00].

Menurut [KAD-03] data merupakan deskripsi tentang benda, kejadian, aktivitas dan tidak berpengaruh langsung kepada pengguna sehingga data merupakan bentuk yang masih mentah yang belum dapat bercerita banyak. Data yang melalui proses transformasi akan menghasilkan informasi.

2.2 Informasi

Informasi adalah data yang telah diatur dan diproses sehingga memberikan manfaat bagi para penggunanya. Dalam mengambil keputusan atau untuk meningkatkan proses pengambilan keputusan akan dibutuhkan informasi. Ketika suatu informasi yang mempunyai kualitas dan kuantitas yang meningkat, maka pengguna dapat membuat keputusan yang lebih baik [ROM-05].

Data diolah melalui suatu model menjadi sebuah infomasi, penerima kemudian menerima informasi tersebut, membuat seuatu keputusan dan melakukan tindakan, yang berarti menghasilkan tindakan lain yang akan membuat sejumlah data kembali, data yang di tangkap dianggap sebagai masukan di proses kembali melalui model, dan begitu seterusnya membentuk sebuah siklus. Data dan informasi yang membentuk siklus pada dilihat pada Gambar 2.1.



Gambar 2.1 Data dan Informasi Membentuk Siklus

2.3 *Data Mining*

2.3.1 Pengertian *Data Mining*

Kehadiran *data mining* dilatar belakangi dengan adanya permasalahan ledakan data atau informasi yang dialami akhir-akhir ini dimana banyak organisasi yang telah mengumpulkan data sekian tahun lamanya [SUC-03].

Secara sederhana definisi dari *data mining* adalah suatu penambangan atau penemuan informasi baru dengan mencari pola atau aturan tertentu dari sejumlah data yang sangat besar [DAV-04]. *Data mining* juga disebut sebagai serangkaian proses untuk mengetahui nilai tambah dari suatu pengetahuan yang selama ini tidak diketahui karena hanya digali secara manual dari suatu kumpulan data [PRA-03].

Data mining adalah kegiatan menemukan pola yang menarik dari data dalam jumlah besar, data dapat disimpan dalam *database*, *data warehouse*, atau penyimpanan informasi lainnya. *Data mining* berkaitan dengan bidang ilmu–ilmu lain, seperti *database system*, *data warehousing*, statistik, *machine learning*, *information retrieval*, dan komputasi tingkat tinggi. Selain itu, *data mining* didukung oleh ilmu lain seperti *neural networ*k, pengenalan pola, *spatial data analysis*, *image database*, *signal processing* [HAN-06]. *Data mining* dapat didefinisikan sebagai proses menemukan pola-pola dalam data. Pola yang

ditemukan harus penuh arti dan pola tersebut memberikan keuntungan, biasanya keuntungan secara ekonomi [WIT-05].

Karakteristik *data mining* adalah sebagai berikut :

1. *Data mining* berhubungan dengan penemuan sesuatu yang tersembunyi dan pola data tertentu yang tidak diketahui sebelumnya.
2. *Data mining* biasa menggunakan data yang sangat besar. Biasanya data yang besar digunakan untuk membuat hasil lebih dipercaya.
3. *Data mining* berguna untuk membuat keputusan yang kritis, terutama dalam strategi [DAV-04].

Berdasarkan beberapa pengertian tersebut dapat ditarik kesimpulan bahwa *data mining* adalah suatu teknik menggali informasi berharga yang terpendam atau tersembunyi pada suatu koleksi data (*database*) yang sangat besar sehingga ditemukan suatu pola yang menarik yang sebelumnya tidak diketahui. Kata mining sendiri berarti usaha untuk mendapatkan sedikit barang berharga dari sejumlah besar material dasar. Karena itu *data mining* sebenarnya memiliki akar yang panjang dari bidang ilmu seperti kecerdasan buatan (*artificial intelligent*), *machine learning*, statistik dan *database*. Beberapa metode yang sering disebut-sebut dalam literatur *data mining* antara lain *clustering, classification, association rules mining, neural network, genetic algorithm* dan lain-lain [PRA-07].

2.3.2 Fungsionalitas *Data Mining*

Fungsionalitas dari *data mining* dan pola pencarian yang dapat ditemukan dalam *data mining* adalah sebagai berikut :

1. Deskripsi Kelas

Data dapat diasosiasikan dengan kelas, yang berguna untuk menggambarkan kelas secara individual dan konsep secara tepat.

2. Analisis *Association*

Analisis *association* adalah penemuan *association rules* yang menunjukkan nilai kondisi dari atribut yang terjadi secara bersama-sama dan terus-menerus dalam membentuk sekumpulan data.

3. **Klasifikasi dan Prediksi**

   Klasifikasi adalah proses untuk menemukan sekumpulan model yang menggambarkan dan membedakan kelas data, yang bertujuan agar model tersebut dapat digunakan untuk memprediksi kelas suatu obyek yang belum diketahui kelasnya. Klasifikasi dapat digunakan untuk memprediksi baik data nominal maupun data diskrit.

4. **Analisis *Cluster***

   Analisis *cluster* berbeda dengan klasifikasi. *Clustering* menganalisis obyek data dengan kelas yang belum diketahui, sedangkan klasifikasi digunakan menganalisis obyek data dari kelas yang telah diketahui.

5. **Analisis *Outlier***

   Outlier dapat dideteksi menggunakan tes yang bersifat statistik yang mengambil sebuah distribusi dan probablilitas model untuk data atau menggunakan ukuran jarak, dimana obyek yang jaraknya jauh dari *cluster* yang lain dianggap sebagai *outlier*.

6. **Analisis *Evolution***

   Data analisis *evolution* ini mencari model atau obyek yang memiliki kebiasaan berubah setiap waktu. Analisis ini berkaitan dengan *time-series*. Analisis *evolution* dapat berupa *characterization, discrimination, association, classification,* dan *clustering* [HAN-01].

Fungsi *data mining* adalah untuk menspesifikasikan pola yang harus ditemukan. Secara umum tugas *data mining* dapat diklasifikasikan dalam dua kategori yaitu deskriptif dan prediktif. Tugas *data mining* secara deskriptif adalah untuk mengklasifikasikan sifat umum suatu data di dalam *database*. Tugas *data mining* secara prediktif adalah untuk mengambil kesimpulan terhadap data dalam membuat prediksi [HAN-01].

2.3.3 Tahap - Tahap *Data Mining*

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

1. ***Data Selection***

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

2. ***Preprocessing* atau *Cleaning***

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

3. ***Transformation***

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

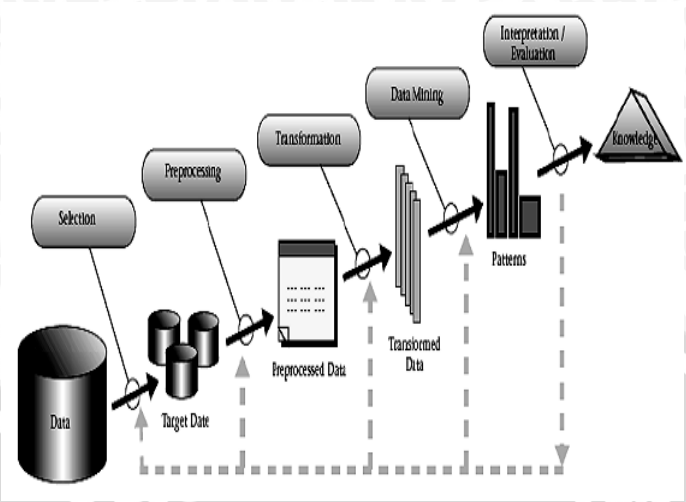
4. ***Data Mining***

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

5. ***Interpretation* atau *Evaluation***

Pola informasi yang dihasilkan dari proses *data mining* perlu ditampilkan dalam bentuk yang mudah dimengerti oleh pihak yang berkepentingan. Tahap ini mencakup pemeriksaan apakah pola atau informasi yang ditemukan bertentangan dengan fakta atau hipotesa yang ada sebelumnya [KUS-07].

Pada Gambar 2.2 diperlihatkan *data mining* sebagai tahapan dalam proses *knowledge discovery in database* (KDD) [HAN-06].



Gambar 2.2 *Data Mining* sebagai Tahapan dalam Proses KDD

2.3.4 Teknik *Data Mining*

[PRA-03] menyatakan bahwa teknik-teknik *data mining* mempunyai teknik yang paling populer antara lain:

a. *Association Rule Mining*

Association Rule Mining adalah teknik *data mining* untuk menemukan aturan asosiasi antara suatu kombinasi item.

b. *Classification*

Classification adalah proses untuk menemukan model atau suatu fungsi yang menjelaskan atau membedakan konsep atau kelas data dengan tujuan untuk dapat memperkirakan kelas dari suatu obyek yang labelnya tidak diketahui.

c. *Clustering*

Clustering adalah proses pengelompokkan data tanpa berdasarkan kelas data tertentu. *Clustering* dapat dipakai untuk memberikan label pada kelas data yang belum diketahui.

2.4 *Association Rule*

2.4.1 Pengertian *Association Rule*

Association rule (aturan asosiatif) adalah salah satu teknik utama dalam *data mining* dan merupakan bentuk yang paling umum dipakai dalam menemukan *pattern* atau pola dari suatu kumpulan data [KAN-03]. *Association rule* (aturan

asosiatif) berusaha menemukan aturan-aturan tertentu yang mengasosiasikan data yang satu dengan data yang lain. Untuk mencari *association rule* dari suatu kumpulan data, pertama-tama harus mencari lebih dulu yang disebut "*frequent itemset*" (sekumpulan item yang sering muncul bersamaan). Setelah semua pola *frequent itemset* ditemukan, barulah dicari aturan asosiatif yang memenuhi syarat yang telah ditentukan [PRA-06].

Association rule adalah suatu prosedur untuk mencari hubungan antar item dalam suatu kumpulan data yang ditentukan. Dalam menentukan suatu *association rule*, terdapat suatu ukuran kepercayaan yang didapatkan dari hasil pengolahan data dengan perhitungan tertentu. *Association rule* memberikan informasi dalam bentuk "*if – then"* atau "jika – maka". Biasanya digunakan istilah *antecedent* untuk mewakili bagian "jika" dan *consequent* untuk mewakili bagian "maka" [SAN-07].

Sebuah *association rule* dengan *confidence* sama atau lebih dari *minimum confidence*, dapat dikatakan sebagai *valid association rule* [RUL-08]. *Rule* yang memenuhi baik *minimum support* maupun *minimum confidence* disebut juga *strong rule* [HAN-04].

2.4.2 *Support*

Support adalah suatu ukuran yang menunjukkan seberapa besar tingkat dominasi suatu *itemset* dari keseluruhan transaksi [HAN-01]. *Support* untuk aturan "X → Y" adalah probabilitas atribut atau kumpulan atribut X dan Y yang terjadi bersamaan dalam suatu transaksi [YUL-02].

Nilai *support* untuk suatu item dapat diperoleh dengan rumus pada Persamaan 2.1, sedangkan nilai *support* untuk kombinasi item *antecedent* A dan *consequent* B dapat diperoleh dengan rumus pada Persamaan 2.2 [ROC-10].

$$Support\ (A) = \frac{\text{Jumlah Transaksi yang Mengandung A}}{\text{Total Transaksi}} \qquad (2.1)$$

$$Support\ (A, C) = P(A \cap C) \quad (2.2)$$
$$= \frac{\text{Jumlah Transaksi yang Mengandung A dan C}}{\text{Total Transaksi}}$$

Nilai *support* digunakan untuk menentukan *frequent itemset*. *Itemset* yang nilai *support*-nya memenuhi parameter minimum *support* (min_sup) masuk dalam *frequent itemset*. *Minimum support* merupakan parameter yang digunakan sebagai batasan frekuensi kejadian atau jumlah *support* yang harus dipenuhi suatu kelompok data yang dijadikan aturan [YUL-02].

2.4.3 *Confidence*

Confidence adalah suatu ukuran yang menunjukkan hubungan antara dua atau lebih item secara *conditional* [HAN-01]. *Confidence* atau tingkat kepercayaan merupakan probabilitas kejadian beberapa produk yang dibeli secara bersamaan dimana salah satu produk sudah pasti dibeli, misalnya jika terdapat n transaksi dimana X yang dibeli dan m transaksi dimana X dan Y yang dibeli bersamaan, maka *confidence* dari aturan X → Y adalah m/n [YUL-02].

Nilai *confidence* dapat diperoleh dengan rumus pada Persamaan 2.3 dan dijabarkan lebih khusus pada Persamaan 2.4 [ROC-10].

$$Confidence\ (A \rightarrow B) = \frac{\text{Support A} \cap \text{B}}{\text{Support A}} \quad (2.3)$$

$$Confidence\ (A \rightarrow B) = P(B|A)$$
$$= \frac{\text{Jumlah Transaksi yang Mengandung A dan B}}{\text{Jumlah Transaksi yang Mengandung A}} \quad (2.4)$$

Nilai *confidence* digunakan dalam menentukan *strong association rule*. *Association rule* yang nilai *confidence*-nya memenuhi parameter *threshold minimum confidence* (min_conf) termasuk dalam *strong association rule. Minimum confidence* adalah parameter yang mendefinisikan *minimum level* dari *confidence* yang dipenuhi oleh aturan yang berkualitas [YUL-02].

2.5 *FP Tree*

Menurut [SAM-08] bahwa *FP Tree* merupakan struktur penyimpanan data yang dimampatkan. *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 pemindaian data transaksi yang terbukti sangat efisien.

Algoritma *FP Growth* mencari himpunan *item* yang sering muncul dengan cara membangun *FP Tree* secara rekursif dan menggabungkan *frequent itemset* yang ada pada *conditional FP Tree* secara berturut – turut. *FP Tree* adalah sebuah *tree* yang terdiri dari satu *header* tabel, satu *root* yang diberi label "null", dan satu himpunan *item prefix subtree* sebagai node dari anak *root*. Masing –masing *entry* dalam *header table* adalah *frequent item*, dan setiap *record* terdiri dari dua atribut yaitu nama *item* dan *head of node-link*. Sedangkan setiap node anak terdiri dari atribut : nama *item*, *count*, dan *node-link*. *Node-link* menghubungkan node ke node berikutnya pada *tree* tersebut yang mempunyai nama *item* yang sama atau *null* jika tidak ada [FEB-09].

Jika *tree* mempunyai lebih dari satu cabang maka satu *frequent pattern* dibangkitkan dan juga dibangun *conditional FP Tree* untuk *frequent pattern* tersebut. Proses pembangunan *conditional FP Tree* sama seperti proses pembangunan *FP Tree*. *Conditional Fptree* dibangun dari pola dasar *conditional* (*conditional pattern base*). *Conditional pattern base* i adalah himpunan *prefix path* dari node i yang diperoleh dari penelusuran *node-link* [FEB-09].

Langkah awal dari algoritma pembuatan *FP Tree* adalah dilakukan pembacaan basis data terlebih dahulu dan membuat kumpulan dari *frequent 1-itemset* beserta nilai *support* kemudian diurutkan secara *descending* berdasarkan nilai *support*. Pembuatan *FP Tree* dimulai dengan membuat *root* dan diberi nama *null*. Kemudian item dari tiap transaksi diurutkan sesuai dengan urutan *frequent 1-itemset* dan dibuat cabang pada *FP Tree* untuk setiap transaksi. Setiap cabang baru akan ditambahkan ke dalam *FP Tree* maka dilakukan penelusuran apakah

node sudah ada. Jika node sudah ada maka dilakukan penambahan nilai *support* sebanyak 1. Dalam proses pembuatan *FP Tree* tidak hanya dilakukan pembuatan struktur pohon tetapi juga dilakukan pembuatan tabel untuk melacak dimana setiap item ditempatkan dalam *FP Tree*. Setiap item yang sama pada cabang yang berbeda dihubungkan dengan *link* yang dinamakan *nodelink*. *Pseudocode* dari algoritma pembentukan *FP Tree* dapat dilihat pada *Pseudocode* 2.1 [YIA-06].

```
Masukan : Basis data Transaksi(D), dan nilai minimum
support(minsup)
Keluaran: FP-tree Tree;
Fungsi : FP-tree dibentuk dengan cara berikut
(1)Transaksi di database (D) ditelusuri sekali agar didapat
   himpunan frequent item(F) dan nilai supportnya.
(2)Lalu F diurutkan mulai dari yang mempunyai nilai support
   yang terbesar sampai terkecil.
(3)Buat root dari FP-tree(T) dan diberi nama null.
(4)Untuk tiap transaksi Trans di D lakukan {
(5)Pilih dan dilakukan pengurutan frequent item dalam
   Trans(disebut [p|P]) mengacu pada F dimana p adalah elemen
   pertama dan P adalah item berikutnya.
(6)Panggil insert_tree([p|P],T);
(7)Fungsi insert_tree([p|P],T) terdiri dari langkah-langkah
   berikut :
(8)inisialisasi N = anak dari T
(9)inisialisasi nama_item = nama dari item yang terdapat pada T
(10)Jika (N.nama_item == p.nama_item){N.count ++;}
(11)Jika tidak {
   buat node baru N;N.count = 1;N.parent_link =T;
   N_link = node yang sama nama itemnya dihubungkan oleh
   struktur node_link}
(12)Jika P tidak kosong{insert_tree(P,N)}
```

***Pseudocode* 2.1** Pembentukan *FP Tree* [YIA-06]

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* [SAM-08].

Misalkan diberikan tabel data transaksi sebagai berikut, dengan minimum *support count* ξ=2 yang dapat dilihat pada Tabel 2.1.

Tabel 2.1 Contoh Data Transaksi Mentah [SAM-08]

No	Transaksi
1	a,b
2	b,c,d,g,h
3	a,c,d,e,f
4	a,d,e
5	a,b,z,c
6	a,b,c,d
7	a,r
8	a,b,c
9	a,b,d
10	b,c,e

Frekuensi kemunculan tiap item dapat dilihat pada tabel berikut:

Tabel 2.2 Frekuensi Kemunculan Tiap Karakter [SAM-08]

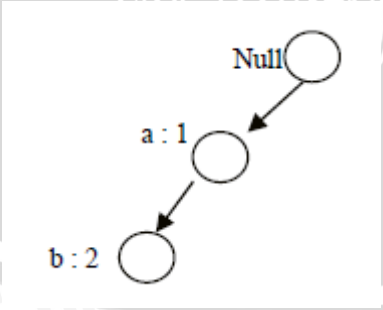
Item	Frekuensi
A	8
B	7
C	6
D	5
E	3
F	1
R	1
Z	1
G	1
H	1

Setelah dilakukan pemindaian pertama didapat *item* yang memiliki frekuensi di atas *support count* ξ=2 adalah a,b,c,d, dan e. Kelima *item* inilah yang akan berpengaruh dan akan dimasukkan ke dalam *FP Tree*, selebihnya (f,r,z,g, dan h) dapat dibuang karena tidak berpengaruh signifikan. Pada Tabel 2.3 berikut mendata kemunculan item yang *frequent* dalam setiap transaksi, diurut berdasarkan yang frekuensinya paling tinggi [SAM-08].

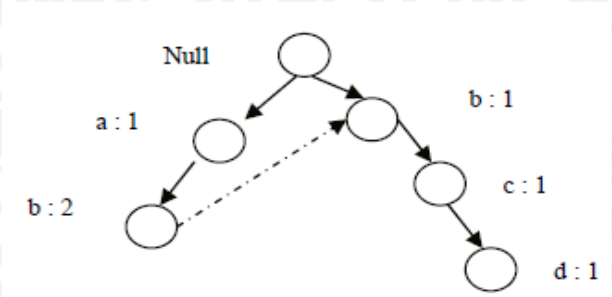
Tabel 2.3 Data Transaksi [SAM-08]

TID	Item
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

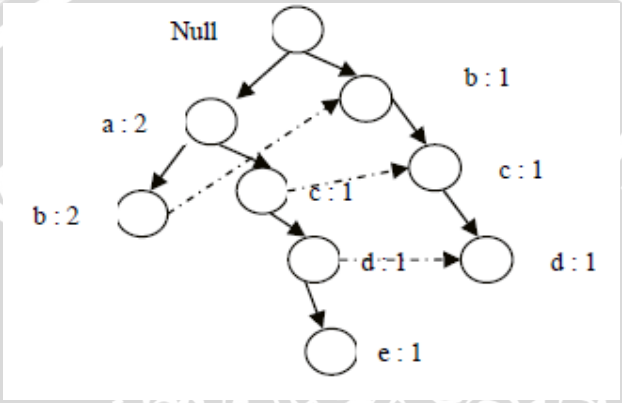
Berdasarkan Tabel 2.3 yaitu berupa data transaksi, kemudian dilakukan pembacaan terhadap TID yang dimulai dari pembacaan TID 1 yang digambarkan pada Gambar 2.3.



Gambar 2.3 Hasil Pembentukan *FP Tree* Setelah Pembacaan TID 1

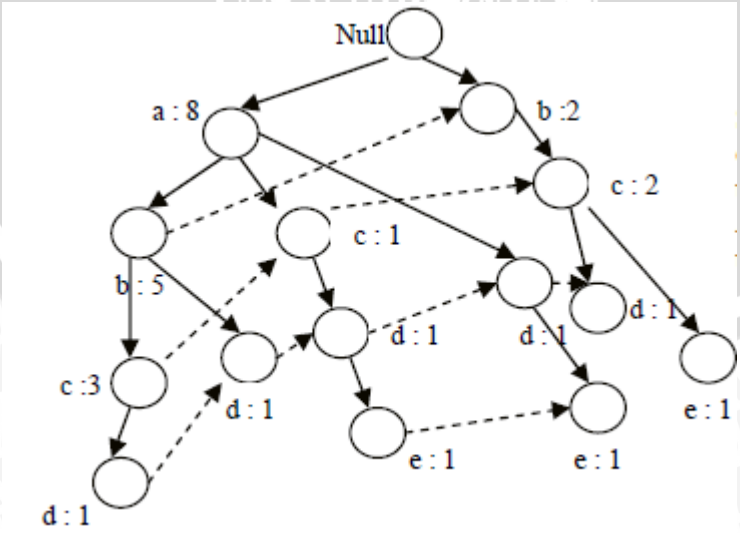


Gambar 2.4 Hasil Pembentukan *FP Tree* Setelah Pembacaan TID 2



Gambar 2.5 Hasil Pembentukan *FP Tree* Setelah Pembacaan TID 3

Setelah melakukan pembacaan TID 1 dan membentuk *FP Tree*, dilanjutkan pembacaan hingga TID 10 sehingga didapatkan pembentukan *FP Tree* yang telah lengkap berdasarkan 10 data transaksi yang ada. Pembentukan *FP Tree* yang didapatkan setelah pembacaan TID 10 digambarkan pada Gambar 2.6.



Gambar 2.6 Hasil Pembentukan *FP Tree* Setelah Pembacaan TID 10

Diberikan 10 data transaksi dengan 5 jenis item dengan frekuensi tertinggi seperti pada Tabel 2.3 digunakan untuk menunjukkan terbentuknya *FP Tree* setiap TID dibaca. Setiap simpul atau *node* pada *FP Tree* mengandung nama sebuah item dan *counter support* yang berfungsi untuk menghitung frekuensi kemunculan item tersebut dalam lintasan transaksi [SAM-08].

FP-tree yang merepresentasikan data transaksi pada Tabel 2.1 dibentuk dengan cara sebagai berikut:

1. Kumpulan data dipindai pertama kali untuk menentukan *support count* dari setiap *item*. Item yang tidak *frequent* dibuang, sedangkan *frequent item* dimasukkan dan disusun dengan urutan menurun, seperti yang terlihat pada Tabel 2.1.
2. Pemindaian kedua, yaitu pembacaan TID pertama {a,b} akan membuat simpul a dan b, sehingga terbentuk lintasan transaksi Null→a→b. *Support count* dari setiap simpul bernilai awal 1 yang digambarkan pada Gambar 2.3.
3. Setelah pembacaan transaksi kedua {b,c,d}, terbentuk lintasan kedua yaitu Null→b→c→d. *Support count* masing-masing count juga bernilai awal 1. Walaupun b ada pada transaksi pertama, namun karena *prefix* transaksinya tidak sama, maka transaksi kedua ini tidak bisa dimampatkan dalam satu lintasan yang digambarkan pada Gambar 2.4.
4. Transaksi keempat memiliki *prefix* transaksi yang sama dengan transaksi pertama, yaitu a, maka lintasan transaksi ketiga dapat ditimpakan di a, sambil menambah *support count* dari a, dan selanjutnya membuat lintasan baru sesuai dengan transaksi ketiga yang digambarkan pada Gambar 2.5.
5. Proses ini dilanjutkan sampai *FP Tree* berhasil dibangun berdasarkan tabel data transaksi yang diberikan.

2.6 Algoritma *FP Growth*

Algoritma *FP-Growth* merupakan pengembangan dari algoritma *Apriori*. Sehingga kekurangan dari algoritma *Apriori* diperbaiki oleh algoritma *FP-Growth. Frequent Pattern Growth* (*FP-Growth*) adalah salah satu alternatif algoritma yang dapat digunakan untuk menentukan himpunan data yang paling sering muncul (*frequent itemset*) dalam sebuah kumpulan data [SAM-08].

Pada algoritma *Apriori* diperlukan *generate candidate* untuk mendapatkan *frequent itemsets*. Akan tetapi, di algoritma *FP-Growth generate candidate* tidak dilakukan karena *FP-Growth* menggunakan konsep pembangunan *tree* dalam pencarian *frequent itemsets*. Hal tersebutlah yang menyebabkan algoritma *FP-Growth* lebih cepat dari algoritma *Apriori*. Karakteristik algoritma *FP-Growth* adalah struktur data yang digunakan adalah *tree* yang disebut dengan *FP-Tree*. Dengan menggunakan *FP-Tree,* algoritma *FP-growth* dapat langsung mengekstrak *frequent Itemset* dari *FP-Tree* [ERW-09].

Penggalian itemset yang *frequent* dengan menggunakan algoritma *FP-Growth* akan dilakukan dengan cara membangkitkan struktur data *tree* atau disebut dengan *FPTree*. Metode *FP-Growth* dapat dibagi menjadi 3 tahapan utama yaitu sebagai berikut [HAN-06]:

1. Tahap pembangkitan *conditional pattern base*

   Conditional Pattern Base merupakan subdatabase 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*

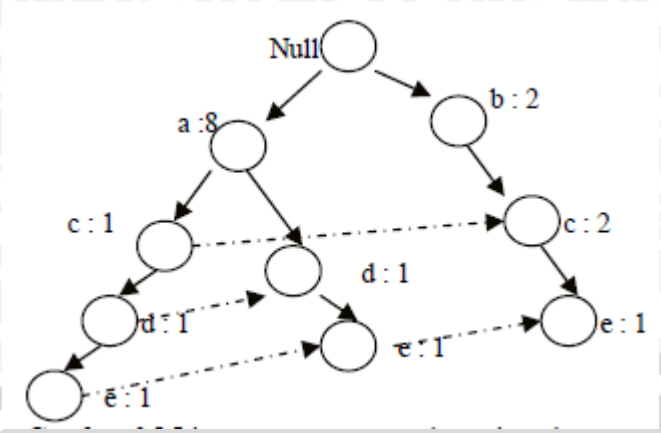
   Apabila *Conditional FP Tree* merupakan lintasan tunggal (*single path),* maka didapatkan *frequent itemset* dengan melakukan kombinasi item untuk setiap *conditional FP Tree*. Jika bukan lintasan tunggal, maka dilakukan pembangkitan *FP Growth* secara rekursif.

Ketiga tahap tersebut merupakan langkah yang akan dilakukan untuk mendapat frequent itemset, yang dapat dilihat pada algoritma berikut :

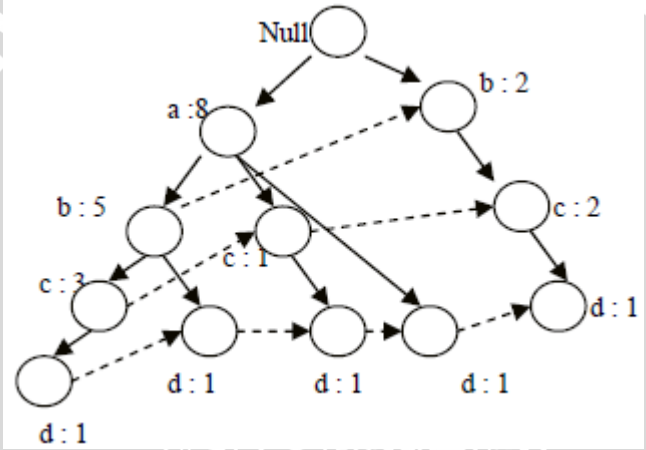
```
Input : FP-Tree Tree
Output : Rt sekumpulan lengkap pola
frequent
Method : FP-growth (Tree, null)
Procedure : FP-growth (Tree, α)
{
01: if Tree mengandung single path P;
02: then untuk tiap kombinasi (dinotasikan β)
dari node-node dalam path do
03: bangkitkan pola β α dengan support dari
node-node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

***Pseudocode* 2.2** Algoritma *FP Growth* [HAN-06]

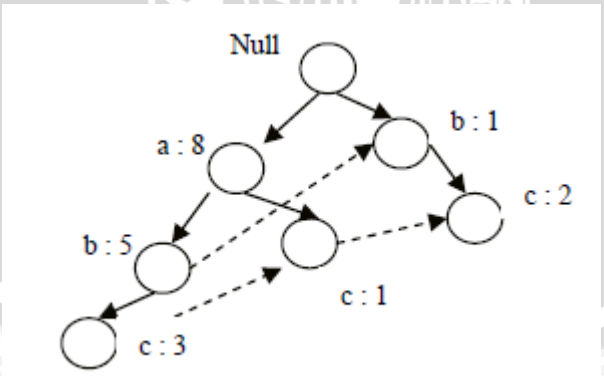
Untuk uji coba yang menerapkan *FP Growth* akan digunakan contoh kasus yang telah dilakukan pembentukan *FP Tree* sebelumnya. Untuk menemukan *frequent itemset* dari contoh 1, maka perlu ditentukan pohon dengan lintasan *y*ang berakhir dengan *support count* terkecil, yaitu e. Berturut-turut ditentukan juga yang berakhir di d,c,b, dan a. Proses pembentukan dapat dilihat pada gambar berikut [SAM-08] :



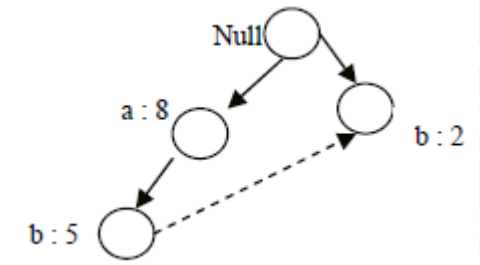
Gambar 2.7 Lintasan yang Mengandung Simpul e



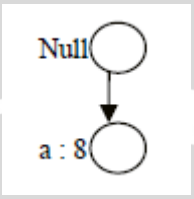
Gambar 2.8 Lintasan yang Mengandung Simpul d



Gambar 2.9 Lintasan yang Mengandung Simpul c

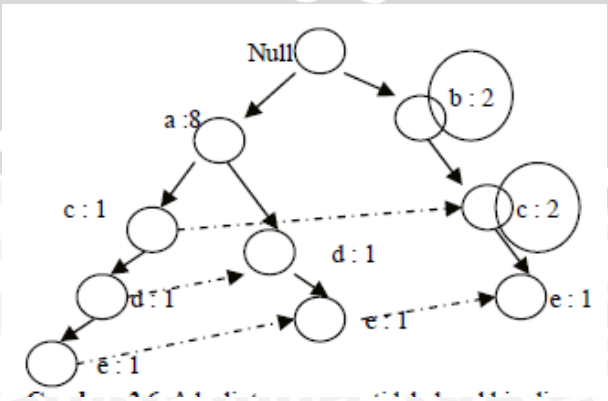


Gambar 2.10 Lintasan yang Mengandung Simpul b



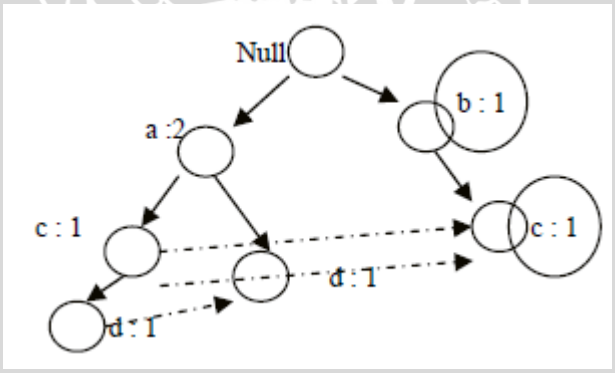
Gambar 2.11 Lintasan yang Mengandung Simpul a

Algoritma FP-growth menemukan *frequent itemset* yang berakhiran *suffix* tertentu dengan menggunakan metode *divide and conquer* untuk memecah problem menjadi subproblem yang lebih kecil. Contohnya, jika kita ingin menemukan semua *frequent itemset* yang berakhiran e. Oleh karena itu, kita harus mengecek apakah *support count* dari e memenuhi minimum *support count* ξ=2. Karena *support count* dari e adalah 3, dan $3 \geq \xi$, maka e adalah item yang *frequent.* Setelah mengetahui bahwa *item* e adalah *item* yang frequent, maka subproblem selanjutnya adalah menemukan *frequent itemset* dengan akhiran de, ce, be, dan ae. Dengan menggabungkan seluruh solusi dari subproblem yang ada, maka himpunan semua *frequent itemset* yang berakhiran *item* e akan didapatkan. Untuk lebih memperjelas, dapat dilihat contoh menemukan *frequent itemset* yang berakhiran dengan *item* e dapat dilihat pada Gambar 2.12 [SAM-08].



Gambar 2.12 Lintasan yang Tidak Berakhir di e yaitu Null → b → c

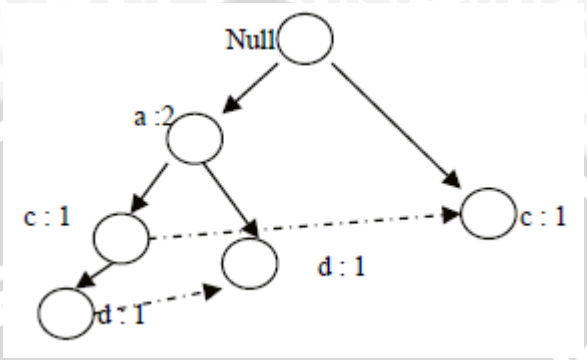
1. Langkah pertama yang dilakukan adalah membangun sebuah pohon *FP Tree* dengan hanya menyertakan lintasan yang berakhir di e.
2. *Support count* dari *item* e dihitung dan dibandingkan dengan minimum *support count ξ=3*. Karena memenuhi, maka {e} termasuk *frequent itemset*, karena *support count=2.*
3. Karena *item* e *frequent,* maka perlu dipecahkan subproblem untuk menemukan *frequent itemset* yang berkahiran dengan de, ce, be, dan ae. Sebelum memecahkan subproblem ini, maka upapohon *FP Tree* tersebut harus diubah terlebih dahulu menjadi *conditional FP Tree*. *Conditional FP Tree* mirip dengan *FP Tree* biasa, namun *conditional FP Tree* dimaksudkan untuk mencari *frequent itemset* yang berakhiran *item* tertentu.
4. *Conditional FP Tree* dapat dibentuk dengan cara :
    a. Setiap lintasan yang tidak mengandung e dibuang. Pada contoh, lintasan terkanan, terdapat lintasan yang tidak mengandung e, yaitu null→b→c. Lintasan ini dapat dibuang dengan cara mengurangi *support count* menjadi 1, sehingga lintasan tersebut hanya mengandung transaksi {b,c,e}, seperti pada Gambar 2.13.



Gambar 2.13 Simpul e Dibuang, *Support Count* Simpul yang Diperbarui

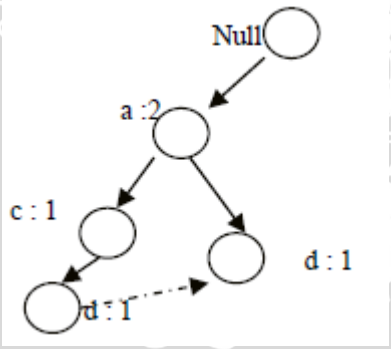
    b. Setelah semua lintasan berakhir di e, maka simpul e dapat dibuang, karena setiap nilai *support count* pada simpul orang tuanya telah mencerminkan transaksi yang berakhir di e. Subproblem selanjutnya yang harus dipecahkan adalah mencari lintasan *frequent itemset* yang berakhir di de, ce, be, dan ae.
    c. Karena nilai *support count* dari b adalah 1, yang berarti transaksi yang mengandung b dan e hanya 1 transaksi, maka berdasarkan prinsip

anti-monotone heuristic, simpul b dan lintasan yang mengandung be dapat dibuang, karena jika *item* b tidak frequent, maka setiap transaksi yang berakhiran be juga tidak *frequent.* Terbentuk *Conditional FP Tree* untuk e, seperti pada Gambar 2.14.



Gambar 2.14 *Conditional FP Tree* untuk e

5. *FP Tree* menggunakan *Conditional FP Tree* untuk membangun pohon lintasan prefix untuk menemukan *frequent itemset* yang berakhir dengan pasangan *item* de, ce, dan ae.
6. Untuk lintasan *prefix* de, yang dibentuk dari *Conditional FP Tree* untuk *item* e dapat dilihat pada Gambar 2.15.



Gambar 2.15 Pohon *Prefix* yang Berakhir di de

7. Dengan menjumlahkan *support count* dari d, yang tidak lain adalah jumlah *frequent itemset* yang berakhir di de, didapat bahwa {d,e} juga termasuk dalam *frequent itemset.*
8. Selanjutnya algoritma akan mengulangi langkah yang sama dengan langkah ketiga, sehingga didapatkan *conditional FP Tree untuk de* hanya berisi satu daun, yaitu a, dengan *support count* 2. Sehingga {a,d,e} termasuk dalam *frequent itemset.*

9. Subproblem berikutnya yaitu dengan menemukan *frequent itemset* yang berakhiran dengan ce. Didapat {c,e} juga merupakan *frequent itemset.* Begitupula dengan {a,e}.

Setelah memeriksa *frequent itemset* untuk beberapa akhiran (*suffix*), maka didapat hasil yang dirangkum dalam Tabel 2.4.

Tabel 2.4 Hasil Pencarian *Frequent Itemset* [SAM-08]

Suffix	***Frequent Itemset***
E	{e},{d,e},{a,d,e},{c,e},{a,e}
D	{d},{c,d},{b,c,d},{a,c,d},{b,d},{a,b,d},{a,d}
C	{c},{b,c},{a,b,c},{a.c}
B	{b},{a,b}
A	{a}

Dengan metode *divide and conquer* ini, maka pada setiap langkah rekursif, algoritma *FP Growth* akan membangun sebuah *conditional FP Tree* baru yang telah diperbaharui nilai *support count*, dan membuang lintasan yang mengandung item-item yang tidak *frequent* lagi [SAM-08].

2.7 *Lift Ratio*

Lift ratio digunakan untuk mengevaluasi kuat tidaknya sebuah aturan asosiasi. *Lift ratio* adalah perbandingan antara *confidence* sebuah aturan adalah nilai *benchmark confidence*. *Benchmark confidence* adalah perbandingan antara jumlah semua *item consequent* terhadap total jumlah transaksi. Rumus *benchmark confidence* dan *lift ratio* dapat dilihat pada persamaan 2.5 dan persamaan 2.6 [SAN-07].

$$Benchmark\ Confidence = \frac{Nc}{N} \qquad (2.5)$$

Keterangan persamaan 2.5 :

Nc = jumlah transaksi dengan item dalam *consequent*

N = jumlah transaksi *database*

$$Lift\ Ratio = \frac{Confidence\ (A, C)}{Benchmark\ \ Confidence\ (A, C)} \qquad (2.6)$$

Apabila nilai *lift ratio* lebih besar dari 1, maka menunjukkan adanya manfaat dari aturan tersebut. Lebih tinggi nilai *lift ratio* maka lebih besar kekuatan asosiasinya [SAN-07].

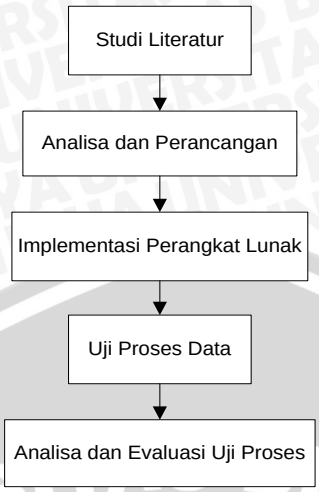
BAB III
METODE PENELITIAN DAN PERANCANGAN

3.1 Metode Penelitian

Pada subbab ini akan dibahas metode atau langkah-langkah yang digunakan dalam penelitian. Berikut ini merupakan langkah-langkah yang dilakukan dalam penelitian :

1. Mempelajari teori-teori dari literatur dan artikel yang berhubungan dengan penelitian. Mempelajari teori tentang *association rule*, algoritma *FP Growth*, *lift ratio* serta mempelajari bagaimana teknis untuk mengimplementasikan metode tersebut dari buku, jurnal, artikel dan dari penelitian-peneilitian sebelumnya.
2. Melakukan analisis dan membuat rancangan model perangkat lunak dengan mengimplementasikan metode yang akan digunakan. Setelah mempelajari *association rule*, algoritma *FP Growth*, *lift ratio* dan cara mengimplementasikannya, melakukan analisis apa saja masukan yang diperlukan dan bagaimana keluaran yang akan ditampilkan. Dari analisis tersebut dibentuk diagram dan gambaran antarmuka yang akan ditampilkan.
3. Membuat perangkat lunak berdasarkan analisis dan perancangan yang telah dilakukan. Setelah membuat diagram dan gambaran antarmuka mulai membuat perangkat lunak berdasarkan rancangan diagram dan antarmuka yang telah dibuat tersebut.
4. Melakukan implementasi algoritma dengan perangkat lunak yang telah dibuat. Setelah proses pembuatan perangkat lunak selesai dilakukan proses pengujian terkait dengan kelayakan perangkat lunak dan implementasi dari *association rule* dan algoritma *FP Growth.*
5. Melakukan analisis dan evaluasi hasil implementasi algoritma. Setelah dilakukan pengujian dilakukan analisis terhadap hasil dari pengujian sehingga dapat ditarik kesimpulan berdasarkan hasil tersebut.

Langkah-langkah penelitian yang dilakukan akan ditunjukkan pada Gambar 3.1 sebagai berikut :



Gambar 3.1 Langkah-Langkah Penelitian

3.2 Perancangan

3.2.1. Perancangan Pembuatan Data Penangkapan

Pengumpulan data yang diperoleh dari kantor Dinas Perikanan dan Kelautan Provinsi Jawa Timur adalah data nama daerah penangkapan dan nama jenis ikan. Data nama daerah terdiri dari 23 kota/kabupaten dan data nama jenis ikan terdiri dari 44 jenis. Pada penelitian ini digunakan data nama daerah, data nama jenis ikan dan data penangkapan. Untuk mendapatkan data waktu penangkapan (kuartal) dan data jumlah ikan (ton), maka dilakukan pembuatan data penangkapan secara random. Dalam pembuatan data penangkapan terdapat batasan aturan yang dibuat untuk mendapatkan data simulasi, yaitu sebagai berikut :

1. Pembuatan data jumlah ikan dibuat dalam *range* antara 100 sampai dengan 1.000 dengan satuan ton.
2. Untuk penyeleksian data jumlah ikan yang akan digunakan dalam data transaksi adalah dengan cara membatasi jumlah ikannya. Jika jumlah jenis ikan lebih besar dari 700 ton, maka jenis ikan tersebut akan digunakan dalam pengolahan data. Namun jika kurang dari 700 ton, maka jenis ikan tersebut tidak digunakan.

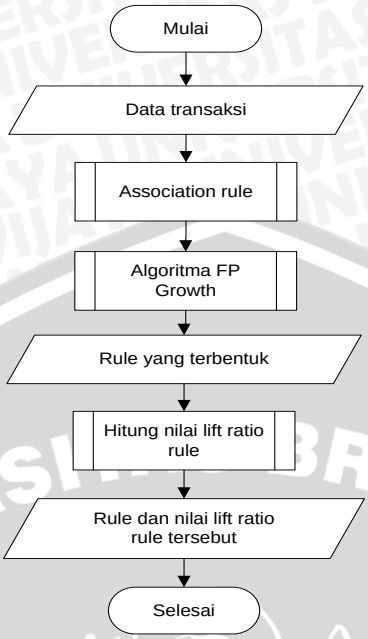
3.2.2. Analisis Kebutuhan Perangkat Lunak

Sistem ini mempunyai dua proses utama yaitu proses pembuatan data penangkapan dan proses penggalian data. Parameter yang digunakan pada sistem ini antara lain nilai *minimum support*, nilai *minimum confidence*, dan parameter daerah pilihan. Dalam parameter daerah pilihan terdapat pilihan nama daerah di Jawa Timur yang akan diolah datanya. Data tersebut yang nantinya akan digunakan sebagai data pengujian oleh sistem untuk dicari pola datanya. Hasil keluaran dari sistem ini berupa pola data yang menunjukkan keterkaitan jenis ikan yang satu dengan jenis ikan yang lain.

Proses pembuatan data penangkapan dilakukan untuk menghasilkan data yang nantinya digunakan sebagai data uji coba pada sistem yang dibangun. Data penangkapan terdiri dari data nama daerah, nama jenis ikan, waktu penangkapan (kuartal) dan jumlah ikan (ton). Untuk menghasilkan data yang sesuai dengan kebutuhan penelitian, maka dibutuhkan suatu proses pembuatan data penangkapan. Data penangkapan adalah data yang dibuat secara random oleh sistem namum tetap mempunyai aturan batasan tertentu seperti yang telah dijelaskan pada subbab 3.2.1.

Proses penggalian data dilakukan setelah didapatkan data transaksi sesuai parameter daerah yang dipilih pengguna. Metode yang digunakan untuk proses penggalian data dalam sistem ini adalah metode *association rule* dengan algoritma *FP Growth* bertujuan untuk melakukan *frequent itemset mining.* Setelah melewati proses perhitungan dan pembentukan *tree*, langkah terakhir adalah mencari nilai *lift ratio* dari setiap *rule* yang dihasilkan. Secara umum proses penggalian data menggunakan metode *association rule* dengan algoritma *FP Growth* digambarkan pada Gambar 3.2 dengan deskripsi alur diagram sebagai berikut :

1. Memasukkan data transaksi ke sistem.
2. Melakukan penggalian data transaksi menggunakan metode *association rule* dengan algoritma *FP Growth* sehingga dihasilkan *rule*.
3. Mencari nilai *lift ratio* dari setiap *rule* yang terbentuk.



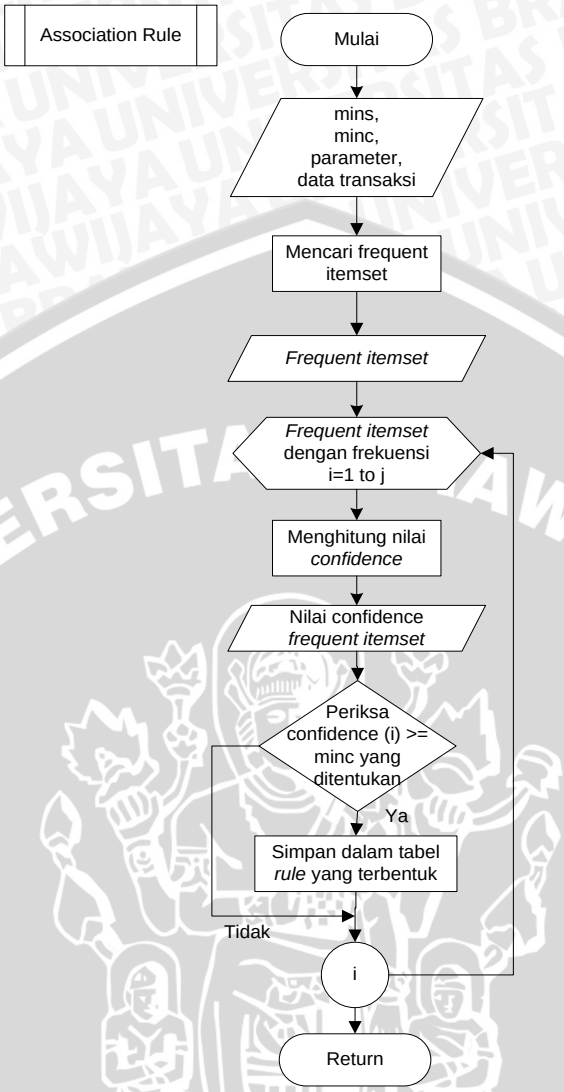
Gambar 3.2 Diagram Alir Proses Penggalian Data Secara Umum

3.2.3 Perancangan Perangkat Lunak

3.2.3.1 Tahap *Association Rule*

Proses pencarian *frequent itemset* pada data transaksi dilakukan dengan menggunakan metode *association rule* yang digambarkan pada Gambar 3.3 dengan deskripsi alur diagram sebagai berikut :

1. Memasukkan nilai *minimum support* (mins), *minimum confidence* (minc), dan parameter daerah pilihan.
2. Sistem menerima masukan data transaksi sesuai parameter daerah yang dipilih pengguna, kemudian sistem mencari *frequent itemset*.
3. Menghitung nilai *confidence* dari setiap *frequent itemset* dan dibandingkan dengan nilai *minimum confidence*. *Frequent itemset* yang memenuhi aturan jika nilai *confidence* lebih besar atau sama dengan nilai *minimum confidence*, maka akan disimpan. Tetapi jika tidak memenuhi aturan, maka *frequent item* tersebut akan diabaikan.

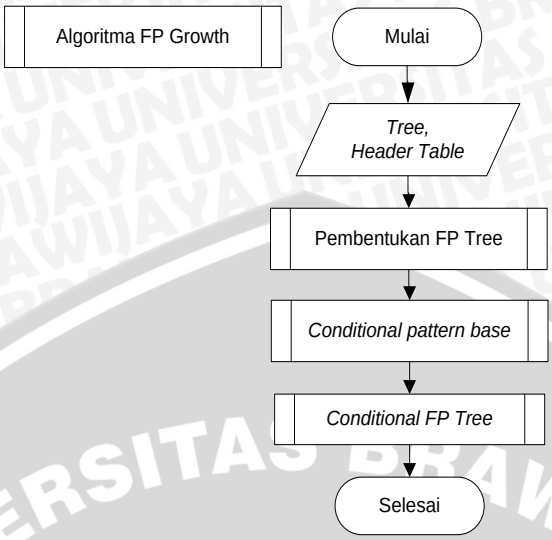


Gambar 3.3 Diagram Alir Proses *Association Rule*

3.2.3.2 Tahap Algoritma *FP Growth*

Proses algoritma *FP Growth* untuk mendapatkan *frequent itemset* akan digambarkan pada Gambar 3.4 dengan deskripsi alur diagram sebagai berikut :

1. Dengan masukan *tree* hasil pembentukan *FP Tree* dan *header* table, langkah pertama adalah pembentukan *FP Tree*.
2. Melakukan proses *conditional pattern base*.
3. Melakukan proses *conditional FP Tree*.

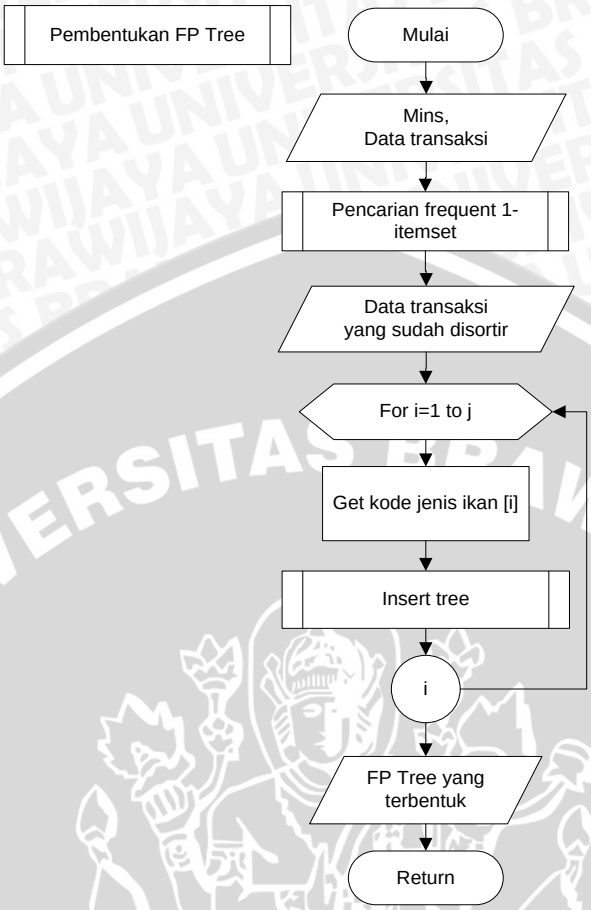


Gambar 3.4 Diagram Alir Algoritma *FP Growth*

3.2.3.3 Tahap Pembentukan *FP Tree*

Proses pembentukan *FP Tree* adalah bagian dari proses algoritma *FP Growth* untuk mendapatkan *frequent itemset* dengan membentuk *tree* dan lintasannya. Pembentukan *FP Tree* digambarkan pada Gambar 3.5 dengan deskripsi alur diagram sebagai berikut :

1. Sistem membaca masukan data transaksi dan *minimum support* (mins).
2. Mencari *frequent 1-itemset* dari data transaksi tersebut, kemudian data transaksi dilakukan penyortiran untuk kode jenis ikan yang *frequent* dan diurutkan nilai frekuensinya dari yang terbesar ke terkecil.
3. Melakukan proses *insert tree* dengan mengambil kode jenis ikan dari data transaksi yang sudah disortir tersebut dan hasilnya adalah terbentuknya *FP Tree*.

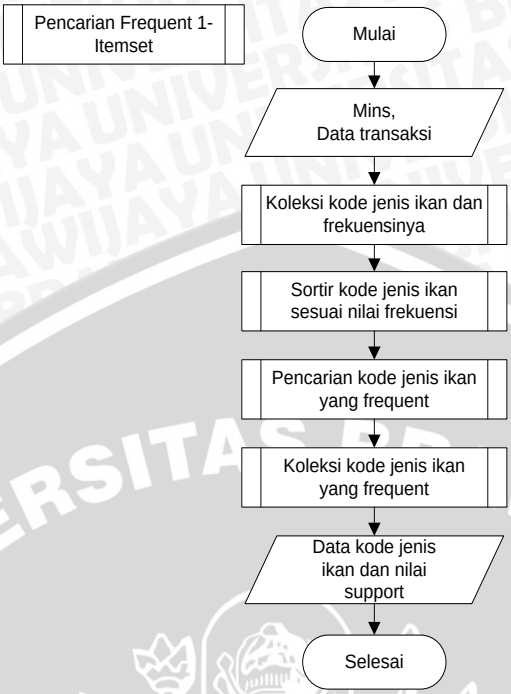


Gambar 3.5 Diagram Alir Pembentukan *FP Tree*

3.2.3.4 Tahap Pencarian *Frequent 1-Itemset*

Proses pencarian *frequent 1-itemset* dilakukan dengan tujuan untuk membentuk *FP Tree* yang digambarkan pada Gambar 3.6 dengan deskripsi alur diagram sebagai berikut :

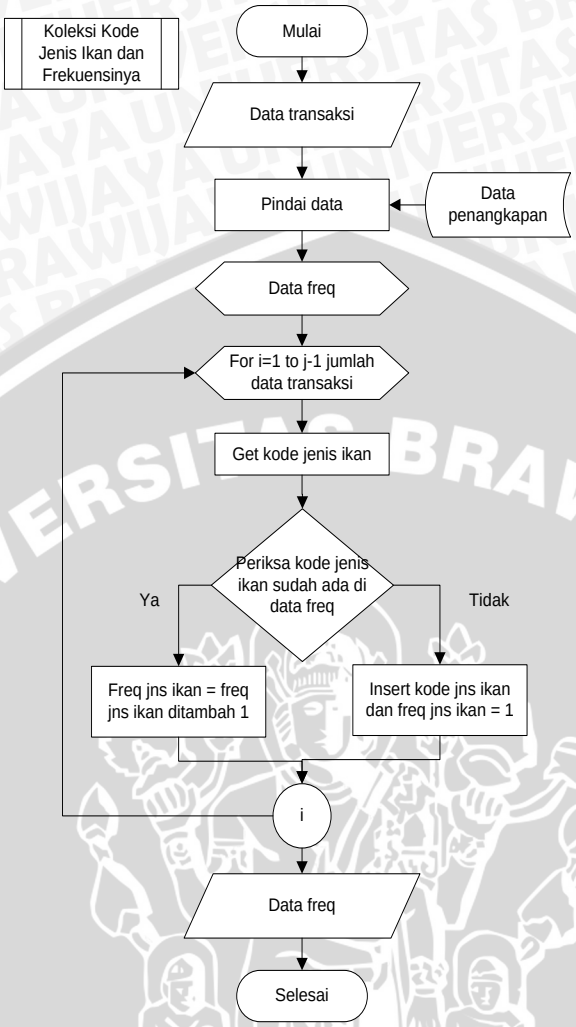
1. Sistem membaca nilai *minimum support* (mins) dan data transaksi, kemudian melakukan pengkoleksian kode jenis ikan beserta nilai frekuensinya.
2. Setelah dikoleksi, dari hasil tersebut dilanjutkan proses penyortiran kode jenis ikan berdasarkan nilai frekuensinya yang diurutkan dari yang terbesar ke terkecil.
3. Mencari kode jenis ikan yang *frequent* dan kemudian dikoleksi.



Gambar 3.6 Diagram Alir Pencarian *Frequent 1-Itemset*

Proses lanjutan dari pencarian *frequent 1-itemset* adalah mengkoleksi kode jenis ikan dan nilai frekuensinya, proses tersebut digambarkan pada Gambar 3.7 dengan deskripsi alur diagram sebagai berikut :

1. Membaca masukan data transaksi dan kemudian dilakukan pemindaian pada data penangkapan yang ada di *database*.
2. Memberikan inisialisasi data freq untuk menyimpan data frekuensi dari setiap kode jenis ikannya.
3. Mengambil kode jenis ikan pada setiap data transaksi, dengan aturan jika kode jenis ikan tersebut sudah ada pada data freq maka nilai frekuensi dari kode jenis ikan akan ditambahkan nilainya 1. Sedangkan jika belum ada pada data freq maka nilai frekuensi kode jenis ikan disimpan dalam data freq beserta nilai frekuensinya = 1.

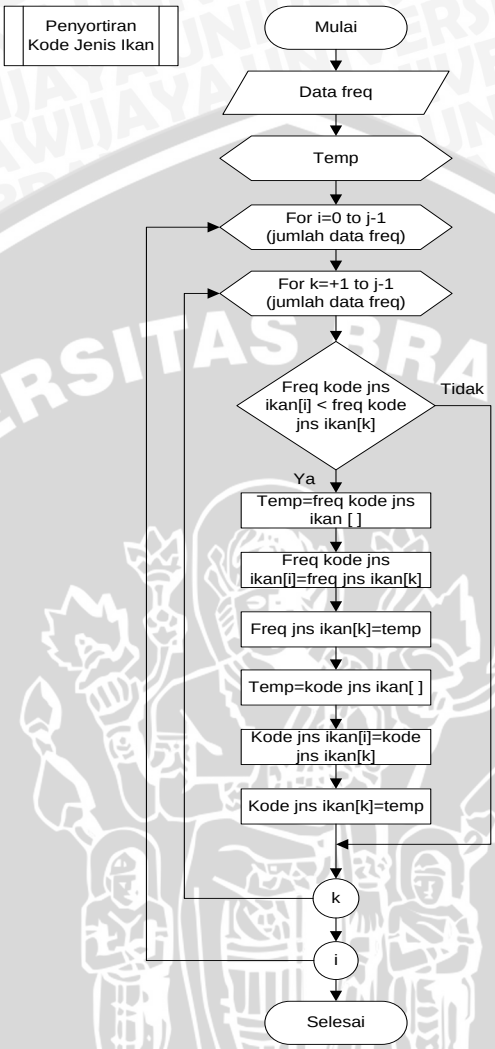


Gambar 3.7 Diagram Alir Pengkoleksian Kode Jenis Ikan dan Frekuensinya

Setelah melewati proses pengkoleksian kode jenis ikan dan frekuensinya, kemudian dilanjutkan penyortiran kode jenis ikan sesuai nilai frekuensinya. Data freq yang diperoleh dari tahap pengkoleksian kode jenis ikan sebelumnya, data tersebut akan digunakan untuk mendapatkan data kode jenis ikan yang sudah tersortir. Proses ini digambarkan pada Gambar 3.8 dengan deskripsi alur diagram sebagai berikut :

1. Sistem membaca data freq kemudian nilai frekuensi terbesar ke terkecil diurutkan secara *descending*.
2. Membaca setiap data freq dan membandingkan nilai frekuensinya dengan data freq yang lainnya. Jika nilai frekuensi kode jenis ikan tersebut lebih

kecil daripada nilai frekuensi kode jenis ikan yang dibandingkan, maka isi data freq akan dilakukan penukaran.

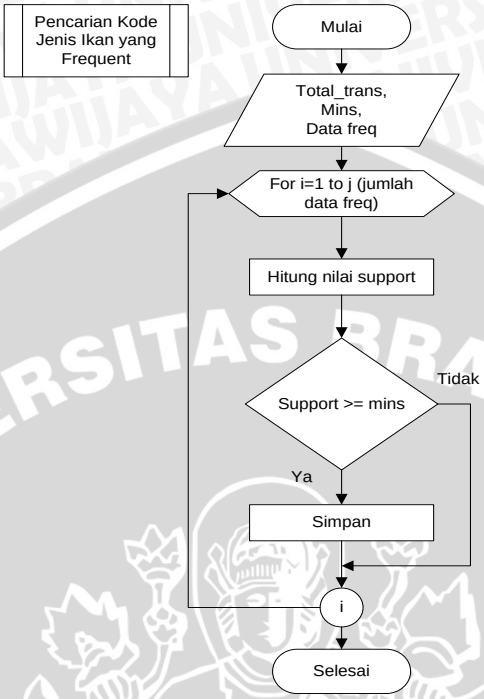


Gambar 3.8 Diagram Alir Kode Jenis Ikan Sesuai Frekuensinya yang Diurutkan

Proses selanjutnya adalah mencari kode jenis ikan yang *frequent* yang digambarkan pada Gambar 3.9 dengan deskripsi alur diagram sebagai berikut :

1. Sistem membaca total_trans, nilai mins, dan data freq, kemudian setiap nilai frekuensi dari kode jenis ikan yang terdapat di data freq untuk dihitung nilai *support*-nya.
2. Nilai *support* dari masing-masing kode jenis ikan dibandingkan dengan nilai *minimum support*. Terdapat aturan jika nilai *support* kode jenis ikan tersebut

lebih besar atau sama dengan nilai *minimum support*, maka data kode jenis ikan dan nilai *support*-nya akan disimpan sebagai *frequent 1-itemset*.

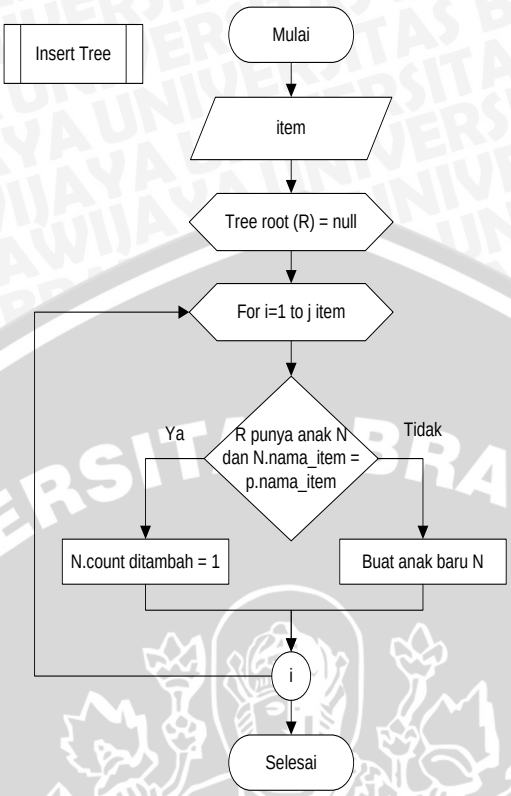


Gambar 3.9 Diagram Alir Pencarian Kode Jenis Ikan yang *Frequent*

3.2.3.5 Tahap *Insert Tree*

Insert tree juga merupakan bagian dari tahap pembentukan *FP Tree*, proses *insert tree* akan digambarkan pada Gambar 3.10 dengan deskripsi alur diaram sebagai berikut :

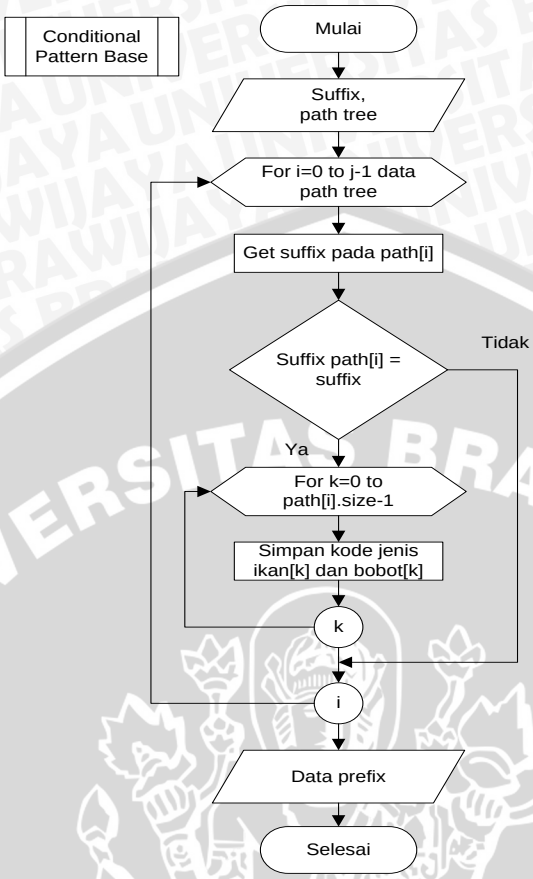
1. Sistem membaca masukan item dan melakukan inisialisasi *root* yaitu *root* = *null*.
2. Untuk setiap item yang dimasukkan akan dilakukan pengecekan *root*, untuk mengetahui apakah *root* tersebut mempunyai *node* (simpul) anak dengan nama_item yang sama dengan item yang dimasukkan tadi.
3. Terdapat aturan jika ada yang sama maka N.count akan ditambah nilainya sebanyak 1, namun jika tidak ada maka buat simpul anak baru N dengan N.nama_item sama dengan nama item tersebut yaitu p.nama_item. N.parent adalah *root*, N.count = 1, dan N.link adalah yang menghubungkan dengan simpul yang memiliki nama_item yang sama.



Gambar 3.10 Diagram Alir *Insert Tree*

3.2.3.6 Tahap *Conditional Pattern Base*

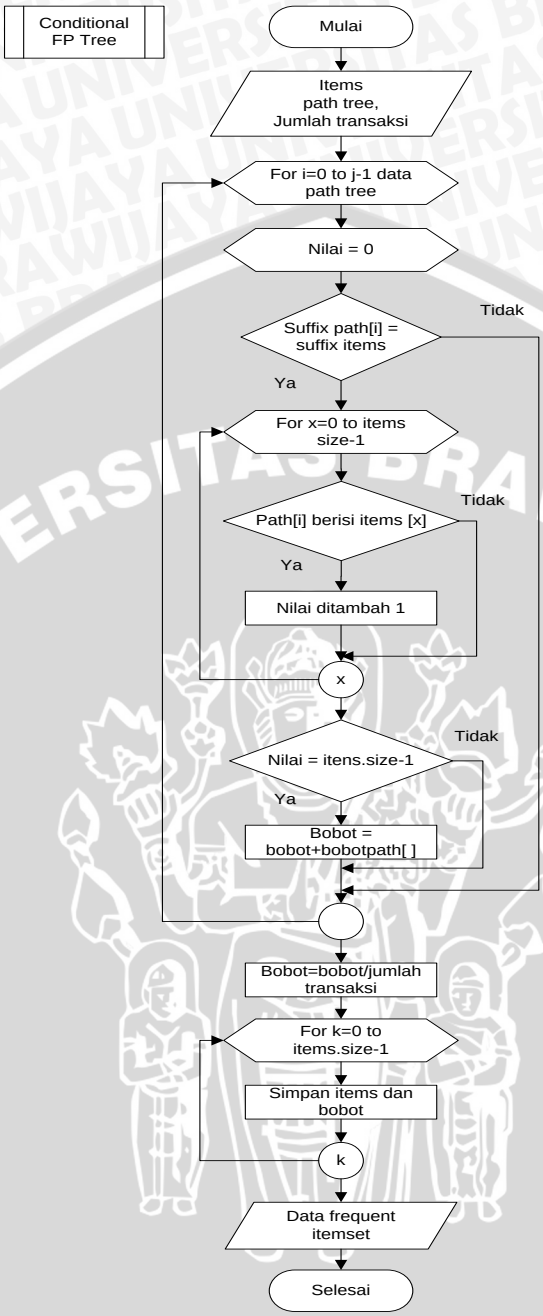
Apabila semua kode jenis ikan pada data transaksi sudah dimasukkan ke dalam pembentukan *FP Tree*, tahap selanjutnya adalah proses *conditional pattern base*. Pada proses ini setiap lintasan (*path*) di *tree* yang dibaca oleh sistem dilakukan pencarian *prefix* yaitu kode jenis ikan yang merangkai lintasan. Pencarian *prefix* untuk setiap *suffix* dimana yang menjadi *suffix* yaitu setiap kode jenis ikan pada *frequent 1-itemset* yang ada. Proses ini digambarkan pada Gambar 3.11 sebagai berikut :



Gambar 3.11 Diagram Alir *Conditional Pattern Base*

3.2.3.7 Tahap *Conditional FP Tree*

Sebagai bagian dari proses kelanjutan dari algoritma *FP Growth* yaitu tahap *conditional FP Tree* dengan tujuan untuk menghitung nilai *support count* setiap kombinasi kode jenis ikan yang menjadi *prefix,* yang mempunyai nilai *support* lebih besar atau sama dengan nilai *minimum support* dengan *suffix* hasil dari proses *conditional pattern base*. Proses dilakukan dengan mengecek pada setiap data lintasan (*path*) apakah memiliki *suffix* yang sama dengan *suffix* kombinasi. Jika sama maka dilakukan pengecekan terlebih dahulu apakah kode jenis ikan penyusun data lintasan tersebut memiliki semua kode jenis ikan yang menjadi *prefix* kombinasi. Jika semua kode jenis ikan dimiliki data lintasan maka bobot data lintasan tersebut dijumlahkan. Kemudian nilai bobot tersebut dibagi dengan jumlah transaksi lalu disimpan sebagai data *frequent itemset*. Proses ini digambarkan pada Gambar 3.12 sebagai berikut :



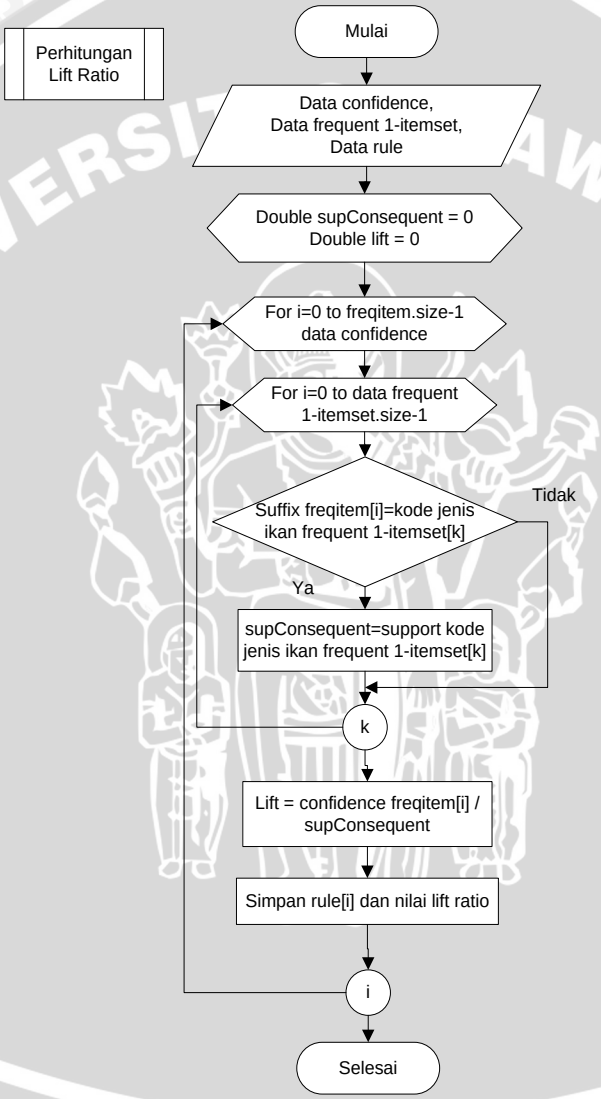
Gambar 3.12 Diagram Alir *Conditional FP Tree*

3.2.3.8 Tahap Perhitungan *Lift Ratio*

Tahap terakhir dari sistem ini adalah tahap perhitungan nilai *lift ratio* dari setiap *rule* yang terbentuk. Proses ini digambarkan pada Gambar 3.13 dengan deskripsi alur diagram sebagai berikut :

1. Setiap data *frequent itemset* yang menjadi *rule* dihitung nilai *lift ratio*-nya.

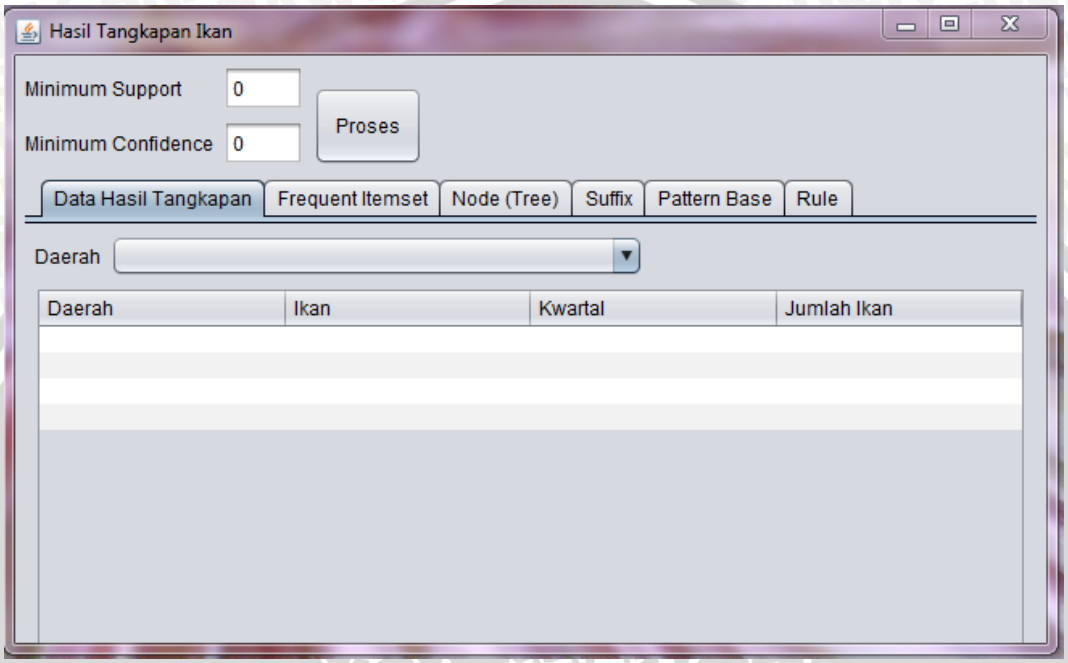
2. Cara menghitungnya adalah setiap *suffix* dari data *frequent itemset* yang menjadi *rule* dilakukan pencarian nilai *support*-nya pada data *frequent 1-itemset* dan disimpan sebagai supConsequent.
3. Nilai *confidence* dari *frequent itemset* yang menjadi *rule* tersebut dibagi dengan nilai supConsequent.
4. Hasil dari proses tersebut disimpan yaitu berupa *rule* dan nilai *lift ratio rule*.



Gambar 3.13 Diagram Alir Perhitungan *Lift Ratio*

3.2.4 Perancangan Antarmuka

Perancangan antarmuka secara umum untuk perangkat lunak implementasi metode *association rule* ini terdapat beberapa bagian utama yaitu bagian masukan, keluaran, dan tombol proses. Tampilan rancangan antarmuka secara umum dapat dilhat pada Gambar 3.14.



Gambar 3.14 Rancangan Antarmuka Secara Umum

Bagian masukan terdiri dari dua data masukan yaitu data masukan nilai *minimum support*, dan data masukan nilai *minimum confidence*. Setelah memasukkan data masukan, maka pengguna menekan tombol proses untuk memulai proses penerapan metode *association rule*.

Ketika tombol proses sudah ditekan, maka akan muncul submenu yang terdiri dari submenu data hasil tangkapan, submenu *frequent itemset*, submenu *node*(tree), submenu *suffix*, submenu *conditional fptree*, dan submenu *rule*.

3.2.5 Perancangan Uji Coba dan Evaluasi

Perancangan pengujian perangkat lunak ini dimaksudkan agar dapat mengetahui kinerja dari perangkat lunak. Selain itu, sebagai bahan untuk mengevaluasi hasil dari implementasi analisis pola hasil tangkapan ikan laut dan perancangan perangkat lunaknya. Tujuan dari uji coba ini adalah untuk

mengetahui pengaruh nilai *minimum support* dan nilai *minimum confidence* serta mengetahui nilai *lift ratio* terhadap *rule* yang dihasilkan. Uji coba pengaruh nilai *minimum support* dan *minimum confidence* dilakukan pada sejumlah data transaksi dengan parameter daerah pilihan yang telah dipilih.

Tabel uji coba pengaruh nilai *minimum support* dan *minimum confidence* ditunjukkan pada Tabel 3.1. Sedangkan untuk tabel uji coba *lift ratio rule* ditunjukkan pada Tabel 3.2.

Tabel 3.1 Tabel Uji Coba Pengaruh Nilai *Minimum Support* dan Nilai *Minimum Confidence* terhadap Jumlah *Rule* yang Dihasilkan

Daerah	**Jumlah *Rule* yang Dihasilkan**	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**

Tabel 3.2 Tabel Uji Coba *Lift Ratio* terhadap *Rule* yang Dihasilkan

Daerah	***Rule* yang Dihasilkan**	**Nilai *Confidence***	**Nilai *Lift Ratio***

3.3 Perhitungan Manual

Proses perhitungan manual dilakukan dengan mengambil sampel data dibuat secara random. Data yang digunakan adalah data asli dari lapangan dan telah ditentukan parameter datasetnya. Parameter dataset yang dipilih untuk contoh perhitungan manual adalah dataset daerah yaitu daerah Kabupaten Banyuwangi. Nilai *minimum support* yang digunakan pada perhitungan manual adalah sebesar 20%, sedangkan untuk nilai *minimum confidence* sebesar 50%. Sebelum melakukan perhitungan manual, terlebih dahulu dilakukan proses seleksi, *preprocessing* dan transformasi jika dibutuhkan untuk mendapatkan data yang sesuai. Proses seleksi dilakukan pada data yaitu dengan cara menyeleksi

ikan-ikan yang mempunyai jumlah ikan yang besar dan hasil proses seleksi dapat dilihat pada Tabel 3.3.

Tabel 3.3 Sampel Data Hasil Seleksi

Id daerah	Jenis ikan	Jumlah Ikan(kg)
14	Cucut	381.100
14	Layang	1.722.800
14	Lemuru	45.057.000
14	Tuna	384.500
14	Cakalang	275.700
14	Tongkol	1.348.700
14	Manyung	490.300
14	Kakap	528.300

Setelah diseleksi, sampel data dilakukan proses *preprocessing* yaitu melakukan perbaikan data dengan menambahkan pengkodean kode transaksi, menghapus id daerah, menghapus jenis ikan, menghapus jumlah ikan dan membuat data kemunculan jenis ikan pada tiap transaksi. Pembuatan data jenis ikan yang muncul pada tiap transaksi penangkapan dilakukan dengan secara random.

Pembuatan kode transaksi dengan cara mengkodekan daerah Banyuwangi dengan D14 dan menambahkan kode kuartal pada semua tahun. Misalkan dalam satu pengkodean kode transaksi mempunyai arti bahwa pada daerah Banyuwangi di kuartal 1 tahun 2004 dituliskan dengan format kode transaksi D1401. Hasil proses *preprocessing* dapat dilihat pada Tabel 3.4. Kode jenis ikan dapat dilihat pada Tabel 3.5. Kemudian dilakukan proses transformasi dengan cara mengganti nama jenis ikan menjadi kode jenis ikan, untuk memudahkan pembacaan yang dapat dilihat pada Tabel 3.6.

Tabel 3.4 Sampel Data Hasil *Preprocessing*

Kode transaksi	Jenis ikan
D1401	Lemuru, tongkol
D1402	Cucut, lemuru, tongkol
D1403	Lemuru, tuna, tongkol
D1404	Cakalang, tongkol
D1405	Lemuru, tongkol
D1406	Lemuru, tongkol
D1407	Cucut,tongkol, kakap
D1408	Cucut, layang, cakalang, tongkol, manyung
D1409	Lemuru, tongkol
D1410	Cucut, cakalang, manyung

Tabel 3.5 Sampel Data Jenis Ikan

Kode jenis ikan	Jenis ikan
1	Cucut
2	Layang
3	Lemuru
4	Tuna
5	Cakalang
6	Tongkol
7	Manyung
8	Kakap

Tabel 3.6 Sampel Data Hasil Transformasi

Kode transaksi	Jenis ikan
D1401	3, 6
D1402	1, 3, 6
D1403	3, 4, 6
D1404	5, 6
D1405	3, 6

D1406	3, 6
D1407	1, 6, 8
D1408	1, 2, 5, 6, 7
D1409	3, 6
D1410	1, 5, 7

Dengan menggunakan sampel data hasil transformasi pada Tabel 3.6 dilakukan perhitungan nilai frekuensi kemunculan tiap jenis ikan yang ditunjukkan pada Tabel 3.7. Kemudian nilai frekuensi kemunculan jenis ikan diurutkan dimulai dari yang terbesar ke terkecil yang ditunjukkan pada Tabel 3.8.

Tabel 3.7 Frekuensi Kemunculan Tiap Jenis Ikan

Kode jenis ikan	Frekuensi Kemunculan
1	4
2	1
3	6
4	1
5	3
6	9
7	2
8	1

Tabel 3.8 Frekuensi Kemunculan Tiap Jenis Ikan yang Telah Diurutkan

Kode jenis ikan	Frekuensi Kemunculan
6	9
3	6
1	4
5	3
7	2
2	1
4	1
8	1

Hasil frekuensi kemunculan tiap kode jenis ikan yang telah diurutkan maka digunakan dalam proses selanjutnya yaitu menghitung nilai *support* pada *frequent 1-itemset* sebagai berikut :

Nilai *minimum support* = 20%

$$Support\ (A) = \frac{\text{Jumlah Transaksi yang Mengandung A}}{\text{Total Transaksi}}$$

support {6} = 9/10 = 0.9
support {3} = 6/10 = 0.6
support {1} = 4/10 = 0.4
support {5} = 3/10 = 0.3
support {7} = 2/10 = 0.2
support {2} = 1/10 = 0.1
support {4} = 1/10 = 0.1
support {8} = 1/10 = 0.1

Berdasarkan hasil perhitungan *support* pada *frequent 1-itemset* dapat dilihat bahwa kode jenis ikan 2, 4, dan 8 tidak *frequent* karena nilai *support* masing – masing kode jenis ikan tersebut lebih kecil dari nilai *minimum support* = 20%. Sedangkan *item* kode jenis ikan 6, 3, 1, 5, dan 7 terpilih menjadi *frequent 1-itemset* karena nilai *support*-nya lebih besar atau sama dengan nilai *minimum support* = 20%. Untuk kode jenis ikan yang memenuhi aturan nilai *minimum support* dapat dilihat pada Tabel 3.9.

Tabel 3.9 Jenis Ikan yang Memenuhi Nilai *Minimum Support*

Kode jenis ikan	***Support***
6	0.9
3	0.6
1	0.4
5	0.3
7	0.2

Setelah didapatkan jenis ikan apa saja yang telah memenuhi nilai *minimum support*, tahap selanjutnya dilakukan proses penghapusan kode jenis ikan yang tidak *frequent* dari data transaksi dan kode jenis ikan pada tiap transaksi diurutkan

berdasarkan nilai *support* tertinggi. Kode jenis ikan 6 adalah kode jenis ikan yang mempunyai nilai *support* tertinggi, maka kode jenis ikan 6 diletakkan pada posisi awal tiap transaksi. Perubahan pada sampel data transaksi dapat dilihat pada Tabel 3.10.

Tabel 3.10 Sampel Data Setelah Penghapusan Kode Jenis Ikan yang Tidak *Frequent* dan Penyortiran

Kode transaksi	Kode jenis ikan
D1401	6,3
D1402	6,3,1
D1403	6,3
D1404	6,5
D1405	6,3
D1406	6,3
D1407	6,1
D1408	6,1,5,7
D1409	6,3
D1410	1,5,7

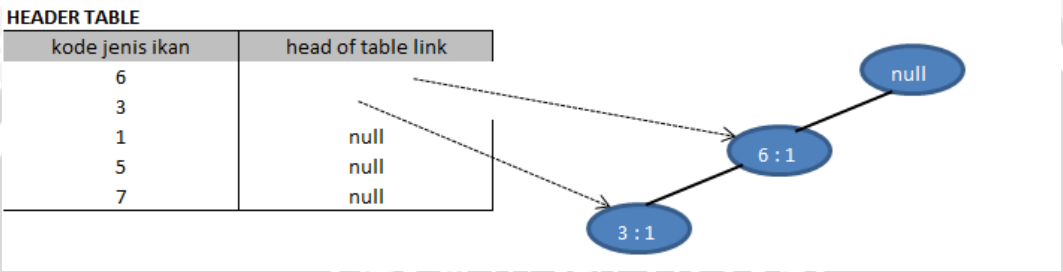
Tahap pembentukan *FP Tree* diawali dengan pembuatan awal *root* yang diberikan nilai *null*. Pada *header table, head of table link* untuk semua kode jenis ikan mempunyai nilai *null*. Tahap pembentukan simpul *FP Tree* awal dapat dilhat pada Gambar 3.15.

HEADER TABLE

kode jenis ikan	head of table link
6	Null
3	Null
1	null
5	null
7	Null

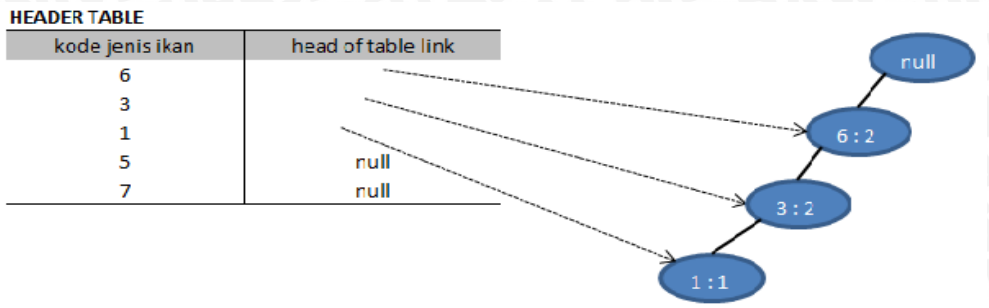
Gambar 3.15 Pembentukan Simpul *FP Tree* Awal

Pembacaan transaksi pertama pada sampel data yaitu transaksi pertama dengan kode transaksi D1401. Berdasarkan urutan kode jenis ikan yang muncul adalah kode jenis ikan 6 dan 3. Kode jenis ikan 6 merupakan *prefix* sehingga simpul anak *root* berisi kode jenis ikan 6 dan mempunyai *count support* bernilai 1 yang artinya kode jenis ikan 6 yang melewati lintasan tersebut adalah sebanyak 1. Selanjutnya kode jenis ikan 3 menjadi simpul anak dari simpul berlabel kode jenis ikan 6. *Head of table link* menunjukkan pada simpul yang mana kode jenis ikan tersebut berada dan disimbolkan tanda panah dengan garis panah putus – putus. Proses pembacaan transaksi D1401 dapat dilihat pada Gambar 3.16.



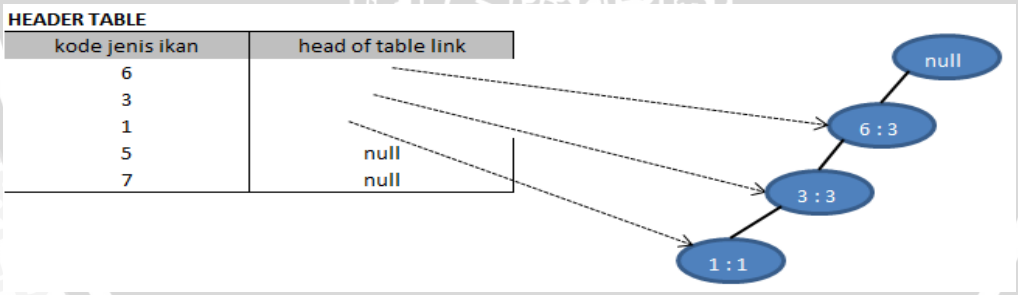
Gambar 3.16 *FP Tree* Setelah Pembacaan Kode Transaksi D1401

Pada transaksi D1402 ini, transaksi dengan urutan kode jenis ikan yang muncul adalah 6, 3, dan 1. *Prefix* dari transaksi ini sama dengan transaksi sebelumnya yaitu kode jenis ikan 6 sehingga tidak perlu dibuat simpul anak baru dari *root*. *Prefix* tersebut ditimpakan pada simpul berlabel kode jenis ikan 6 sehingga pada simpul tersebut *count support* bertambah 1 dan *count support* menjadi bernilai = 2. Kode jenis ikan selanjutnya yang dimasukkan ke dalam *tree* adalah kode jenis ikan 3. Pada simpul berlabel kode jenis ikan 6 ditelusuri simpul anaknya. Begitu juga dengan simpul anak dari simpul kode jenis ikan 6 adalah simpul berlabel kode jenis ikan 3, maka kode jenis ikan 3 pada transaksi D1402 ditimpakan juga pada simpul berlabel kode jenis ikan 3 yang sebelumnya telah ada dan *count support* pada simpul berlabel kode jenis ikan 3 ditambahkan 1 menjadi *count support* = 2. Kemudian kode jenis ikan 1 dimasukkan ke dalam *tree* menjadi simpul anak dari simpul kode jenis ikan 3. Karena pada simpul kode jenis ikan 3 pada pembacaan transaksi sebelumnya tidak terbentuk simpul anak, maka simpul 1 dibuat dengan *count support* = 1. Pembacaan transaksi kedua dengan kode transaksi D1402 dapat dilhat pada Gambar 3.17.



Gambar 3.17 *FP Tree* Setelah Pembacaan Kode Transaksi D1402

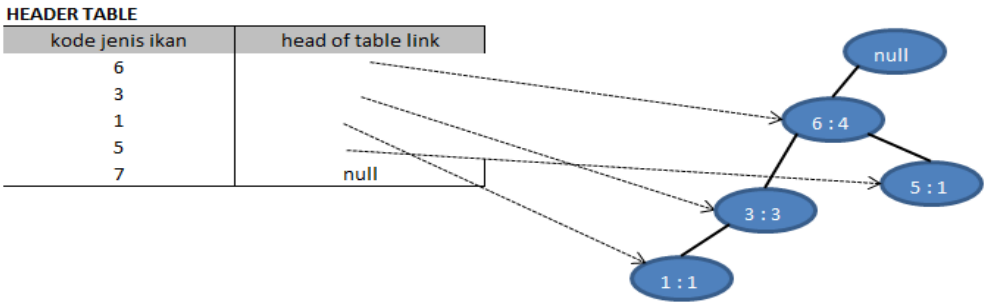
Transaksi ketiga yang dibaca adalah kode transaksi D1403 dengan urutan kode jenis ikan yang muncul adalah 6 dan 3. *Prefix* dari transaksi D1403 yaitu diawali kode jenis ikan 6, karena sama dengan transaksi sebelumnya sehingga ditimpakan pada simpul anak *root* yang berlabel kode jenis ikan 6 dan simpul tersebut mempunyai *count support* bertambah 1 menjadi *count support* = 3. Kode jenis ikan selanjutnya adalah kode jenis ikan 3. Pada simpul berlabel kode jenis ikan 6 ditelusuri simpul anaknya. Simpul anak dari simpul kode jenis ikan 6 pada transaksi D1403 adalah simpul berlabel kode jenis ikan 3, maka kode jenis ikan 3 ditimpakan juga pada simpul berlabel kode jenis ikan 3 yang telah ada dan mempunyai *count support* = 3. Pembacaan transaksi D1403 dapat dilihat pada Gambar 3.18.



Gambar 3.18 *FP Tree* Setelah Pembacaan Kode Transaksi D1403

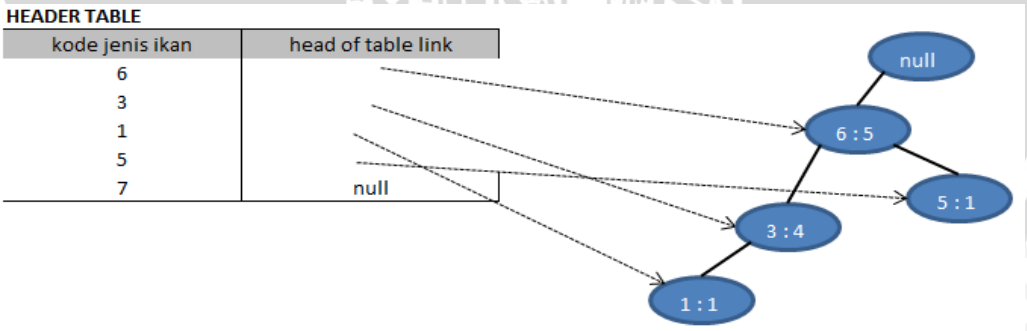
Transaksi keempat yang dibaca adalah kode transaksi D1404 dengan urutan kode jenis ikan yang muncul adalah 6 dan 5. *Prefix* dari transaksi D1404 yaitu kode jenis ikan 6 sama dengan transaksi sebelumnya sehingga ditimpakan pada simpul anak *root* yang berlabel kode jenis ikan 6 dan pada simpul tersebut mempunyai *count support* = 4. Selanjutnya kode jenis ikan 5 dimasukkan ke dalam *tree*. Simpul kode jenis ikan 6 tidak mempunyai simpul anak berlabel kode jenis ikan 5, maka dibuatkan simpul anak baru dari simpul kode jenis ikan 6

dengan label kode jenis ikan 5 dan mempunyai *count support* = 1. Pembacaan transaksi D1404 dapat dilhat pada Gambar 3.19.



Gambar 3.19 *FP Tree* Setelah Pembacaan Kode Transaksi D1404

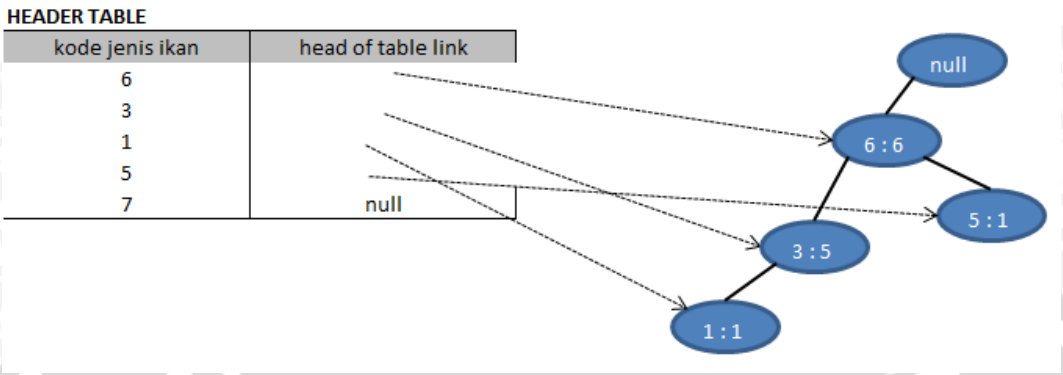
Transaksi kelima yang dibaca adalah kode D1405 dengan urutan kode jenis ikan yang muncul adalah 6 dan 3. *Prefix* dari transaksi ini yaitu kode jenis ikan 6 sama dengan transaksi sebelumnya sehingga ditimpakan pada simpul anak *root* yang berlabel kode jenis ikan 6 dan simpul tersebut mempunyai *count support* = 5 karena *count support* ditambah 1. Selanjutnya kode jenis ikan 3 dimasukkan ke dalam *tree*. Pada simpul berlabel kode jenis ikan 6 ditelusuri simpul anaknya, karena terdapat simpul anak berlabel kode jenis ikan 3 maka kode jenis ikan 3 ditimpakan pada simpul berlabel kode jenis ikan 3 dan mempunyai nilai *count support* = 4. Proses pembacaan transaksi D1405 dapat dilihat pada Gambar 3.20.



Gambar 3.20 *FP Tree* Setelah Pembacaan Kode Transaksi D1405

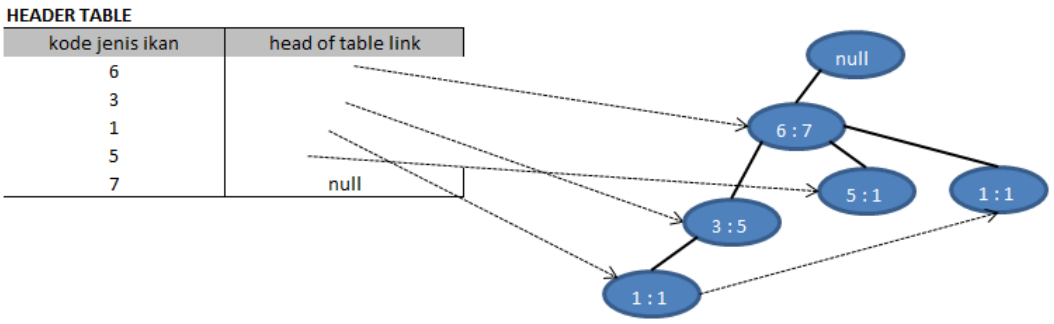
Transaksi keenam yang dibaca adalah kode transaksi D1406 dengan urutan kode jenis ikan yang muncul adalah 6 dan 3. *Prefix* dari transaksi D1406 yaitu kode jenis ikan 6 sama dengan transaksi sebelumnya sehingga ditimpakan pada simpul anak *root* yang berlabel kode jenis ikan 6 dan mempunyai nilai *count support* = 6. Selanjutnya kode jenis ikan 3 dimasukkan ke dalam *tree* dan simpul

berlabel kode jenis ikan 6 ditelusuri simpul anaknya. Karena simpul kode jenis ikan 6 memiliki simpul anak berlabel kode jenis ikan 3, maka nilai *count support* simpul kode jenis ikan 3 menjadi *count support* = 5. Proses pembacaan transaksi D1406 dapat dilihat pada Gambar 3.21.



Gambar 3.21 *FP Tree* Setelah Pembacaan Kode Transaksi D1406

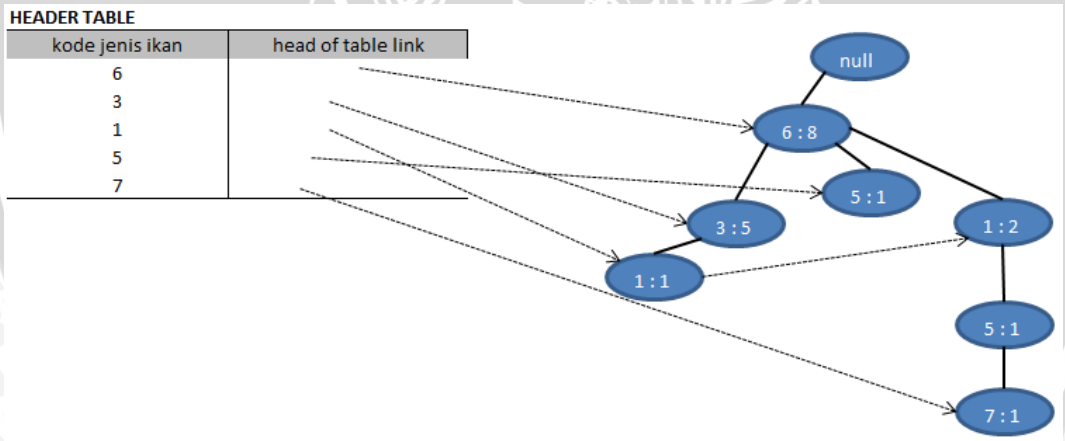
Transaksi yang dibaca pada gambar 3.22 adalah kode D1407 dengan urutan kode jenis ikan yang muncul adalah 6 dan 1. *Prefix* dari transaksi D1407 yaitu kode jenis ikan 6 sama dengan transaksi sebelumnya sehingga ditimpakan pada simpul anak *root* yang berlabel kode jenis ikan 6 dan simpul tersebut mempunyai nilai *count support* = 7. Kode jenis ikan selanjutnya yang dimasukkan ke dalam *tree* adalah kode jenis ikan 1 dan menelusuri simpul anak dari simpul berlabel kode jenis ikan 6. Karena simpul kode jenis ikan 6 tidak mempunyai simpul anak berlabel kode jenis ikan 1, maka dibuatkan simpul anak baru dengan berlabel kode jenis ikan 1 dengan nilai *count support* = 1 dan dihubungkan dengan simpul berlabel 1 pada lintasan lain.



Gambar 3.22 *FP Tree* Setelah Pembacaan Kode Transaksi D1407

Pada kode transaksi D1408 dengan urutan kode jenis ikan yang muncul adalah 6, 1, 5, dan 7. *Prefix* dari transaksi D1408 yaitu kode jenis ikan 6 sama

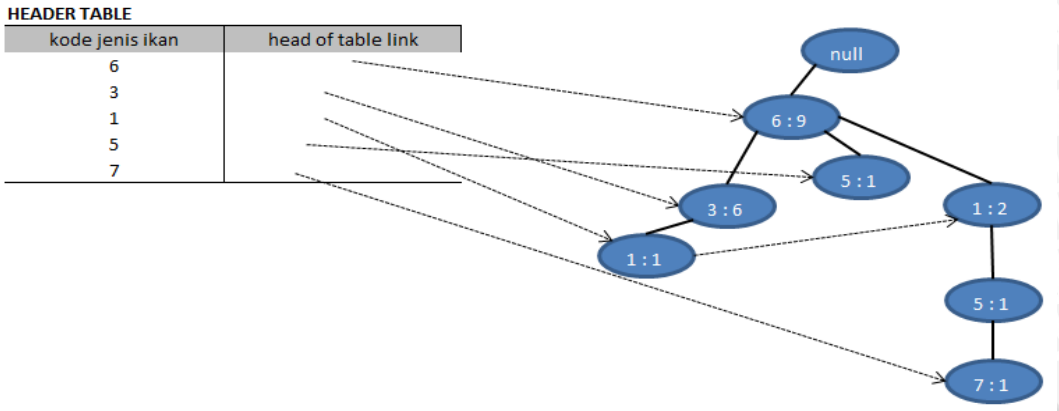
dengan transaksi sebelumnya sehingga ditimpakan pada simpul anak *root* yang berlabel kode jenis ikan 6 dan simpul tersebut mempunyai nilai *count support* = 8. Selanjutnya kode jenis ikan 1 dimasukkan ke dalam *tree* maka simpul berlabel kode jenis ikan 6 ditelusuri simpul anaknya. Simpul kode jenis ikan 6 memiliki simpul anak berlabel kode jenis ikan 1, sehingga *count support* simpul berlabel kode jenis ikan 1 dari *root* simpul berlabel kode jenis ikan 6 ditambahkan 1 (*count support* = 2). Kode jenis ikan 5 kemudian dimasukkan ke dalam *tree* sehingga simpul kode jenis ikan 1 pada lintasan tersebut ditelusuri simpul anaknya. Karena tidak ada simpul anak pada simpul berlabel kode jenis ikan 1 maka dibuat simpul anak baru berlabel kode jenis ikan 5 dengan nilai *count support* = 1. Kemudian dilanjutkan dengan kode jenis ikan 7 yang dimasukkan ke dalam *tree* sehingga simpul kode jenis ikan 5 yang telah dibuat tersebut ditelusuri simpul anaknya. Karena tidak memiliki simpul anak maka dibuatkan simpul anak baru berlabel kode jenis ikan 7 dari simpul berlabel kode jenis ikan 5 dengan nilai *count support* = 1. Proses pembacaan transaksi D1408 dapat dilihat pada Gambar 3.23.



Gambar 3.23 *FP Tree* Setelah Pembacaan Kode Transaksi D1408

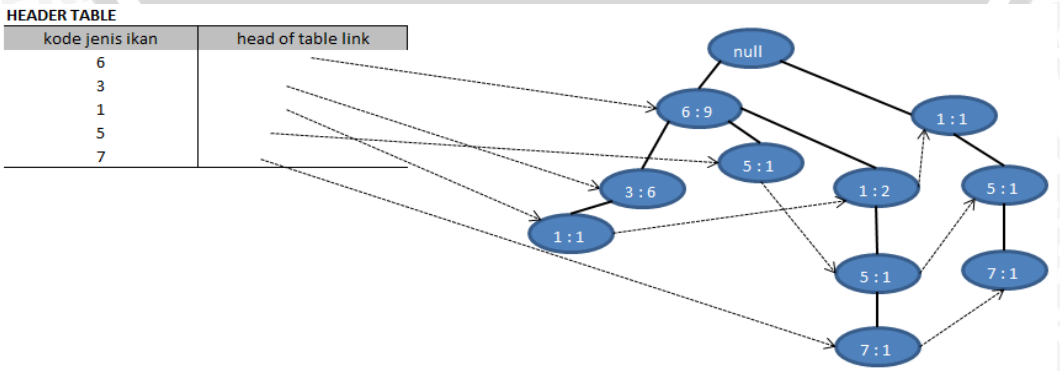
Pada Gambar 3.24, transaksi yang dibaca adalah kode transaksi D1409 dengan urutan kode jenis ikan yang muncul adalah 6 dan 3. *Prefix* dari transaksi ini yaitu kode jenis ikan 6 sehingga ditimpakan pada simpul anak *root* yang berlabel kode jenis ikan 6 dan simpul tersebut mempunyai nilai *count support* = 9. Selanjutnya kode jenis ikan 3 dimasukkan ke dalam *tree*. Pada simpul berlabel kode jenis ikan 6 ditelusuri simpul anaknya. Karena simpul kode jenis ikan 6

memiliki simpul anak berlabel kode jenis ikan 3, maka nilai *count support* simpul kode jenis ikan 3 ditambahkan 1 (*count support* = 6).



Gambar 3.24 *FP Tree* Setelah Pembacaan Kode Transaksi D1409

Pada Gambar 3.25, transaksi terakhir yang dibaca adalah kode transaksi D1410 dengan urutan kode jenis ikan yang dibeli adalah 1, 5 dan 7. *Prefix* dari transaksi D1410 yaitu kode jenis ikan 1. Pada *root* ditelusuri simpul anak berlabel kode jenis ikan 1. Karena simpul anak dari *root* hanya ada simpul anak berlabel kode jenis ikan 6 maka dibuat simpul anak berlabel kode jenis ikan 1 dengan *count support* = 1. Selanjutnya kode jenis ikan 5 dimasukkan ke dalam *tree*. Pada simpul berlabel kode jenis ikan 1 yang telah dibuat ditelusuri simpul anaknya. Karena simpul kode jenis ikan 1 tidak memiliki simpul anak maka dibuat simpul anak berlabel kode jenis ikan 5 dari simpul kode jenis ikan 1 dengan *count support* = 1. Kode jenis ikan berikutnya adalah kode jenis ikan 7. Pada simpul berlabel kode jenis ikan 5 dalam lintasan yang sama ditelusuri simpul anaknya. Karena simpul kode jenis ikan 5 tidak memiliki anak maka dibuat simpul anak berlabel kode jenis ikan 7 dari simpul kode jenis ikan 5 dengan *count support* = 1.



Gambar 3.25 *FP Tree* Setelah Pembacaan Kode Transaksi D1410

Tahap selanjutnya adalah *conditional pattern base* yaitu mencari *suffix pattern* (pola akhiran) dan *prefix path* (lintasan awal) pada *FP Tree* yang telah terbentuk seperti yang dapat dilihat pada Tabel 3.11.

Pencarian *conditional pattern base* dimulai dari *suffix* {7} sebagai simpul paling bawah dari *FP Tree* yang terbentuk seperti pada Gambar 3.26. Berdasarkan pembacaan sampel data pada Tabel 3.10 bahwa s*uffix* {7} mempunyai dua lintasan yaitu lintasan pertama {6, 1, 5} dan lintasan kedua {1, 5}. Masing-masing simpul diberikan nilai frekuensi = 1, karena simpul tersebut hanya melintasi masing-masing satu kali. Sehingga didapatkan *conditional pattern base* lintasan pertama {6:1, 1:1, 5:1} dan lintasan kedua {1:1, 5:1}.

Berdasarkan sampel data Tabel 3.10, s*uffix* {5} melintasi tiga lintasan yaitu lintasan pertama {6}, lintasan kedua {1} dan lintasan ketiga {6,1}. Masing-masing simpul diberikan nilai frekuensi = 1, karena simpul tersebut hanya melintasi masing-masing satu kali. Sehingga didapatkan *conditional pattern base* lintasan pertama {6:1), lintasan kedua {1:1} dan lintasan ketiga {6:1, 1:1}.

Berdasarkan sampel data Tabel 3.10, *suffix* {1} melintasi dua lintasan yaitu lintasan pertama {6,3} dan lintasan kedua {6}. Pada lintasan pertama, masing-masing simpul diberikan nilai frekuensi = 1. Sedangkan lintasan kedua, *suffix* {1} telah melintasi simpul 6 sebanyak dua kali yaitu pada transaksi D1407 dan D1408 sehingga simpul 6 pada *suffix* {1} diberikan nilai frekuensi = 2. *Conditional pattern base* dari *suffix* {1} yang didapatkan adalah lintasan pertama {6:1, 3:1} dan lintasan kedua {6:2}.

Berdasarkan sampel data Tabel 3.10, *suffix* {3} melintasi satu lintasan saja yaitu pada lintasan dengan simpul 6. Namun *suffix* {3} yang melintasi lintasan dengan simpul 6 tersebut berulang sebanyak 6 kali transaksi yaitu pada transaksi dengan kode D1401, D1402, D1403, D1405, D1406 dan D1409. Sehingga dengan *suffix* {3} didapatkan *conditional pattern base* yaitu {6:6}. Sedangkan untuk *suffix* {6} hanya melintasi *root* maka *conditional pattern base* yang didapatkan adalah *null*.

Tabel 3.11 *Conditional Pattern Base* dari Sampel Data

Suffix	***Conditional Pattern Base***
{7}	{6:1,1:1,5:1},{1:1,5:1}
{5}	{6:1},{1:1},{6:1,1:1}
{1}	{6:1,3:1},{6:2}
{3}	{6:6}
{6}	Null

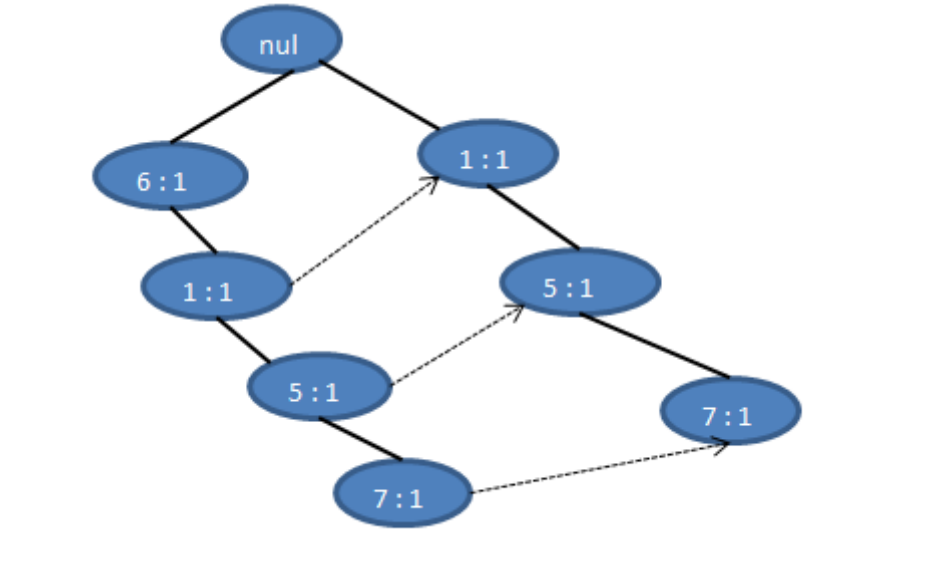
Tahap *conditional FP-tree* yaitu *support count* setiap kode jenis ikan dalam *prefix path* (lintasan awal) setiap *suffix* dijumlahkan. Kode jenis ikan yang memiliki jumlah *support count* lebih besar atau sama dengan nilai *minimum support* = 20% akan dibangkitkan dengan *conditional FP Tree*. Untuk *suffix* {6} tidak dihitung *support count-nya,* karena *suffix* {6} mempunyai *conditional pattern base* bernilai null. Perhitungan *support count* dilihat berdasarkan *conditional pattern base* yang dapat dilihat pada Tabel 3.11.

$$Support\ (A) = \frac{\text{Jumlah Transaksi yang Mengandung A}}{\text{Total Transaksi}}$$

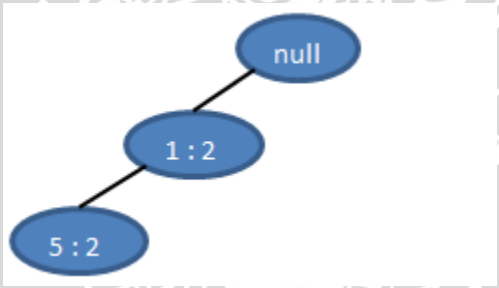
suffix {7} :
support {6} = 1/10 = 0.1
support {1} = 2/10 = 0.2
support {5} = 2/10 = 0.2
suffix {5} :
support {6} = 2/10 = 0.2
support {1} = 2/10 = 0.2
suffix {1} :
support {6} = 2/10 = 0.2
support {3} = 1/10 = 0.1
suffix {3} :
support {6} = 6/10 = 0.6

Setelah dihitung *support* count-nya, dilanjutkan proses pembentukan *conditional FP Tree*. Pada Gambar 3.26 menggambarkan *FP Tree* dengan lintasan yang memiliki *suffix* {7}, sedangkan pembentukan c*onditional FP Tree* yang dibangkitkan untuk *suffix* {7} dan diberikan nilai frekuensi = 2 dapat dilihat pada Gambar 3.27. Karena kode jenis ikan 6 memiliki nilai *support* = 0.1 yaitu dibawah

nilai *minimum support* sehingga tidak ikut dibangkitkan pada *conditional FP Tree*.

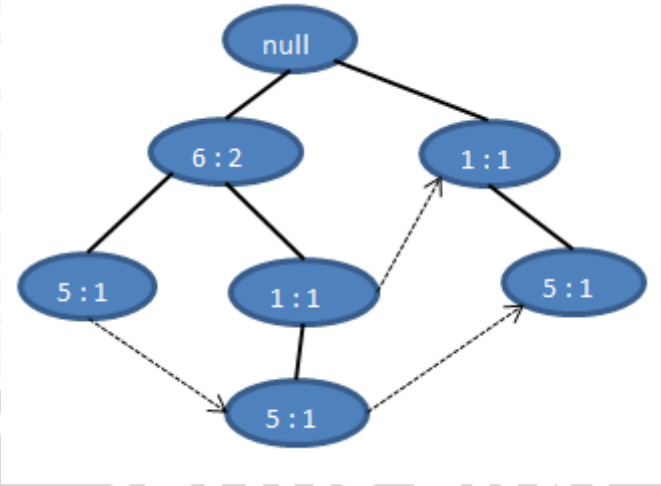


Gambar 3.26 Lintasan yang Memiliki *Suffix* {7}

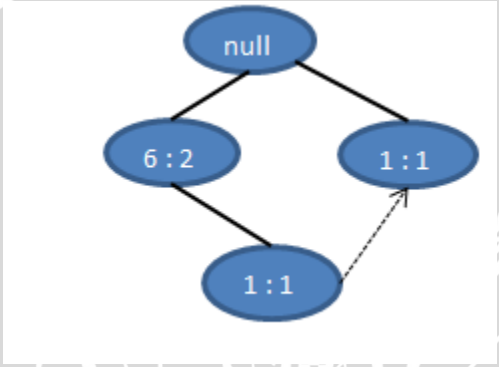


Gambar 3.27 *Conditional FP Tree* untuk *Suffix* {7}

Berdasarkan Gambar 3.27, karena lintasan yang terbentuk berupa *single path* maka *frequent itemset* yang didapat dari *suffix* {7} adalah {1,7}, {5,7}, dan {1,5,7} dengan masing–masing frekuensi sebanyak 2. Pada Gambar 3.28 menggambarkan lintasan yang memiliki *suffix* {5}. Pembentukan c*onditional FP Tree* yang dibangkitkan untuk *suffix* {5} dapat dilihat pada Gambar 3.29.



Gambar 3.28 Lintasan yang Memiliki *Suffix* {5}



Gambar 3.29 *Conditional FP-tree* untuk *Suffix* {5}

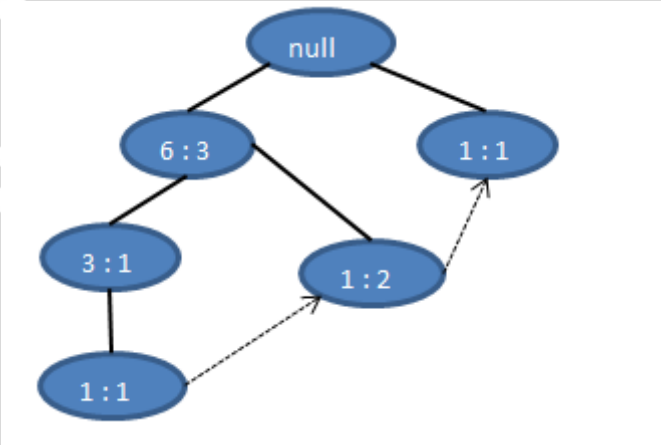
Lintasan yang terbentuk dengan *suffix* {1,5} merupakan bukan *single path* maka dilakukan *generate conditional FP Tree* kembali dengan *suffix* kode jenis ikan 1 dan 5 yang dapat dilihat pada Gambar 3.30.

Gambar 3.30 *Conditional FP Tree* untuk *Suffix* {1,5}

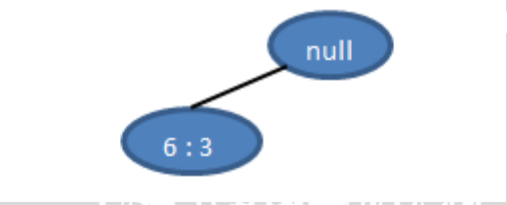
Hasil *conditional FP-tree* untuk *suffix* {1,5} adalah *null* pada Gambar 3.30, karena *support* dari lintasan {6,1,5} kurang dari nilai *minimum support*. Berdasarkan Gambar 3.30, lintasan yang dihasilkan adalah *single path* sehingga *frequent itemset* yang didapat dari *suffix* {1,5} adalah {1,5} dengan frekuensi sebanyak 2. Kemudian dilakukan *generate conditional FP Tree* kembali dengan *suffix* {6,5} yang dapat dilihat pada Gambar 3.31.

Gambar 3.31 *Conditional FP Tree* untuk *Suffix* {6,5}

Hasil *conditional FP Tree* untuk *suffix* {6,5} merupakan *single path*, sehingga *frequent itemset* yang didapat dari *suffix* {6,5} adalah {6,5} dengan frekuensi sebanyak 2. Pada Gambar 3.32 menggambarkan lintasan yang memiliki *suffix* {1}. *Conditional FP Tree* untuk *suffix* {1} dapat dilihat pada Gambar 3.33.

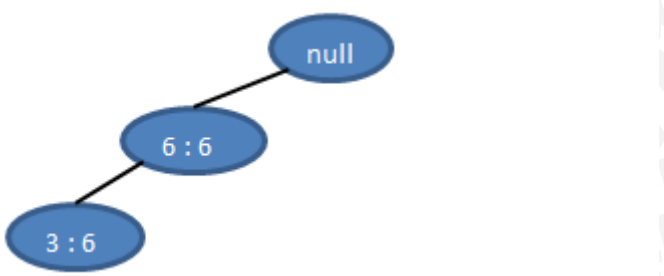


Gambar 3.32 Lintasan yang Memiliki *Suffix* {1}

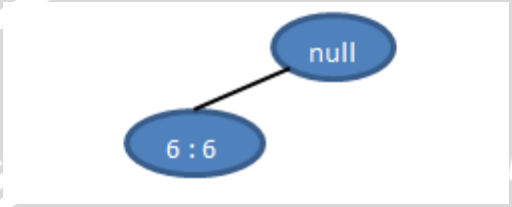


Gambar 3.33 *Conditional FP Tree* untuk *Suffix* {1}

Berdasarkan Gambar 3.33, lintasan dari *conditional FP-tree* merupakan *single path* sehingga *frequent itemset* yang didapat untuk *suffix* {1} adalah {6,1} dengan frekuensi sebanyak 3. Pada Gambar 3.34 menggambarkan lintasan yang memiliki *suffix* {3}. *Conditional FP Tree* untuk *suffix* {3} dapat dilihat pada Gambar 3.35.

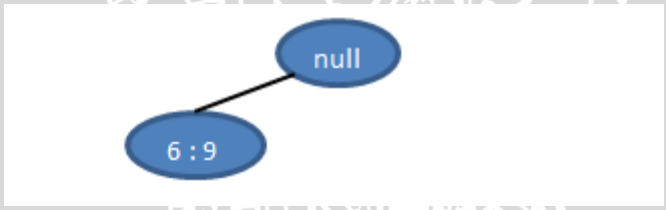


Gambar 3.34 Lintasan yang Memiliki *Suffix* {3}



Gambar 3.35 *Conditional FP Tree* untuk *Suffix* {3}

Lintasan yang terbentuk pada Gambar 3.35 merupakan *single path* maka *frequent itemset* yang didapatkan untuk *suffix* {3} adalah {6,3} dengan frekuensi sebanyak 6. Gambar 3.36 menggambarkan lintasan yang memiliki *suffix* {6}, sedangkan Gambar 3.37 menggambarkan *conditional FP Tree* untuk *suffix* {6}.



Gambar 3.36 Lintasan yang Memiliki *Suffix* {6}

Gambar 3.37 *Conditional FP Tree* untuk *Suffix* {6}

Berdasarkan Gambar 3.37, lintasan yang dihasilkan dari *conditional FP Tree* maka *frequent itemset* yang didapatkan tidak ada (*null*). Hasil *frequent itemset* dari setiap proses *conditional FP Tree* digabungkan menjadi satu dalam suatu tabel yaitu tabel *Frequent Itemset* seperti pada Tabel 3.12. Nilai frekuensi dari masing-masing *frequent itemset* didapatkan dari tingkat kemunculan pola *frequent itemset* tersebut dalam *tree* dan kemudian dihitung nilai *support*-nya.

Tabel 3.12 Hasil *Frequent Itemset* Sampel Data

Frequent itemset	**Frekuensi**	***Support*(A,B)**
5 → 7	2	0.2
1 → 7	2	0.2
1,5 → 7	2	0.2
6 → 5	2	0.2
1 → 5	2	0.2
6 → 1	3	0.3
6 → 3	6	0.6

Proses selanjutnya adalah *frequent itemset* yang ada pada Tabel 3.12 selanjutnya dilakukan perhitungan nilai *confidence*. Hasil perhitungan nilai *confidence frequent itemset* dapat dilihat pada Tabel 3.13. Nilai *confidence* ini akan menentukan apakah *frequent itemset* tersebut dapat dibangkitkan menjadi *rule*. Bila nilai *confidence* lebih besar atau sama dengan nilai *minimum confidence* = 50% maka *frequent itemset* tersebut dapat dibangkitkan untuk menjadi *rule*, sedangkan bila nilai *confidence* tersebut lebih kecil dari nilai *minimum confidence* = 50% maka *frequent itemset* tersebut tidak dapat dibangkitkan menjadi *rule*.

$$Confidence\ (A \rightarrow B) = \frac{\text{Support A} \cap \text{B}}{\text{Support A}}$$

confidence {6} → {3} = 0.6/0.9 = 0.67

confidence {6} → {1} = 0.3/0.9 = 0.33

confidence {6} → {5} = 0.2/0.9 = 0.22

confidence {1} → {5} = 0.2/0.4 = 0.5

confidence {5} → {7} = 0.2/0.3 = 0.67

confidence {1} → {7} = 0.2/0.4 = 0.5

confidence {1,5} → {7} = 0.2/0.2 = 1

Tabel 3.13 *Confidence Frequent Itemset* Sampel Data

Frequent itemset	***Support* (A,B)**	***Support* (A)**	***Confidence***
6 → 3	0.6	0.9	0.67
6 → 1	0.3	0.9	0.33
6 → 5	0.2	0.9	0.22
1 → 5	0,2	0.4	0.5
5 → 7	0.2	0.3	0.67

1 → 7	0.2	0.4	0.5
1,5 → 7	0.2	0.2	1

Berdasarkan Tabel 3.13, *frequent itemset* yang terpilih untuk dibangkitkan menjadi *rule* adalah 6 → 3, 1 → 5, 5 → 7, 1 → 7, dan 1,5 → 7 karena nilai *confidence* masing-masing *frequent itemset* tersebut lebih besar atau sama dengan nilai *minimum confidence* = 50%. *Rule* yang terbentuk dan yang telah memenuhi aturan nilai *minimum confidence* dapat dilihat pada Tabel 3.14.

Tabel 3.14 *Rule* yang Terbentuk

Rules	**Keterangan**
6 → 3	Jika ada jenis ikan Tongkol maka ada jenis ikan Lemuru
1 → 5	Jika ada jenis ikan Cucut maka ada jenis ikan Cakalang
5 → 7	Jika ada jenis ikan Cakalang maka ada jenis ikan Ikan lainnya
1 → 7	Jika ada jenis ikan Cucut maka ada jenis ikan Ikan lainnya
1,5 → 7	Jika ada jenis ikan Cucut dan Cakalang maka ada jenis ikan Ikan lainnya

Selanjutnya *rule* yang terdapat pada Tabel 3.14 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 3.15.

$$Benchmark\ Confidence = \frac{Nc}{N}$$

Nc = jumlah transaksi dengan item dalam *consequent*

N = jumlah transaksi *database*

$$Lift\ Ratio = \frac{Confidence\ (A, C)}{Benchmark\ Confidence\ (A, C)}$$

lift ratio {6} → {3} = 0.67/0.6 = 1.12

lift ratio {1} → {5} = 0.5/0.2 = 2.5

lift ratio {5} → {7} = 0.67/0.2 = 3.35

lift ratio {1} → {7} = 0.5/0.2 = 2.5

lift ratio {1,5} → {7} = 1/0.2 = 5

Tabel 3.15 *Lift Ratio Rules*

Rules	***Confidence***	**Frekuensi item *consequent***	***Benchmark confidence***	***Lift ratio***
6 → 3	0.67	6	0.6	1.12
1 → 5	0.5	2	0.3	2.5
5 → 7	0.67	2	0.2	3.35
1 → 7	0.5	2	0.2	2.5
1,5 → 7	1	2	0.2	5

Berdasarkan hasil perhitungan *lift ratio* pada Tabel 3.15, dapat dilihat bahwa semua *rule* yang dihasilkan memiliki kekuatan yang bagus karena nilai *lift ratio* semua *rule* lebih besar dari 1. *Rule* yang memiliki nilai *lift ratio* paling besar adalah 1,5 → 7 yaitu sebesar 5, dan artinya *rule* 1,5 → 7 memiliki kekuatan asosiasi yang tinggi sehingga dapat menunjukkan adanya manfaat. Selain itu, dapat dilihat juga bahwa nilai *confidence* yang tinggi belum dapat menjadi acuan bahwa *rule* tersebut memiliki kekuatan asosiasi yang tinggi karena dengan *lift ratio*, peluang munculnya kode jenis ikan tersebut bersamaan akan dibandingkan dengan peluang munculnya masing–masing kode jenis ikan tersebut muncul secara independen. Sehingga apabila nilai *confidence* dari suatu *rule* tinggi sedangkan nilai peluang munculnya masing–masing kode jenis ikan tersebut muncul secara independen juga tinggi maka kekuatan dari *rule* tersebut menjadi rendah.

BAB IV
IMPLEMENTASI

Pada bab ini akan dijelaskan implementasi dari seluruh proses yang sudah dirancang pada bab sebelumnya.

4.1 Lingkungan Implementasi

Implementasi merupakan proses transformasi representasi rancangan ke dalam bahasa pemrograman yang dapat dimengerti oleh komputer. Pada bab ini, lingkungan implementasi yang akan dijelaskan meliputi lingkungan implementasi perangkat keras dan perangkat lunak.

4.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan dan pengujian sistem yang menggunakan metode *association rule* dan algoritma *FP Growth* ini adalah sebuah notebook dengan spesifikasi sebagai berikut :

1. Monitor : 13,3”
2. CPU : Intel Ivy Bridge Dual – Core 1.8 GHz
3. *Hard Disk* : 500 GB
4. Memori : 2 GB
5. *Mouse*
6. *Keyboard*

Perangkat keras ini akan difungsikan sebagai tempat perangkat lunak untuk penelitian dijalankan.

4.1.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan sistem kriptografi dan steganografi ini adalah :

1. Sistem operasi *Windows 7™ Ultimate Service Pack 1 64-bit.*
2. *Netbeans IDE 8.0* dan JDK 8 sebagai *programming software development* dalam pembuatan sistem.
3. MySQL dan *mysql connector*.

4.2 Implementasi Perangkat Lunak

4.2.1 Implementasi Program

Berdasarkan rancangan pembuatan sistem pada bab 3, maka pada subbab ini akan dijelaskan implementasi proses – proses tersebut. Secara garis besar proses dikelompokkan menjadi 6 tahap, yaitu tahap data hasil tangkapan, tahap *frequent itemset*, tahap *node (tree)*, tahap *suffix*, tahap *pattern base* dan tahap *rule*.

4.2.1.1 Implementasi Tahap Data Hasil Tangkapan

Proses awal yang dilakukan adalah pembuatan data penangkapan dan ditampilkan di dalam submenu data hasil tangkapan, data tersebut akan digunakan sebagai data transaksi dan akan dilakukan pencarian *frequent itemset*. Implementasi dari proses ini dapat ditunjukkan pada *Source Code* 4.1.

```
  sql = "SELECT * FROM ikan ORDER BY id_ikan";
  rs1 = st1.executeQuery(sql);
  while (rs1.next())
  { ikan.add(rs1.getInt("id_ikan")); }
  sql = "SELECT * FROM daerah ORDER BY id_daerah";
  rs1 = st1.executeQuery(sql);
  while (rs1.next())
  { daerah.add(rs1.getInt("id_daerah")); }

  model = (DefaultTableModel)tbTangkapan.getModel();
  model.setNumRows(0);
  sql = "SELECT * FROM tangkapan t, daerah d, ikan i " +
        "WHERE d.id_daerah=t.id_daerah AND t.id_ikan=i.id_ikan ";
  if (!cbKodeDaerah.getSelectedItem().toString().equals("x"))
  { sql += " AND d.id_daerah=" +
  cbKodeDaerah.getSelectedItem().toString(); }
ql += " ORDER BY t.kwartal, d.id_daerah, i.id_ikan ";
s1 = st1.executeQuery(sql);
hile (rs1.next())
    {
          String str1 = rs1.getString("nm_daerah") + " (" +
          rs1.getString("id_daerah") + ")";
          String str2 = rs1.getString("nm_ikan") + " (" +
          rs1.getString("id_ikan") + ")";
          String str3 = rs1.getString("kwartal");
          String str4 = rs1.getString("jumlah");
          model.addRow(new Object[]{str1,str2,str3,str4});
}
int nRow = 0;
       if (rs1!=null) { rs1.beforeFirst(); rs1.last(); nRow =
       rs1.getRow(); lb1.setText(String.valueOf(nRow)); }
```

***Source Code* 4.1** Tahap Data Hasil Tangkapan

Pada tahap data hasil tangkapan, langkah pertama adalah melakukan *query* SQL untuk mendapatkan data id_ikan dan data id_daerah yang ada di *database*.

Kemudian disimpan ke dalam tbTangkapan yang berisi data id_daerah, id_ikan, kwartal dan jumlah.

4.2.1.2 Implementasi Tahap *Frequent Itemset*

Tahap pencarian *frequent itemset*, dimulai dengan memasukan masukan nilai *minimum support*, nilai *minimum confidence* dan parameter daerah ke sistem. Kemudian sistem melakukan seleksi data hasil tangkapan dengan aturan jenis ikan dengan jumlah ikan lebih dari 700, maka akan digunakan dalam pencarian *frequent itemset*. Hasil penyeleksian tersebut dihitung frekuensinya dan dibandingkan dengan nilai *minimum support* yang sudah ditentukan. Jika kurang dari nilai *minimum support*, maka kode jenis ikan tersebut dihapus dan tidak ditampilkan ke dalam submenu *frequent itemset*.

```
  int nTrans = 4; //karena ada 4 kuartal (kuartal=transaksi)
  minSup = (int)Math.round((minSup/100.0)*nTrans);
nConf = minConf/100.0;

  st1.execute("DELETE FROM frequentitemset");
  sql = "SELECT * FROM tangkapan " +
"WHERE jumlah>=700"; //500: batas banyak/sedikit
  if (!cbKodeDaerah.getSelectedItem().toString().equals("x"))
  { sql += " AND id_daerah=" +
  cbKodeDaerah.getSelectedItem().toString(); }
  sql += " ORDER BY id_ikan";
  rs1 = st1.executeQuery(sql);
  String id_ikan = "";
  while (rs1.next())
     {
String str = rs1.getString("id_ikan");
if (!id_ikan.equals(str))
    {
        id_ikan = str;
               st2.execute("INSERT INTO frequentitemset SET
               id_ikan="+ id_ikan +", support=1");
    }
        else if (id_ikan.equals(str))
    {
               rs2 = st2.executeQuery("SELECT support FROM
               frequentitemset WHERE id_ikan="+ id_ikan);
        rs2.first();
        int sup = rs2.getInt("support");
        sup++;
               st2.execute("UPDATE frequentitemset SET support=" +
               sup + " WHERE id_ikan="+ id_ikan);
     }
      }

/hapus yang tidak memenuhi minSup
  st1.execute("DELETE FROM frequentitemset WHERE
  support<"+minSup);
```

```
/tampilkan pada tabel frequent
odel = (DefaultTableModel)tbFrequent.getModel();
odel.setNumRows(0);
  sql = "SELECT * FROM frequentitemset ORDER BY support DESC,
  id_ikan ASC ";
s1 = st1.executeQuery(sql);
  while (rs1.next())
       {
         String str1 = rs1.getString("id_ikan");
         String str2 = rs1.getString("support");
         model.addRow(new Object[]{str1,str2});
       }
       nRow = 0;
           if (rs1!=null) { rs1.beforeFirst(); rs1.last(); nRow =
           rs1.getRow(); lb2.setText(String.valueOf(nRow)); }
```

***Source Code* 4.2** Tahap *Frequent Itemset*

4.2.1.3 Implementasi Tahap *Node* (*Tree*)

Tahap *node (tree)* adalah tahap pembentukan *FP Tree* setelah didapatkan data *frequent itemset*. Dalam submenu *node (tree)* berisi kolom nama, *parent*, level dan *support*. Kolom-kolom yang terdapat di submenu *node (tree)* merupakan proses pembentukan *FP Tree*. Kolom nama adalah kolom yang berisi kode jenis ikan yang *frequent*, nama diurutkan secara *descending* menurut besar nilai *support*-nya. Kolom *parrent* digunakan untuk inisialisasi simpul awal yaitu simpul sebagai *parent*, untuk *root* awal mempunyai nilai *null* maka diinisialisasikan dengan kolom yang kosong. Kolom level bertujuan untuk memberi tanda bahwa lintasan tersebut berada pada level berapa. Kolom *support* berisi nilai *support* dari masing-masing jenis ikan.

```
 //pembentukan tree
 int kwartal=0, kat, kwt;
 int lv=0;
 String par="";
 st1.execute("DELETE FROM node");
 sql = "SELECT * FROM tangkapan t, frequentitemset f " +
"WHERE t.id_ikan=f.id_ikan AND t.jumlah>=700 ";
 if (!cbKodeDaerah.getSelectedItem().toString().equals("x"))
  { sql += " AND t.id_daerah=" +
  cbKodeDaerah.getSelectedItem().toString(); }
 sql += " ORDER BY t.kwartal ASC, f.support DESC, f.id_ikan ASC ";
 rs1 = st1.executeQuery(sql);
 while (rs1.next())
     {
 kat = rs1.getInt("id_ikan");
 kwt = rs1.getInt("kwartal");
 String str2 = rs1.getString("support");
 if (kwt != kwartal)
```

```
 {
         kwartal = kwt;
         lv = 1; par = "";
 }
         else { lv++; }

         sql = "SELECT * FROM node WHERE nama=" + kat + " AND
         level=" + lv + " AND parent='" + par + "'";
  rs2 = st2.executeQuery(sql);
         nRow = 0; if (rs2!=null) { rs2.beforeFirst(); rs2.last();
         nRow = rs2.getRow(); }
          if (nRow==0)
         { st2.execute("INSERT INTO node SET nama="+ kat +",
         parent='"+ par + "', level=" + lv + ", support=1"); }
else
{
      rs2 = st2.executeQuery("SELECT support FROM node WHERE
      nama="+ kat +" AND parent='"+ par + "' AND level=" + lv);
      rs2.first();
      int sup = rs2.getInt("support");
      sup++;
       st2.execute("UPDATE node SET support=" + sup + " WHERE
      nama="+ kat +" AND parent='"+ par + "' AND level=" + lv);
}
if (lv==1) { par = Integer.toString(kat); }
else { par = par + "-" + kat; }
}

//tampilkan pada tabel node
model = (DefaultTableModel)tbNode.getModel();
model.setNumRows(0);
sql = "SELECT * FROM node ORDER BY level";
rs1 = st1.executeQuery(sql);
while (rs1.next())
      {
        String str1 = rs1.getString("nama");
        String str2 = rs1.getString("parent");
        String str3 = rs1.getString("level");
        String str4 = rs1.getString("support");
        model.addRow(new Object[]{str1,str2,str3,str4});
      }
nRow = 0;
if (rs1!=null) { rs1.beforeFirst(); rs1.last(); nRow =
rs1.getRow(); lb3.setText(String.valueOf(nRow)); }
```

***Source Code* 4.3** Tahap *Node* (*Tree*)

4.2.1.4 Implementasi Tahap *Suffix*

Tahap pencarian *suffix* dilakukan dengan cara melakukan penambangan *frequent itemset* yaitu menambang id_ikan yang ada di tabel *frequentitemset*, kemudian mengkoleksi *node* dari id_ikan tersebut. Id_ikan tersebut dihitung nilai *support*-nya dan dilakukan pembentukan *tree* pada kolom *parrent*. Setiap *parrent*

dengan *suffix* dan nama tertentu diberikan inisialisasi level untuk memudahkan pembacaan.

```
//penambangan frequentitemset
String chk,pr;
String [] split;
int nS,nS1,tp;
st1.execute("DELETE FROM suffix");
sql = "SELECT * FROM frequentitemset ORDER BY id_ikan ASC";
rs1 = st1.executeQuery(sql);
while (rs1.next())
      {
        kat = rs1.getInt("id_ikan");
        sql = "SELECT * FROM node WHERE nama=" + kat;
        rs2 = st2.executeQuery(sql);
        while (rs2.next())
        {
          nS = rs2.getInt("support");
          par = rs2.getString("parent");
          chk = par;
          if (!chk.equals(""))
          {
            split = chk.split("-");
            lv = split.length;
                  tp = Integer.parseInt(split[split.length-1]);
                  // yang terakhir
                  if (lv>1) //copy karakter2 sebelum karakter
                  akhir
            {
              pr = "";
              for (int i=0; i<split.length-1; i++)
              { if (i>0) { pr += "-"; } pr += split[i]; }
            }
            else { pr = ""; }

sql = "SELECT * FROM suffix WHERE sfx="+kat+" AND nama="+tp+" AND
level="+lv+" AND parent='"+pr+"'";
rs3 = st3.executeQuery(sql);
nRow = 0; if (rs3!=null) { rs3.beforeFirst(); rs3.last(); nRow =
rs3.getRow();
if (nRow==0)
            { st3.execute("INSERT INTO suffix SET sfx="+kat+",
              nama="+tp+", level="+lv+", parent='"+pr+"',
              support="+nS); }
            else
            {
                  rs3 = st3.executeQuery("SELECT support FROM
                  suffix WHERE sfx="+kat+" AND nama="+tp+" AND
                  level="+lv+" AND parent='"+pr+"'");
             rs3.first();
             int sup = rs3.getInt("support");
             nS1 = sup+nS;
                  st2.execute("UPDATE suffix SET support="+nS1+"
                  WHERE sfx="+kat+" AND nama="+tp+" AND
                  level="+lv+" AND parent='"+pr+"'");
            }
            chk = pr;
```

```
            }
          }
        }
//tampilkan pada tabel suffix
        model = (DefaultTableModel)tbSuffix.getModel();
        model.setNumRows(0);
        sql = "SELECT * FROM suffix ORDER BY level,nama";
        rs1 = st1.executeQuery(sql);
        while (rs1.next())
        {
          String str1 = rs1.getString("sfx");
          String str2 = rs1.getString("nama");
          String str3 = rs1.getString("level");
          String str4 = rs1.getString("parent");
          String str5 = rs1.getString("support");
          model.addRow(new Object[]{str1,str2,str3,str4,str5});
        }
        nRow = 0;
if (rs1!=null) { rs1.beforeFirst(); rs1.last(); nRow =
rs1.getRow(); lb4.setText(String.valueOf(nRow)); }
```

***Source Code* 4.4** Tahap *Suffix*

4.2.1.5 Implementasi Tahap *Pattern Base*

Setelah melewati tahap pencarian *suffix*, langkah selanjutnya adalah melakukan proses *conditional pattern base* dengan mengumpulkan *suffix* dengan lintasannya sehingga didapatkan *suffix pattern*.

```
//MENDAPATKAN FREQUENT ITEMSET DARI MASING-MASING SUFFIX PATTERN
        int nA=0, n;
        boolean mulai, ketemu=false;
        String nama, str;
        int sup;
        ArrayList<ArrayList<String>> tNode;
        ArrayList<ArrayList<Integer>> tSup;
        ArrayList<Integer> ary = new ArrayList<>();

        st1.execute("DELETE FROM patternbase");
        st1.execute("DELETE FROM rule");
        sql = "SELECT * FROM frequentitemset ORDER BY id_ikan";
        rs1 = st1.executeQuery(sql);
        while (rs1.next())
        {
          tNode = new ArrayList<ArrayList<String>>();
          tSup = new ArrayList<ArrayList<Integer>>();
                kat = rs1.getInt("id_ikan");
```

***Source Code* 4.5** Tahap *Pattern Base*

4.2.1.6 Implementasi Tahap *Rule*

Tahap *rule* dilakukan setelah melewati penggalian data dengan algoritma *FP Growth*. Dari hasil *conditional pattern base* dan *conditional FP Tree* didapatkan lintasan yang membentuk pola atau *rule*. Setelah *rule* terbentuk, maka selanjutnya dihitung nilai *support rule* tersebut. Kemudian dilakukan perhitungan *confidence* dan dibandingkan dengan nilai *minimum confidence*. Tahap terakhir yaitu menghitung nilai *lift ratio* dengan mengambil nilai *confidence* dan dibagi dengan nilai *benchmark confidence*.

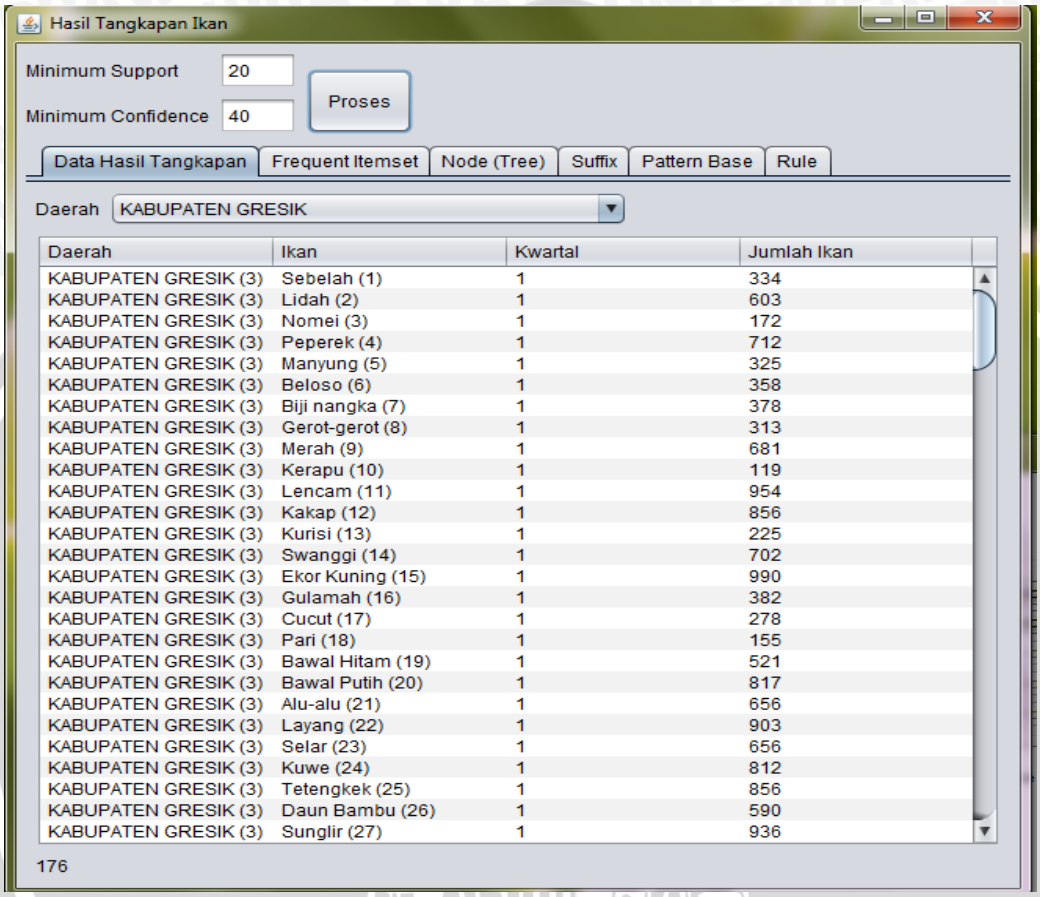
```
int a,b,d,e,f,idx;
double c,lift;
String st="",tStr="";
int tSp=0;
ArrayList<Integer> tmp = new ArrayList<>();
a = 0;
sql = "SELECT * FROM patternbase";
rs1 = st1.executeQuery(sql);
       nRow = 0; if (rs1!=null) { rs1.beforeFirst(); rs1.last();
       nRow = rs1.getRow(); }
      if (nRow>0)
      {
        txMemo.removeAll();
        rs1.beforeFirst();
        while (rs1.next())
        {
          st = rs1.getString("lintasan");
          split = st.split("-");
          if (split.length==2)
          {
            a = rs1.getInt("support");
                  sql = "SELECT support FROM patternbase WHERE
                  lintasan='"+split[0]+"'";
            rs3 = st3.executeQuery(sql);
            rs3.first(); b = rs3.getInt("support");
            c = (double)a/(double)b;
            if (c>=minConf)
            {
                    sql = "SELECT support FROM patternbase WHERE
                    lintasan='"+st+"'";
              rs4 = st4.executeQuery(sql);
              rs4.first();
              d = rs4.getInt("support");
              lift = c/(((double)d/(double)nTrans));

              //txMemo.append(split[0]+" - "+split[1]+"
Confidence:"+c+" Lift:"+lift+"\n");
              sql = "INSERT INTO rule SET "+
                    "jika='"+split[0]+"', maka='"+split[1]+"', "+
                    "sxy="+a+", sx="+b+", conf="+c+", lift="+lift;
              st4.execute(sql);
            }
          }
        }
      }
```

```
      //tampilkan pada tabel rule
      model = (DefaultTableModel)tbRule.getModel();
      model.setNumRows(0);
      sql = "SELECT * FROM rule ORDER BY jika,maka";
      rs1 = st1.executeQuery(sql);
      while (rs1.next())
      {
        String str1 = rs1.getString("jika");
        String str2 = rs1.getString("maka");
        String str3 = rs1.getString("sxy");
        String str4 = rs1.getString("sx");
        String str5 = String.format("%1.2f",
rs1.getDouble("conf"));
        String str6 = String.format("%1.2f",
rs1.getDouble("lift"));
        model.addRow(new Object[]{str1,str2,str3,str4,str5,str6});
      }
      nRow = 0;
      if (rs1!=null) { rs1.beforeFirst(); rs1.last(); nRow =
rs1.getRow(); lb6.setText(String.valueOf(nRow)); }
      rs1.close();
      st1.close();
      st2.close();
      st3.close();
      conn.close();
    }
    catch (Exception e) {
      JOptionPane.showMessageDialog(this, "Database Error:
"+e.getMessage());
    }
  }
```

***Source Code* 4.6** Tahap *Rule*

4.2.2 Implementasi Antarmuka Pembuatan Data Penangkapan

Berdasarkan rancangan antar muka yang dikemukakan pada Bab III, maka dihasilkan antar muka yang ditunjukkan pada Gambar 4.1.

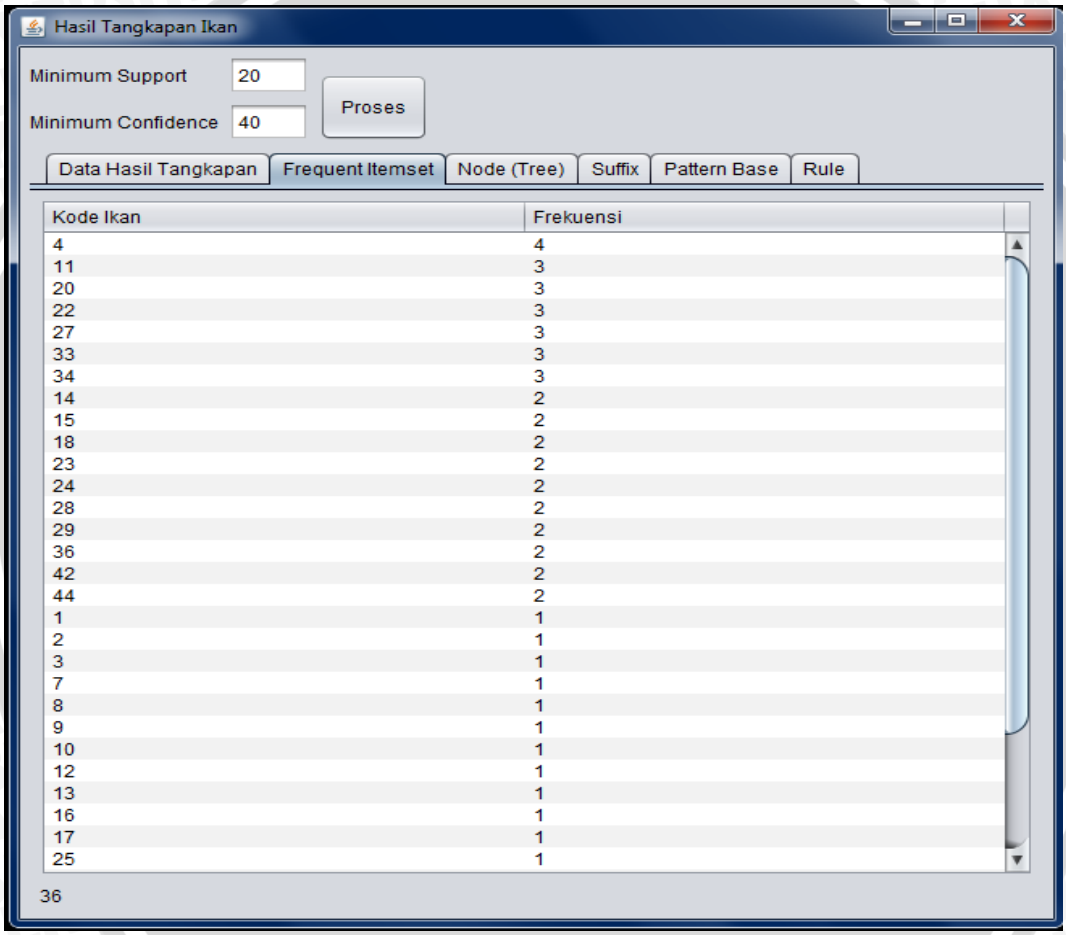


Gambar 4.1 Antarmuka Awal dengan Sudah Diberikan Masukan

Pada Gambar 4.1 dapat dilihat contoh antarmuka yang sudah diberikan masukan nilai *minimum support*, nilai *minimum confidence* dan parameter daerah pilihan. Contoh pada gambar tersebut dengan memberikan masukan dan menekan tombol proses, maka akan muncul data hasil tangkapan dan jumlah data transaksi tersebut. Data hasil tangkapan berisi tabel yang terdiri dari kolom daerah, ikan, kuartal dan jumlah ikan.

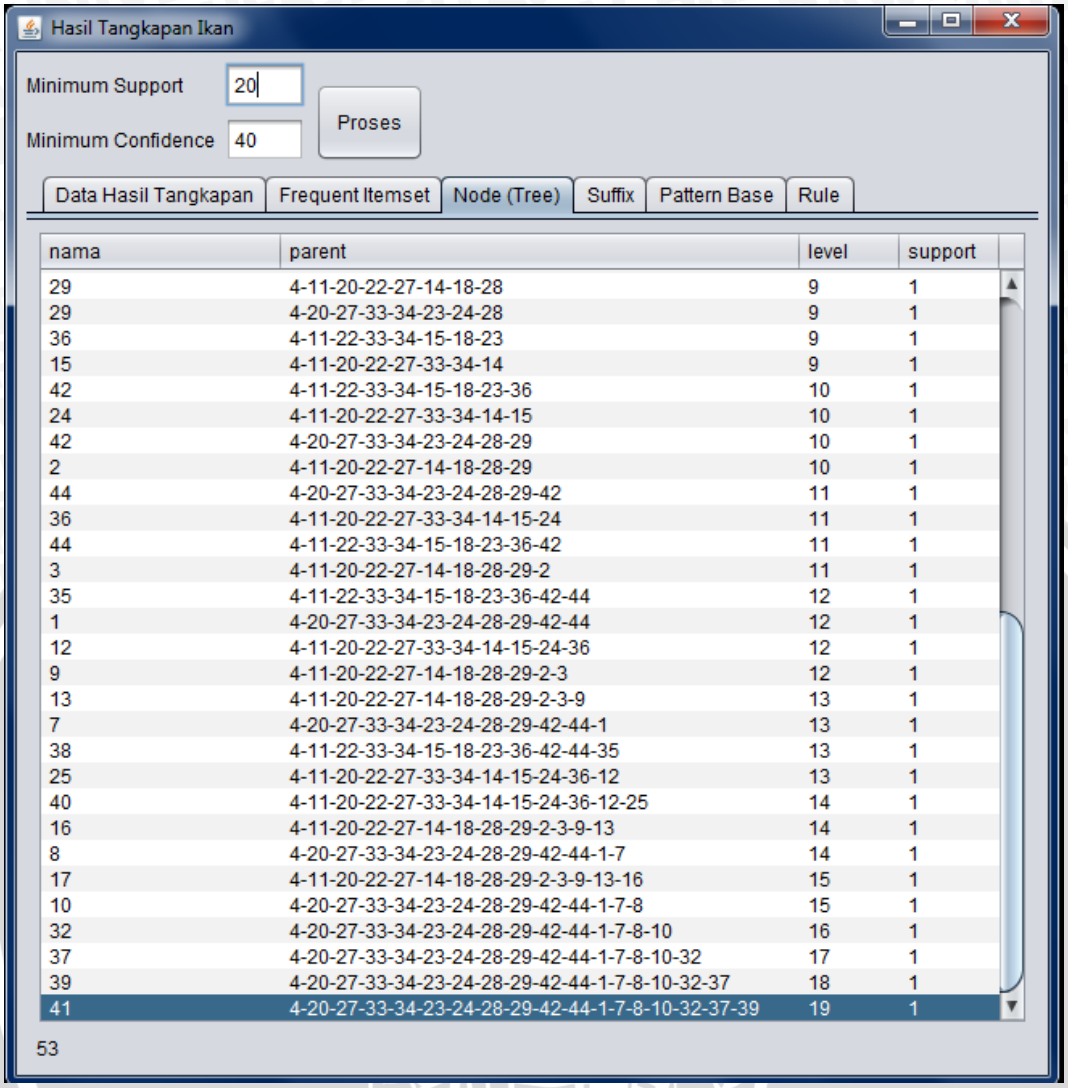
4.2.3 Implementasi Antarmuka Penggalian Data

Berdasarkan rancangan antar muka kelanjutan dari Gambar 4.1, dalam antarmuka penggalian data terdiri dari submenu *frequent itemset*, submenu *node (tree)*, submenu *suffix*, submenu *conditional Fp Tree* dan submenu *rule*. Antarmuka submenu *frequent itemset* dapat dilihat pada Gambar 4.2. submenu *frequent itemset* berisi tabel yang terdiri dari kolom kode ikan dan frekuensi.



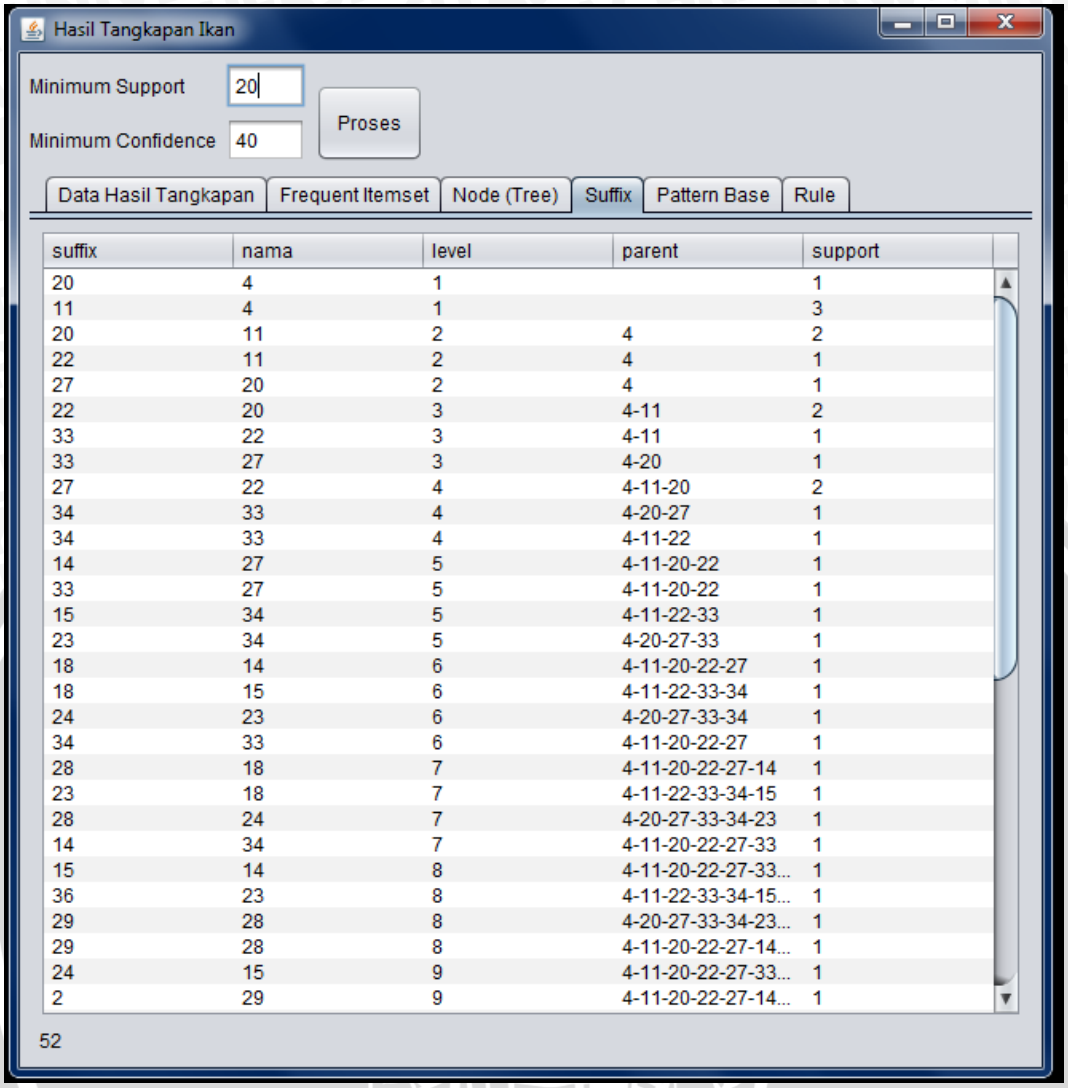
Gambar 4.2 Antarmuka Submenu *Frequent Itemset*

Antarmuka submenu *node (tree)* terdiri dari kolom nama, *parrent*, level dan *support* dapat dilihat pada Gambar 4.3. Kolom nama yaitu kolom yang berisi id_ikan, sedangkan kolom *parrent* berisi pembentukan *tree*.

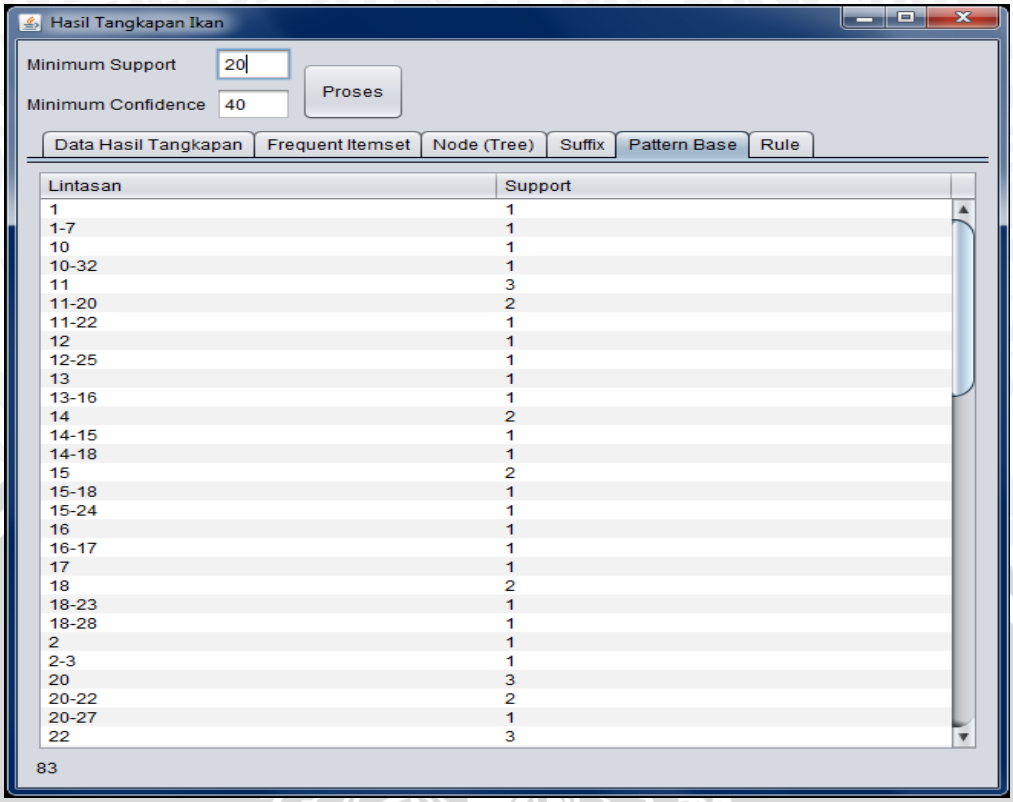


Gambar 4.3 Antarmuka Submenu *Node (tree)*

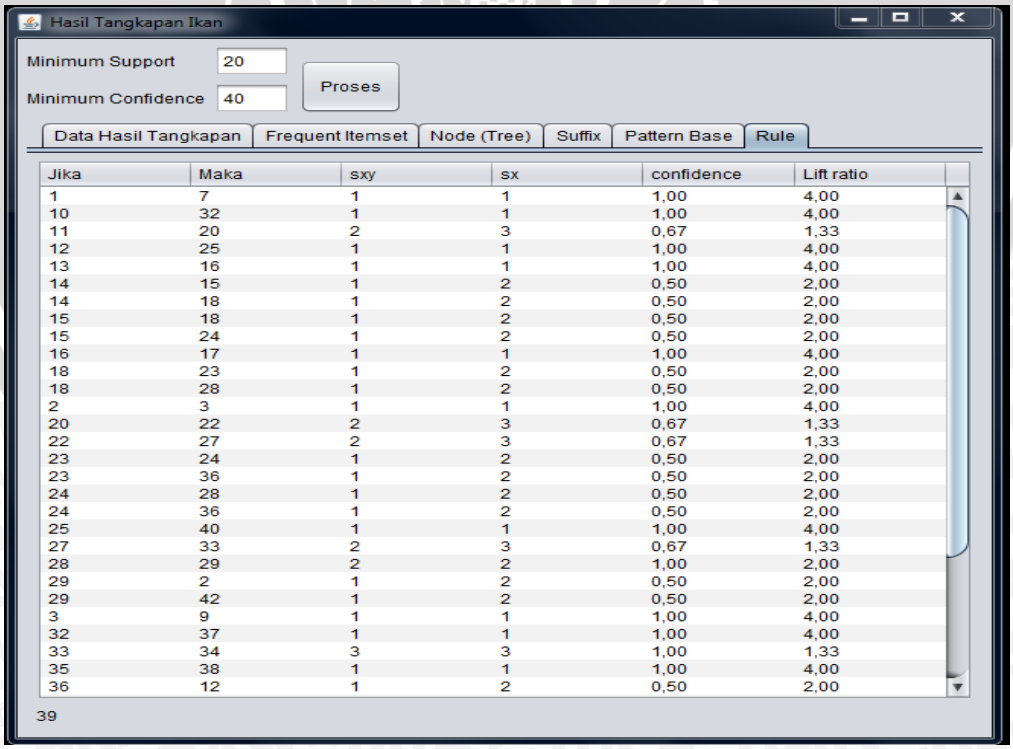
Antarmuka submenu *suffix* dapat dilihat pada Gambar 4.4 terdiri dari kolom suffix, nama, level, parrent dan support. Antarmuka submenu *pattern base* dapat dilihat pada Gambar 4.5 terdiri dari kolom lintasan dan kolom nilai *support*. Antarmuka submenu *rule* dapat dilihat pada Gambar 4.6 terdiri dari kolom jika, maka, sxy, sx, *confidence* dan *lift ratio*..



Gambar 4.4 Antarmuka Submenu *Suffix*



Gambar 4.5 Antarmuka Submenu *Pattern Base*



Gambar 4.6 Antarmuka Submenu *Rule*

BAB V
PENGUJIAN DAN ANALISIS

Pada bab ini akan dijelaskan hasil pengujian dan analisis terhadap hasil dari pengujian yang dilakukan menggunakan sistem yang telah dibuat.

5.1 Pengujian

Pengujian pertama dilakukan untuk mengetahui pengaruh nilai *minimum support* dan nilai *minimum confidence* terhadap jumlah *rule* yang dihasilkan. Pengujian data dilakukan dengan masukan nilai *minimum support* antara 10% sampai dengan 80% dan nilai *minimum confidence* antara 10% sampai dengan 80%, sehingga diketahui jumlah *rule* yang dihasilkan berdasarkan masukan yang diujicobakan pada sistem. Pada pengujian ini dapat dilihat pada Tabel 5.1 dengan mengambil sampel data pada daerah Kabupaten Tuban, sedangkan untuk hasil uji coba keseluruhan daerah dapat dilihat pada lampiran.

Tabel 5.1 Tabel Uji Coba Pengaruh Nilai *Minimum Support* dan Nilai *Minimum Confidence* terhadap Jumlah *Rule* yang Dihasilkan

Daerah	Nilai *Minimum Support* (%)	Nilai *Minimum Confidence* (%)	Jumlah *Rule* yang Dihasilkan
Tuban	10	10	49
	10	20	49
	10	30	48
	10	40	46
	10	50	46
	10	60	17
	10	70	15
	10	80	14
	20	10	49
	20	20	49
	20	30	48
	20	40	46
	20	50	46
	20	60	17

	20	70	15
	20	80	14
	30	10	49
	30	20	49
	30	30	48
	30	40	46
	30	50	46
	30	60	17
	30	70	15
	30	80	14
	40	10	9
	40	20	9
	40	30	9
	40	40	9
	40	50	9
	40	60	9
	40	70	7
	40	80	6
	50	10	9
	50	20	9
	50	30	9
	50	40	9
	50	50	9
	50	60	9
	50	70	7
	50	80	6
	60	10	9
	60	20	9
	60	30	9
	60	40	9
	60	50	9
	60	60	9
	60	70	7
	60	80	6

	70	10	3
	70	20	3
	70	30	3
	70	40	3
	70	50	3
	70	60	3
	70	70	3
	70	80	2
	80	10	3
	80	20	3
	80	30	3
	80	40	3
	80	50	3
	80	60	3
	80	70	3
	80	80	2

Pengujian kedua dilakukan untuk mengetahui tingkat kekuatan (*lift ratio*) terhadap *rule* yang dihasilkan. Pada pengujian ini, nilai *mimnimum support* yang digunakan adalah sebesar 40%, nilai *minimum confidence* yang digunakan adalah 50% dan uji coba ini dilakukan pada semua daerah penangkapan. Digunakannya batasan nilai tersebut karena berdasarkan pengujian pada sistem, dengan batasan nilai tersebut telah dihasilkan *rule* yang memiliki nilai *confidence* dan nilai *lift ratio* yang bagus sehingga dapat dijadikan sebagai *rule* yang bermanfaat. Untuk contoh sampel data yang digunakan adalah data daerah Pamekasan, Bangkalan dan Muncar, sedangkan untuk hasil pengujian kedua secara lengkap untuk daerah yang lain dapat dilihat pada lampiran. Hasil uji tingkat kekuatan (*lift ratio*) terhadap *rule* yang dihasilkan dapat dilihat pada Tabel 5.2.

Tabel 5.2 Hasil Uji Coba Tingkat Kekuatan (*Lift Ratio*) terhadap *Rule* yang Dihasilkan

Daerah	***Rule* yang Dihasilkan**	**Nilai *Confidence***	**Nilai *Lift Ratio***
Pamekasan	10 → 12	1.00	2.00

	12 → 13	1.00	2.00
	19 → 33	0.67	1.33
	32 → 35	1.00	2.00
	41 → 42	1.00	2.00
		Rata-rata Nilai Lift Ratio	**1.86**
Bangkalan	11 → 8	0.75	1.00
	24 → 41	0.67	1.33
	41 → 10	0.67	1.33
	8 → 24	0.67	1.33
		Rata-rata Nilai Lift Ratio	**1.24**
Muncar	15 → 24	0.75	1.00
	24 → 38	0.67	1.33
	38 → 43	0.67	1.33
	43 → 2	0.67	1.33
		Rata-rata Nilai Lift Ratio	**1.24**

Berdasarkan Tabel 5.2 dapat dilihat *rule* yang terbentuk pada daerah Pamekasan, Bangkalan dan Muncar, keterangan *rule* hasil uji coba dapat dilihat pada Tabel 5.3. *Rule* yang dihasilkan pada Tabel 5.2 merupakan *rule* yang memenuhi aturan lebih besar dari nilai *minimum support* = 40% dan nilai *minimum confidence* = 50%. Dengan mengambil contoh sampel data daerah didapatkan informasi bahwa rata–rata nilai *lift ratio rule* yang dihasilkan pada daerah Pamekasan adalah 1.86, sedangkan rata-rata nilai *lift ratio rule* yang dihasilkan pada daerah Bangkalan dan Muncar adalah 1.24.

Tabel 5.3 Keterangan *Rule* Hasil Uji Kekuatan dari Nilai *Lift Ratio*

Rule	**Keterangan**
PAMEKASAN	
10 → 12	Jika ada jenis ikan Kerapu maka ada jenis ikan Kakap
12 → 13	Jika ada jenis ikan Kakap maka ada jenis ikan Kurisi
19 → 33	Jika ada jenis ikan Bawal Hitam maka ada jenis ikan Japuh

32 → 35	Jika ada jenis ikan Teri maka ada jenis ikan Lemuru
41 → 42	Jika ada jenis ikan Layur maka ada jenis ikan Tuna
BANGKALAN	
11 → 8	Jika ada jenis ikan Lencam maka ada jenis ikan Gerot-gerot
24 → 41	Jika ada jenis ikan Kuwe maka ada jenis ikan Layur
41 → 10	Jika ada jenis ikan Layur maka ada jenis ikan Kerapu
8 → 24	Jika ada jenis ikan Gerot-gerot maka ada jenis ikan Kuwe
MUNCAR	
15 → 24	Jika ada jenis ikan Ekor Kuning maka ada jenis ikan Kuwe
24 → 38	Jika ada jenis ikan Kuwe maka ada jenis ikan Kembung
38 → 43	Jika ada jenis ikan Kembung maka ada jenis ikan Cakalang
43 → 2	Jika ada jenis ikan Cakalang maka ada jenis ikan Tongkol

Tabel 5.3 merupakan tabel yang berisi keterangan *rule* hasil uji coba pada Tabel 5.2 yaitu pola jenis ikan yang muncul pada daerah Pamekasan, Bangkalan dan Muncar. *Rule* 10 → 12 dapat dibaca dengan keterangan jika ada jenis ikan Kerapu maka ada jenis ikan Kakap. Pembacaan *rule* dimulai dengan kode jenis ikan pertama kemudian dilanjutkan kode jenis ikan kedua. Hal ini mengartikan bahwa di daerah Pamekasan terdapat beberapa kemunculan ikan dengan 5 *rule* yaitu 10 → 12, 12 → 13, 19 → 33, 32 → 35, dan 41 → 42. Daerah Bangkalan terdapat beberapa kemunculan ikan dengan 4 *rule* yaitu 11 → 8, 24 → 41, 41 → 10 dan 8 → 24. Sedangkan daerah Muncar terdapat kemunculan ikan dengan 4 *rule* yaitu 15 → 24, 24 → 38, 38 → 43 dan 43 → 2.

5.2 Analisa

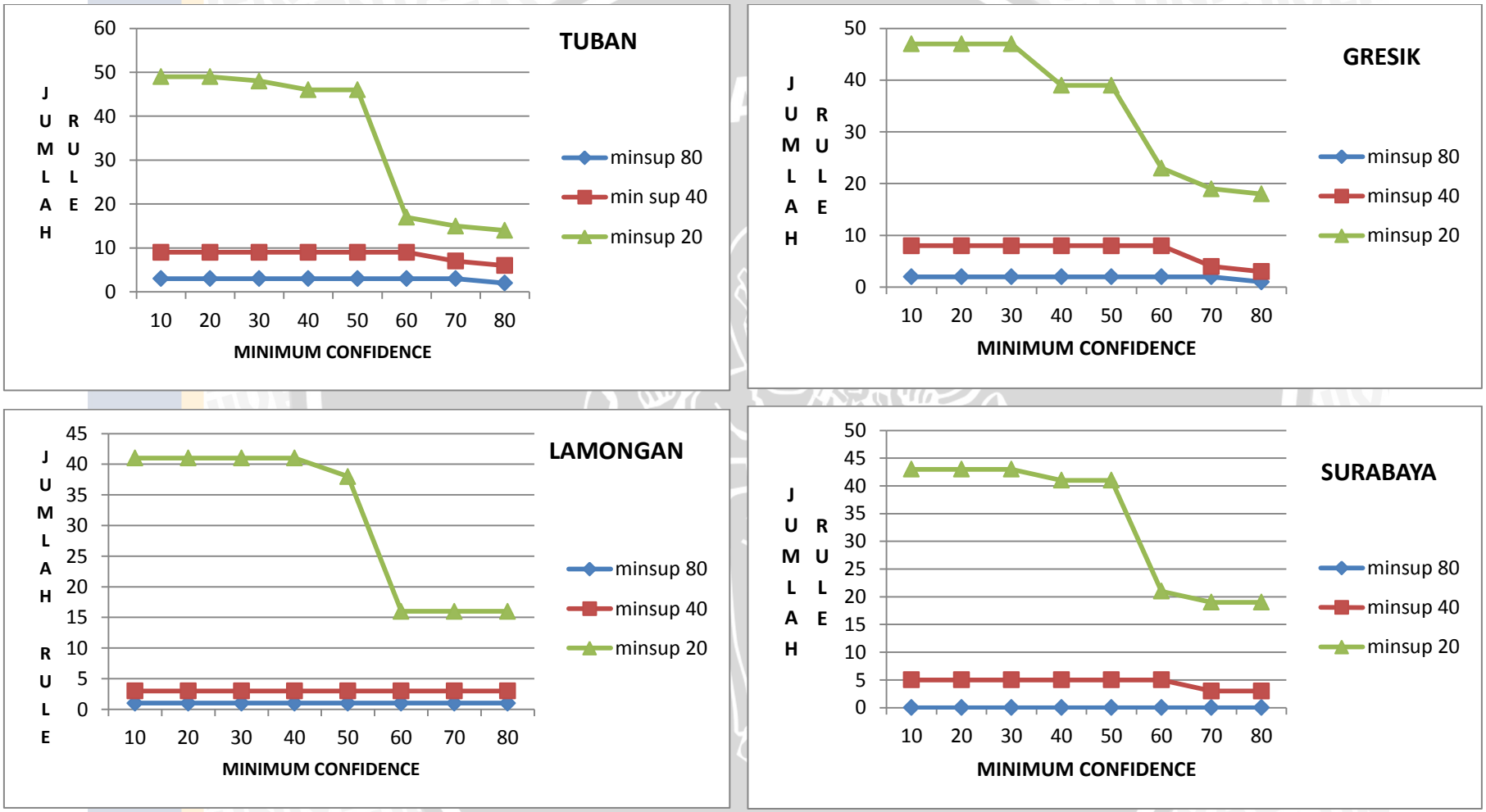
Hasil pengujian pengaruh nilai *minimum support* dan nilai *minimum confidence* terhadap jumlah *rule* yang dihasilkan, didapatkan jumlah *rule* terbanyak yang dihasilkan pada daerah Tuban adalah 49 *rule* dengan uji coba pengaruh nilai *minimum support* sebesar 10% sampai dengan 30% dan nilai *minimum confidence* sebesar antara 10% sampai 20%. Sedangkan jumlah *rule* terkecil yang dihasilkan pada daerah Tuban adalah 2 *rule* dengan *minimum support* 70% sampai 80% dan *minimum confidence* sebesar 80%. Dapat lihat dari

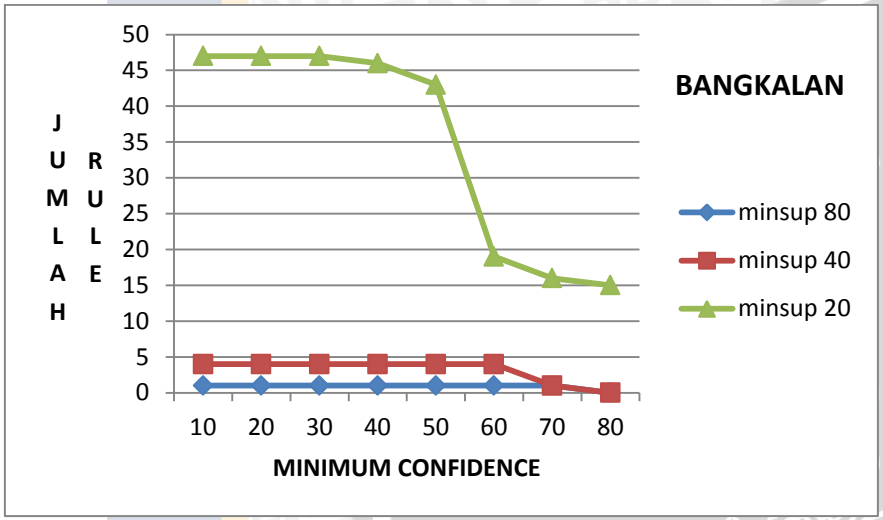
Tabel 5.1 bahwa semakin besar nilai *minimum support* dan nilai *minimum confidence* maka jumlah *rule* yang dihasilkan semakin sedikit.

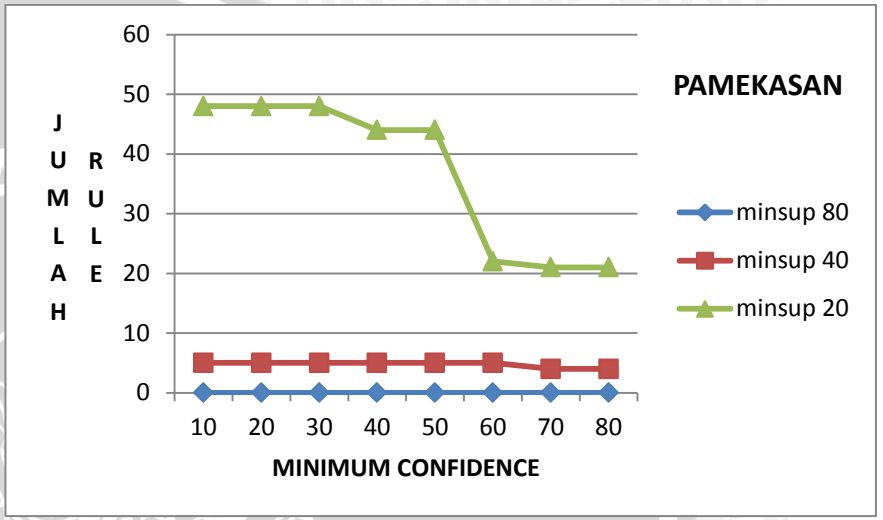
Untuk hasil pengujian daerah yang lainnya dapat dilihat di lampiran dengan pengujian yang sama yaitu dengan nilai *minimum support* antara 10% sampai 80% dan nilai *minimum confidence* antara 10% sampai 80%. Berdasarkan pengujian yang dilakukan pada semua daerah, didapatkan jumlah *rule* tertinggi sebesar 53 *rule* pada daerah Sumenep dengan *minimum support* antara 10% sampai 30% dan *minimum confidence* antara 10% sampai 20%. Sedangkan pada daerah Surabaya, Bangkalan, Pamekasan, Kab. Pasuruan, Kab. Probolinggo, Banyuwangi, Jember, Lumajang, Tulungagung, Pacitan dan Muncar terdapat pengujian yang tidak menghasilkan *rule*. Hasil itu terjadi pada pengujian dengan nilai *minimum support* dan nilai *minimum confidence* antara 70% sampai 80%.

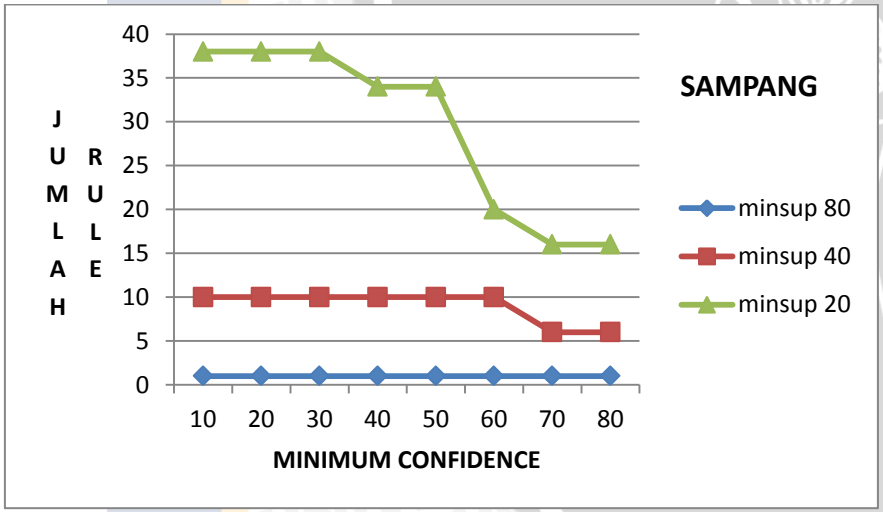
Pengaruh nilai *minimum support* dan nilai *minimum confidence* terhadap jumlah *rule* yang dihasilkan berdasarkan hasil pengujian adalah berbanding terbalik. Hal ini berarti semakin tinggi nilai *minimum support* dan nilai *minimum confidence* yang digunakan maka semakin sedikit jumlah *rule* yang dihasilkan seperti yang digambarkan pada Gambar 5.1. Berbanding terbaliknya nilai *minimum support* dan nilai *minimum confidence* terhadap jumlah *rule* tersebut dikarenakan semakin tinggi nilai *minimum support* yang digunakan berarti semakin tinggi nilai batas *support* yang harus dicapai kandidat *frequent itemset* untuk menjadi *frequent itemset* sehingga jumlah *frequent itemset* semakin sedikit dan bila semakin tinggi nilai *minimum confidence* yang digunakan maka semakin tinggi nilai batas *confidence* yang harus dicapai *frequent itemset*, yang merupakan kandidat *rule* untuk menjadi *rule* sehingga semakin sedikit jumlah *rule* yang dihasilkan.

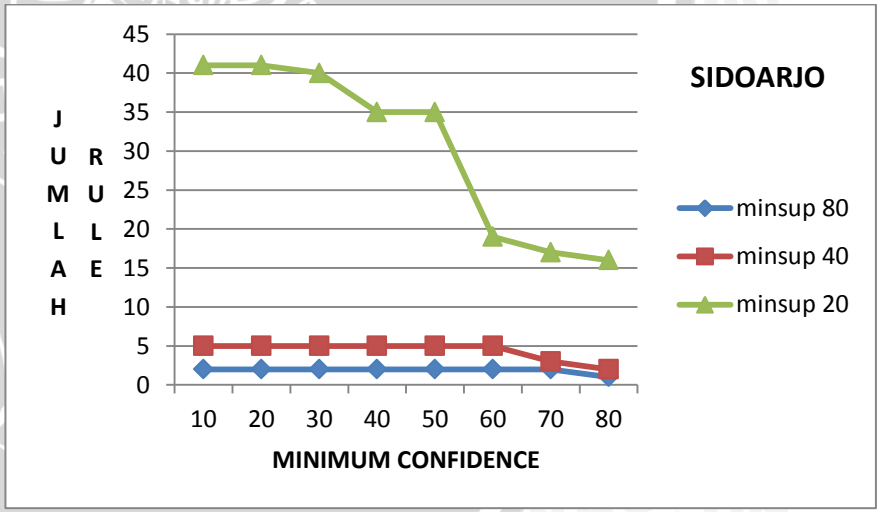
Gambar 5.1 Grafik Pengaruh Nilai Minimum *Support* dan Nilai Minimum *Confidence* terhadap Jumlah *Rule*

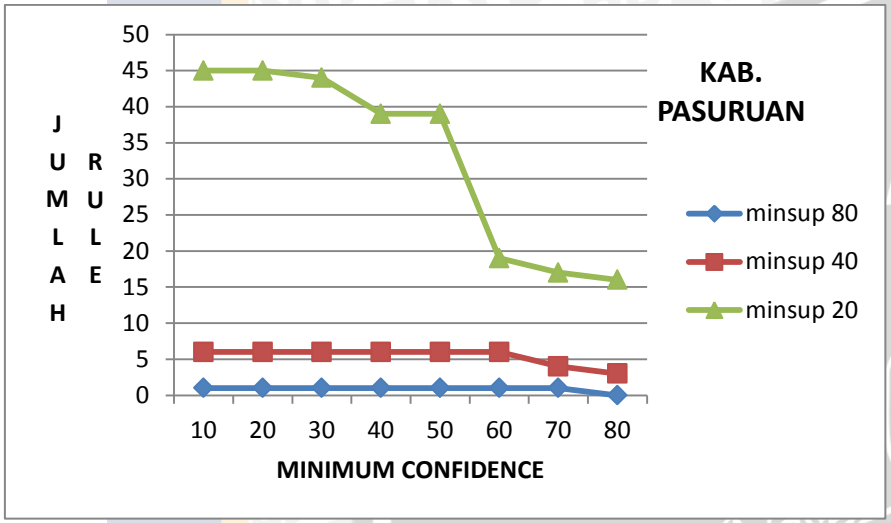


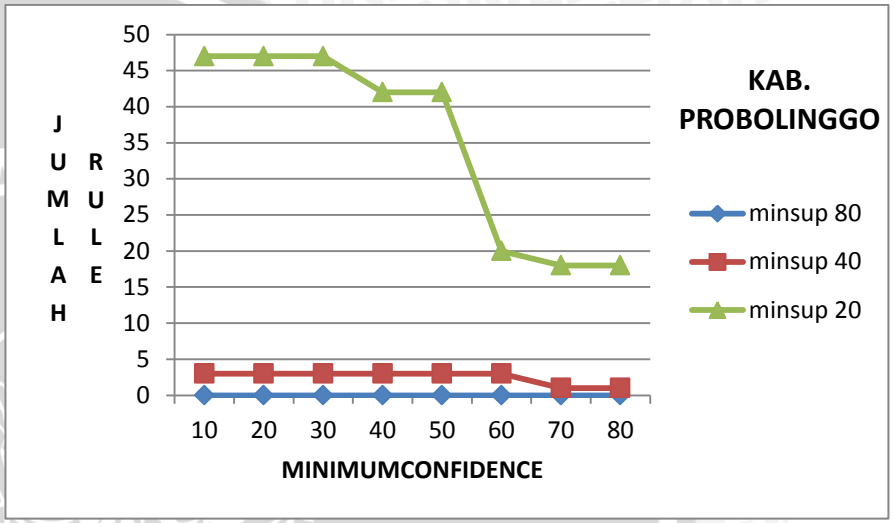

BANGKALAN
JUMLAH RULE
50
45
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

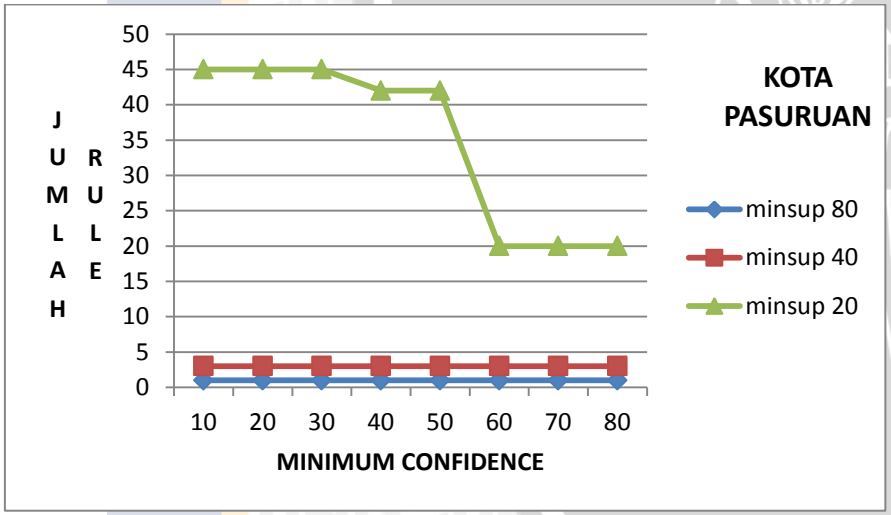

PAMEKASAN
JUMLAH RULE
60
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

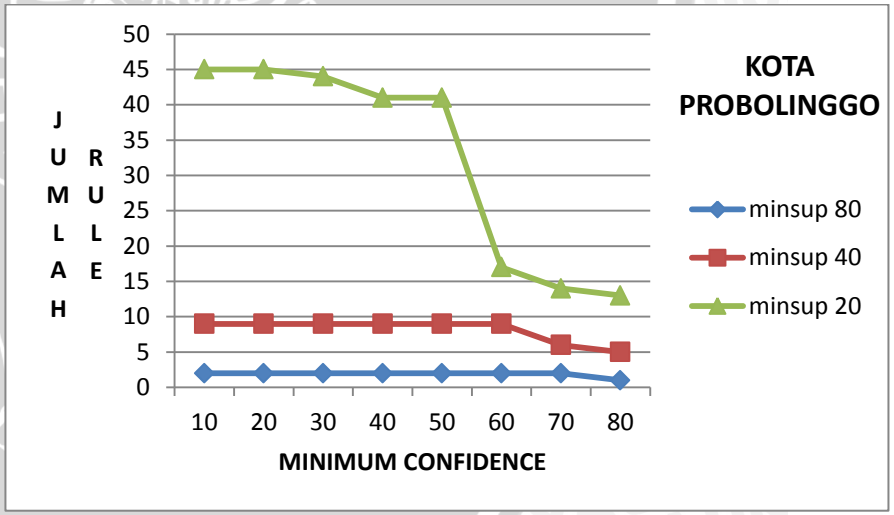

SAMPANG
JUMLAH RULE
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

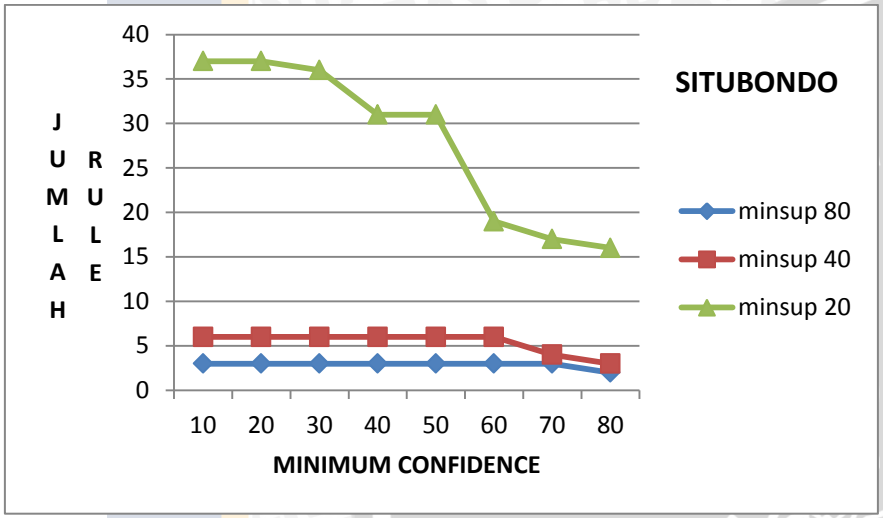

SIDOARJO
JUMLAH RULE
45
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

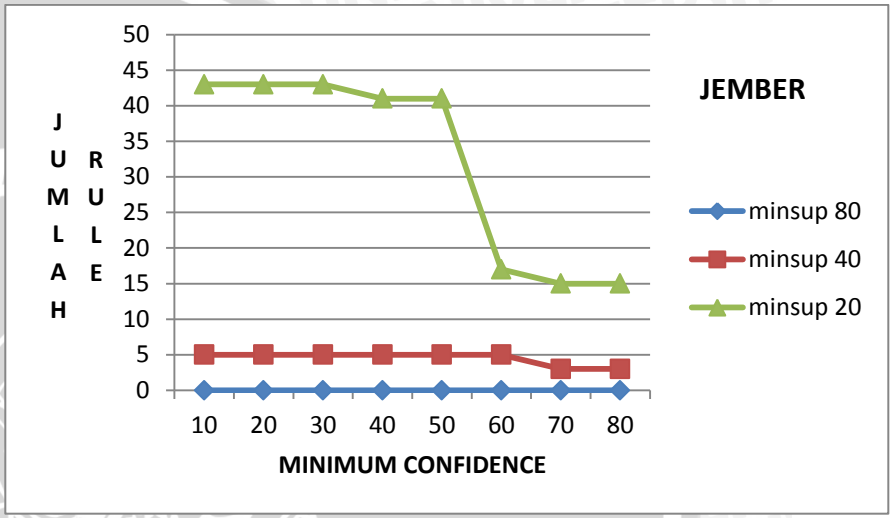

KAB. PASURUAN
JUMLAH RULE
50
45
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

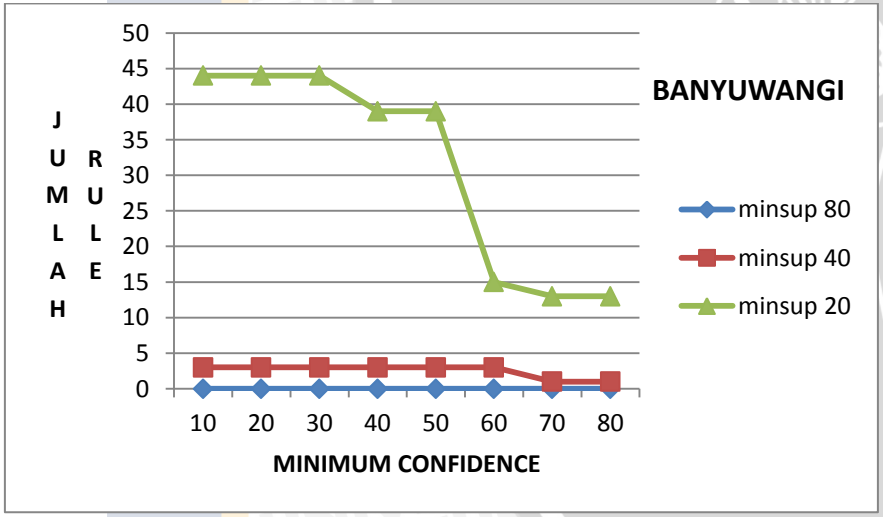

KAB. PROBOLINGGO
JUMLAH RULE
50
45
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUMCONFIDENCE
minsup 80
minsup 40
minsup 20

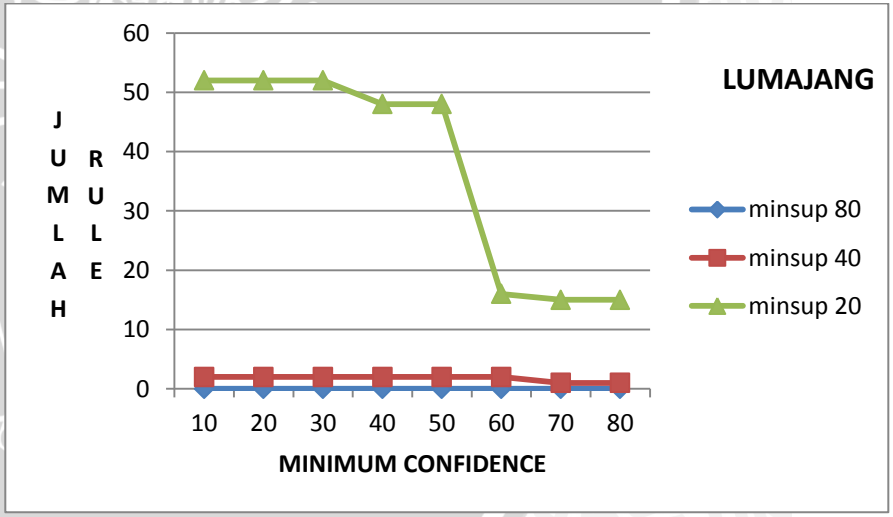

KOTA PASURUAN
JUMLAH RULE
50
45
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

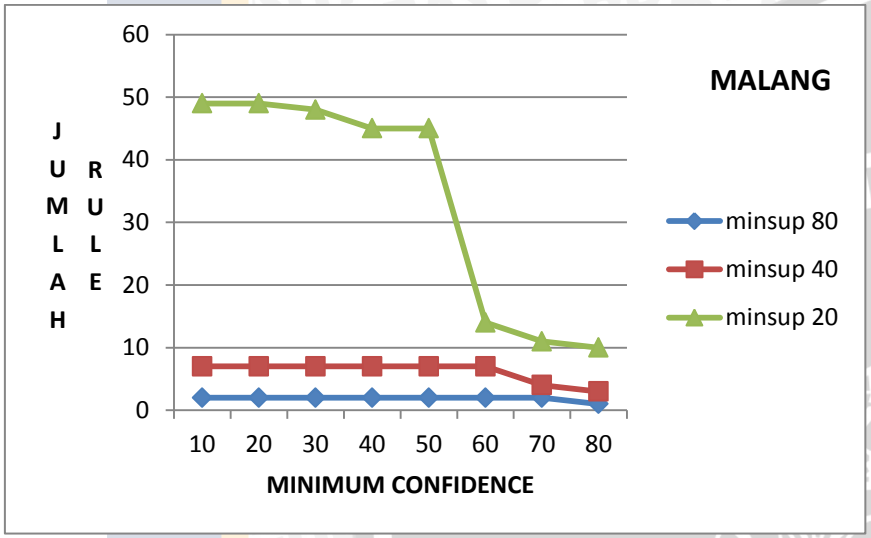

KOTA PROBOLINGGO
JUMLAH RULE
50
45
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

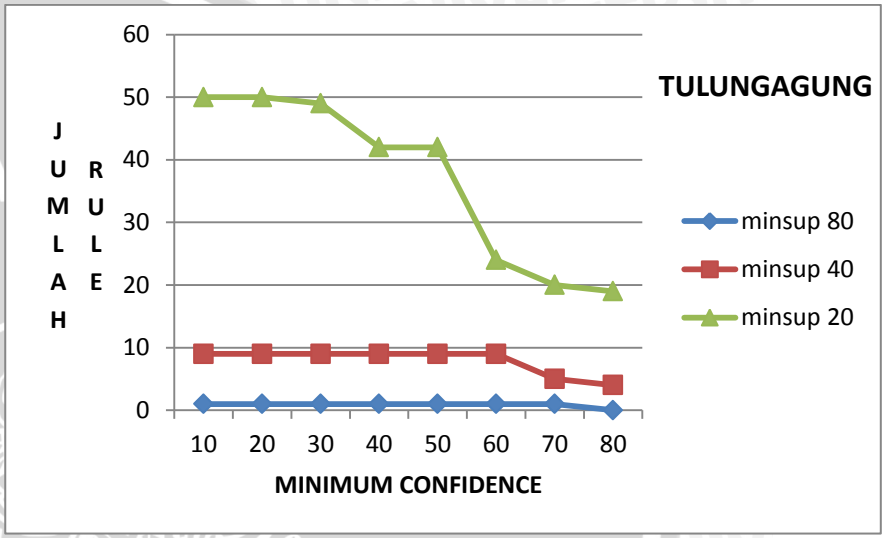

SITUBONDO
JUMLAH RULE
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

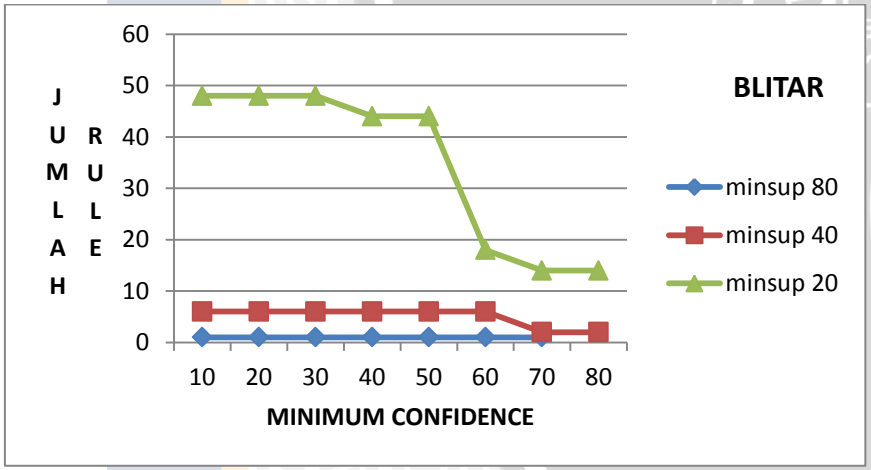

JEMBER
JUMLAH RULE
50
45
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

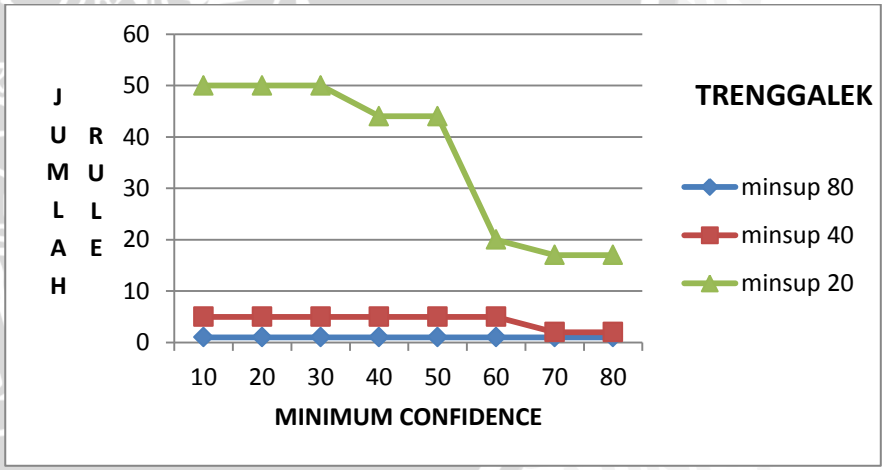

BANYUWANGI
JUMLAH RULE
50
45
40
35
30
25
20
15
10
5
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

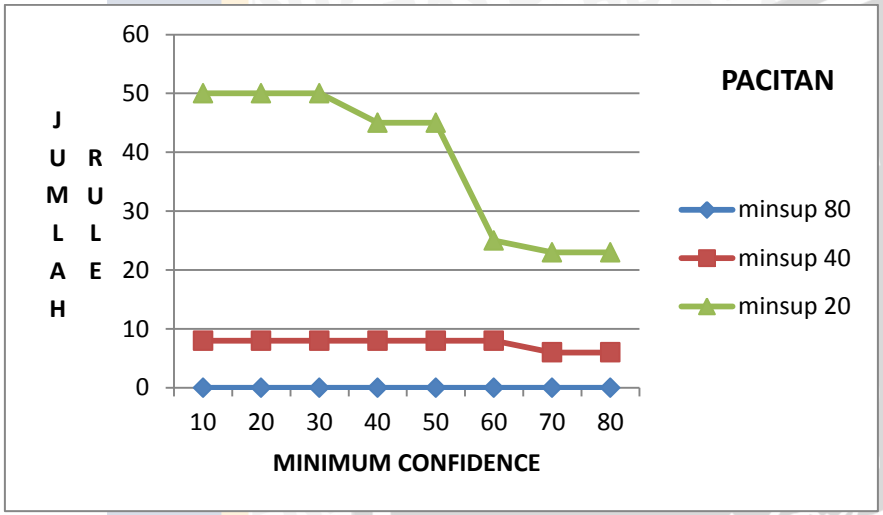

LUMAJANG
JUMLAH RULE
60
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

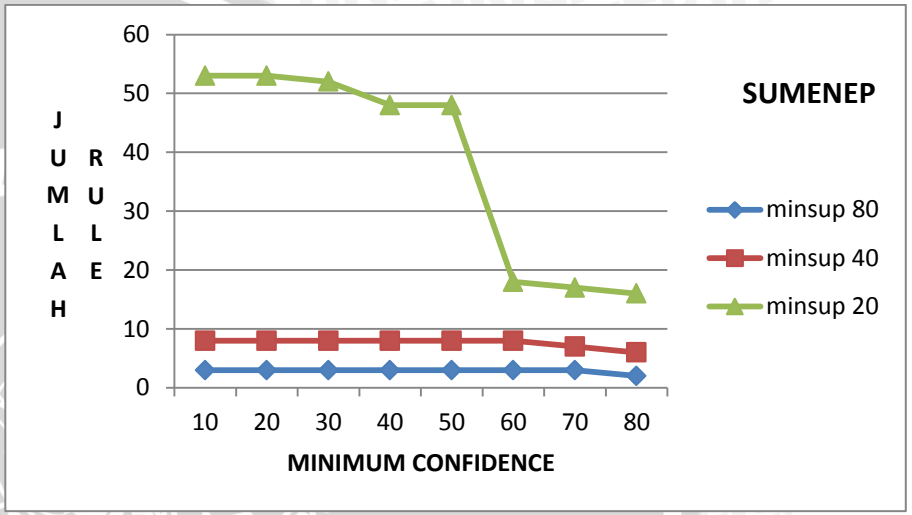

MALANG
JUMLAH RULE
60
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

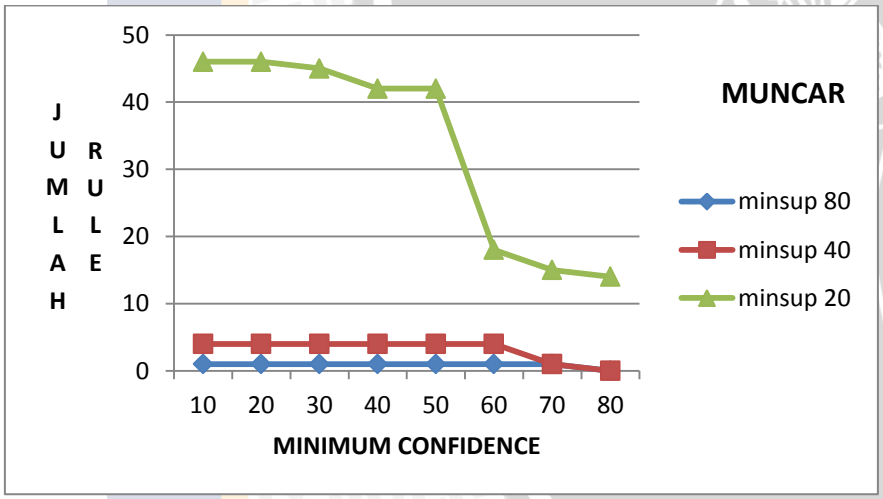

TULUNGAGUNG
JUMLAH RULE
60
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20


BLITAR
JUMLAH RULE
60
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20


TRENGGALEK
JUMLAH RULE
60
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20


PACITAN
JUMLAH RULE
60
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20


SUMENEP
JUMLAH RULE
60
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20


MUNCAR
JUMLAH RULE
50
40
30
20
10
0
10 20 30 40 50 60 70 80
MINIMUM CONFIDENCE
minsup 80
minsup 40
minsup 20

Untuk menggambarkan uji pengaruh nilai *minimum support* dan nilai *minimum confidence* terhadap jumlah *rule* yang dihasilkan pada semua daerah dapat dilhat pada Gambar 5.1. Berdasarkan hasil pengujian bahwa secara umum pengaruh nilai *minimum support* antara 10% hingga 30% dengan nilai *minimum confidence* antara 10% hingga 80% memiliki jumlah *rule* yang sama. Pengaruh nilai *minimum support* antara 40% hingga 60% dengan nilai *minimum confidence* antara 10% hingga 80% juga memiliki jumlah *rule* yang sama pada masing-masing pengujian. Begitu juga dengan uji pengaruh nilai *minimum support* 70% hingga 80% dengan nilai *minimum confidence* antara 10% sampai 80% memiliki jumlah *rule* yang sama. Sehingga ketika digambarkan ke dalam grafik, maka akan terjadi penumpukan posisi garis dan hanya terlihat tiga garis kurva. Dengan mengambil titik batasan nilai *minimum support* 20%, 40% dan 80% yang digunakan dalam menggambarkan grafik bertujuan untuk mengetahui perbandingan uji coba pengaruh nilai *minimum support* dan nilai *minimum confidence* terhadap *rule* yang dihasilkan pada semua daerah. Setelah melakukan pengujian ini dapat diketahui titik batasan nilai *minimum support* dan nilai *minimum confidence* yang dapat menghasilkan *rule* yang bermanfaat adalah pada nilai sebesar 40% sampai 50%.

Pada pengujian kedua yaitu uji coba tingkat kekuatan (*lift ratio*) terhadap *rule* yang dihasilkan, dapat diketahui bahwa setelah dilakukan pengujian pada semua daerah penangkapan mempunyai nilai tingkat kekuatan (*lift ratio*) sebesar lebih dari 1. Pengujian ini menggunakan masukan dengan nilai yang tetap yaitu nilai *minimum support* sebesar 40% dan nilai *minimum confidence* sebesar 50%. Dari hasil uji coba yang dilakukan telah diketahui nilai *lift ratio* terendah adalah 1 dan nilai *lift ratio* tertinggi adalah 2. Sedangkan untuk nilai rata-rata *lift ratio* terendah adalah sebesar 1.24 yang terjadi pada daerah Bangkalan dan Muncar, untuk nilai rata-rata *lift ratio* tertinggi adalah sebesar 1.86 yang terjadi pada daerah Pamekasan.

Berdasarkan nilai *confidence* dan nilai *lift ratio rule* yang dihasilkan dalam pengujian ini, *rule* yang memiliki nilai *confidence* tinggi belum tentu memiliki nilai *lift ratio* yang tinggi pula. Hal ini menjelaskan bahwa meskipun nilai kepercayaan dari *rule* tersebut tinggi namun belum tentu *rule* tersebut bermanfaat

karena nilai *lift ratio* menggambarkan keseimbangan antara frekuensi kemunculan tiap jenis ikan dalam *rule* tersebut secara independen dengan frekuensi kemunculannya secara bersamaan, sedangkan nilai *confidence* menggambarkan tingkat kepercayaan jenis ikan dalam *rule* tersebut muncul secara bersamaan. Nilai *minimum support* yang digunakan juga mempengaruhi untuk mendapatkan *rule* yang benar–benar bermanfaat. Dari hasil pengujian yang ada di sistem yang memenuhi nilai *minimum confidence* = 50% adalah daerah Pamekasan terdapat 5 *rule*, daerah Bangkalan terdapat 4 *rule* dan daerah Muncar terdapat 4 *rule*. *Rule* yang dapat dikatakan bermanfaat dari hasil pengujian ini dapat dilihat pada Tabel 5.4 dan detail informasi *rule* dapat dilihat pada Tabel 5.5.

Tabel 5.4 *Rule* yang Bermanfaat

Rule	**Keterangan**
PAMEKASAN	
10 → 12	Jika ada jenis ikan Kerapu maka ada jenis ikan Kakap
12 → 13	Jika ada jenis ikan Kakap maka ada jenis ikan Kurisi
19 → 33	Jika ada jenis ikan Bawal Hitam maka ada jenis ikan Japuh
32 → 35	Jika ada jenis ikan Teri maka ada jenis ikan Lemuru
41 → 42	Jika ada jenis ikan Layur maka ada jenis ikan Tuna
BANGKALAN	
11 → 8	Jika ada jenis ikan Lencam maka ada jenis ikan Gerot-gerot
24 → 41	Jika ada jenis ikan Kuwe maka ada jenis ikan Layur
41 → 10	Jika ada jenis ikan Layur maka ada jenis ikan Kerapu
8 → 24	Jika ada jenis ikan Gerot-gerot maka ada jenis ikan Kuwe
MUNCAR	
15 → 24	Jika ada jenis ikan Ekor Kuning maka ada jenis ikan Kuwe
24 → 38	Jika ada jenis ikan Kuwe maka ada jenis ikan Kembung
38 → 43	Jika ada jenis ikan Kembung maka ada jenis ikan Cakalang
43 → 2	Jika ada jenis ikan Cakalang maka ada jenis ikan Tongkol

Tabel 5.5 Detail Informasi *Rule* yang Bermanfaat

Rule	***Support* (%)**	***Confidence* (%)**	***Lift Ratio***
PAMEKASAN			
10 → 12	20	100	2.00

12 → 13	20	100	2.00
19 → 33	20	67	1.33
32 → 35	20	100	2.00
41 → 42	20	100	2.00
BANGKALAN			
11 → 8	30	75	1.00
24 → 41	20	67	1.33
41 → 10	20	67	1.33
8 → 24	20	67	1.33
MUNCAR			
15 → 24	30	75	1.00
24 → 38	20	67	1.33
38 → 43	20	67	1.33
43 → 2	20	67	1.33

Masing-masing daerah mempunyai pola ikan dengan kemunculan jenis ikan yang berbeda-beda, sehingga setelah dilakukan pengujian maka hasil *rule* yang bermanfaat dari masing-masing daerah yang didapatkan juga berbeda. Berdasarkan Tabel 5.4 dan Tabel 5.5, hal ini mengartikan bahwa yang dimaksud dengan *rule* yang bermanfaat adalah dari masing-masing pola jenis ikan tersebut merupakan jenis ikan yang paling sering muncul di daerah penangkapan.

BAB VI
PENUTUP

6.1 Kesimpulan

Setelah melakukan penelitian ini maka dapat diperoleh kesimpulan adalah sebagai berikut:

1. Dalam penelitian ini metode *association rule* dengan algoritma *FP Growth* dapat diterapkan dan digunakan untuk mengetahui pola jenis ikan yang muncul pada suatu daerah tertentu. Parameter masukan yang berpengaruh untuk mendapatkan *rule* dengan nilai *confidence* dan nilai *lift ratio* yang tinggi adalah penggunaaan batasan masukan nilai *minimum support* dan nilai *minimum confidence*.
2. *Rule* yang dihasilkan dari penerapan algoritma *FP Growth* dalam permasalahan data hasil tangkapan ikan laut ini memiliki tingkat kekuatan (*lift ratio*) sebesar lebih dari 1 untuk *rule* dengan nilai *minimum confidence* 50% jenis ikan tersebut muncul secara bersamaan. Untuk nilai *lift ratio* tertinggi sebesar 2.00 dan untuk nilai *lift ratio* terendah sebesar 1.00. Nilai rata-rata *lift ratio* tertinggi sebesar 1.86 pada daerah Pamekasan, sedangkan nilai rata-rata terendah sebesar 1.24 pada daerah Bangkalan dan Muncar.

6.2 Saran

Saran yang dapat Penulis berikan setelah mengerjakan penelitian ini adalah sebagai berikut :

1. Untuk pembuatan data penangkapan, harus diperhatikan batasan-batasan ketika akan membuat data tersebut sehingga bisa didapatkan data seperti yang dibutuhkan.
2. Dari hasil penelitian ini dapat dikembangkan lebih luas lagi, dengan memperhatikan kuartal, pengaruh musim, cuaca dan faktor-faktor lainnya yang dapat mempengaruhi hasil tangkapan ikan laut.

DAFTAR PUSTAKA

[AMS-00] Amsyah, Zulkifli. 2000, *Manajemen Sistem Informasi*, PT Gramedia Pustaka Utama, Jakarta.

[DAV-04] Davies, and Paul Beynon. 2004, *Database Systems Third Edition*, Palgrave Macmillan, New York.

[ERW-09] Erwin. 2009, *Analisis Market Basket dengan Algoritma Apriori dan FP-Growth*, Jurnal Generic Volume 4 No. 2 Fakultas Ilmu Komputer Universitas Sriwijaya, Sumatera Selatan.

[FEB-09] Febriyana, Fatma Rika. 2009. *Perbandingan Kecepatan dalam Pencarian Frequent Itemset antara Algoritma FP – Growth dan Cut Both Ways*. Universitas Brawijaya. Malang

[HAN-01] Han, Jiawei dan Micheline Kamber. 2001, *Data Mining : Concepts and Techniques*, Morgan Kaufmann, California.

[HAN-04] Handojo, Andreas, Gregorius Satia Budhi, Hendra Rusly. 2004, *Aplikasi Data Mining untuk Meneliti Asosiasi Pembelian Item Barang di Supermarket dengan Metode Market Basket Anaysis*, Seminar Nasional Teknologi Informasi Universitas Kristen Petra, Surabaya.

[HAN-06] Han, Jiawei dan Micheline Kamber. 2006, *Data Mining : Concepts and Techniques 2nd Edition*, Morgan Kaufmann, California.

[HUD-10] Huda, Nuqson Masykur. 2010. *Aplikasi Data Mining untuk Menampilkan Informasi Tingkat Kelulusan Mahasiswa*. Semarang : Universitas Diponegoro.

[KAN-03] Kantardzic, Mehmed. 2003, *Data Mining : Concepts, Models, Methods, and Algorithms*, John Willey & Sons, Inc., New Jersey.

[KUS-07] Kusnawi. 2007, *Pengantar Solusi Data Mining*, Seminar Nasional Teknologi (SNT) STMIK AMIKOM, Yogyakarta.

[KUS-07] Kusrini, M.Kom. 2007, *Konsep dan Aplikasi Sistem Pendukung Keputusan*, CV Andi Offset, Yogyakarta.

[PRA-03] Pramudiono, Iko. 2003, *Pengantar Data Mining : Menambang Permata Pengetahuan di Gunung Data*, http://www.ilmukomputer.com.

[PRA-06] Prasetyo, Bowo. 2006, *Analisis Perilaku Pengunjung Menggunakan Data Mining*, http://www.beritaiptek.com/zberita-beritaiptek-2006-01-05-Analisis-Perilaku-Pengunjung-Menggunakan-Data-Mining.shtml.

[ROC-10] Rochmah, Affriantari. 2010, *Perancangan Fitur Rekomendasi Film Website Solo Movie dengan Menggabungkan Metode Algoritma Apriori*, Universitas Sebelas Maret, Surakarta.

[ROM-05] Romney, Marshall B. 2005, *Sistem Informasi Akuntansi*, Salemba Empat, Jakarta.

[RUL-08] Ruldeviyani, Yova dan Muhammad Fahrian. 2008, *Implementasi Algoritma-Algoritma Association Rules Sebagai Bagian dari Pengembangan Data Mining Algorithms Collection*, Konferensi Nasional Sistem dan Informatika, Bali.

[SAM-08] Samuel, David. 2008, *Penerapan Struktur FP-Tree dan Algoritma FP-Growth dalam Optimasi Penentuan Frequent Itemset*, Institut Teknologi Bandung, Bandung.

[SAN-07] Santosa, Budi. 2007, *Data Mining Teknik Pemanfaatan Data untuk Keperluan Bisnis*, Graha Ilmu, Yogyakarta.

[SEL-08] Selamet, Rachmat. 2008, *Association Rule*, Sekolah Tinggi Manajemen Informatika dan Komputer LIKMI, Bandung.

[SUC-03] Sucahyo, Yudho Giri. 2003, *Data Mining Menggali Informasi yang Terpendam*, http://www.ilmukomputer.com.

[VER-08] Verhein, Florian. 2008. *Frequent Pattern Growth (FP – Growth) Algorithm*. School of Information Technologies The University of Sydney. Australia.

[WIT-05] Witten, I. H dan Frank, E. 2005, *Data Mining : Practical Machine Learning Tools and Techniques Second Edition*, Morgan Kauffman, San Fransisco.

[YIA-06] Yiapanis, Paraskevas. 2006, *Parallel Mining of Minimal Sample Unique Itemset*, School of Computer Science University of Manchester, Manchester.

[YUL-02] Yulita, Marsela dan Veronica S. Moertini. 2002, *Analisis Keranjang Pasar dengan Algoritma Hash-Based pada Transaksi Penjualan di Apotek*. Integral Majalah Ilmiah Matematika dan Ilmu Pengetahuan Alam, Volume 9 No. 3, Jurusan Ilmu Komputer Universitas Katolik Parahyangan, Bandung.

LAMPIRAN

Tabel Data Daerah

Id Daerah	Nama Daerah
1	KABUPATEN TUBAN
2	KABUPATEN LAMONGAN
3	KABUPATEN GRESIK
4	KOTA SURABAYA
5	KABUPATEN BANGKALAN
6	KABUPATEN SAMPANG
7	KABUPATEN PAMEKASAN
8	KABUPATEN SIDOARJO
9	KABUPATEN PASURUAN
10	KOTA PASURUAN
11	KABUPATEN PROBOLINGGO
12	KOTA PROBOLINGGO
13	KABUPATEN SITUBONDO
14	KABUPATEN BANYUWANGI
15	KABUPATEN JEMBER
16	KABUPATEN LUMAJANG
17	KABUPATEN MALANG
18	KABUPATEN BLITAR
19	KABUPATEN TULUNGAGUNG
20	KABUPATEN TRENGGALEK
21	KABUPATEN PACITAN
22	DAREK MUNCAR
23	KABUPATEN SUMENEP

Tabel Data Ikan

Id Ikan	Nama Ikan
1	SEBELAH
2	LIDAH
3	NOMEI
4	PEPEREK
5	MANYUNG
6	BELOSO
7	BIJI NANGKA
8	GEROT-GEROT
9	MERAH
10	KERAPU
11	LENCAM
12	KAKAP
13	KURISI
14	SWANGGI
15	EKOR KUNING
16	GULAMAH
17	CUCUT
18	PARI
19	BAWAL HITAM
20	BAWAL PUTIH
21	ALU-ALU
22	LAYANG
23	SELAR
24	KUWE
25	TETENGKEK
26	DAUN BAMBU
27	SUNGLIR
28	IKAN TERBANG
29	BELANAK
30	KURO
31	JULUNG-JULUNG
32	TERI
33	JAPUH
34	TEMBANG
35	LEMURU
36	GOLOK-GOLOK
37	TERUBUK
38	KEMBUNG
39	TENGGIRI PAPAN
40	TENGGIRI
41	LAYUR
42	TUNA
43	CAKALANG
44	TONGKOL

Tabel Uji Coba Pengaruh Nilai *Minimum Support* dan Nilai *Minimum Confidence* terhadap Jumlah *Rule* yang Dihasilkan

Daerah	Nilai *Minimum Support* (%)	Nilai *Minimum Confidence* (%)	Jumlah *Rule* yang Dihasilkan
Lamongan	10	10	41
	10	20	41
	10	30	41
	10	40	41
	10	50	38
	10	60	16
	10	70	16
	10	80	16
	20	10	41
	20	20	41
	20	30	41
	20	40	41
	20	50	38
	20	60	16
	20	70	16
	20	80	16
	30	10	41
	30	20	41
	30	30	41
	30	40	41
	30	50	38
	30	60	16
	30	70	16
	30	80	16
	40	10	3
	40	20	3
	40	30	3
	40	40	3
	40	50	3
	40	60	3
	40	70	3
	40	80	3
	50	10	3
	50	20	3
	50	30	3
	50	40	3
	50	50	3
	50	60	3

	50	70	3
	50	80	3
	60	10	3
	60	20	3
	60	30	3
	60	40	3
	60	50	3
	60	60	3
	60	70	3
	60	80	3
	70	10	1
	70	20	1
	70	30	1
	70	40	1
	70	50	1
	70	60	1
	70	70	1
	70	80	1
	80	10	1
	80	20	1
	80	30	1
	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	1

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Gresik	10	10	47
	10	20	47
	10	30	47
	10	40	39
	10	50	39
	10	60	23
	10	70	19
	10	80	18
	20	10	47
	20	20	47
	20	30	47
	20	40	39
	20	50	39

	20	60	23
	20	70	19
	20	80	18
	30	10	47
	30	20	47
	30	30	47
	30	40	39
	30	50	39
	30	60	23
	30	70	19
	30	80	18
	40	10	8
	40	20	8
	40	30	8
	40	40	8
	40	50	8
	40	60	8
	40	70	4
	40	80	3
	50	10	8
	50	20	8
	50	30	8
	50	40	8
	50	50	8
	50	60	8
	50	70	4
	50	80	3
	60	10	8
	60	20	8
	60	30	8
	60	40	8
	60	50	8
	60	60	8
	60	70	4
	60	80	3
	70	10	2
	70	20	2
	70	30	2
	70	40	2
	70	50	2
	70	60	2
	70	70	2
	70	80	1

	80	10	2
	80	20	2
	80	30	2
	80	40	2
	80	50	2
	80	60	2
	80	70	2
	80	80	1

Daerah	Nilai *Minimum Support* (%)	Nilai *Minimum Confidence* (%)	Jumlah *Rule* yang Dihasilkan
Surabaya	10	10	43
	10	20	43
	10	30	43
	10	40	41
	10	50	41
	10	60	21
	10	70	19
	10	80	19
	20	10	43
	20	20	43
	20	30	43
	20	40	41
	20	50	41
	20	60	21
	20	70	19
	20	80	19
	30	10	43
	30	20	43
	30	30	43
	30	40	41
	30	50	41
	30	60	21
	30	70	19
	30	80	19
	40	10	5
	40	20	5
	40	30	5
	40	40	5
	40	50	5
	40	60	5
	40	70	3

	40	80	3
	50	10	5
	50	20	5
	50	30	5
	50	40	5
	50	50	5
	50	60	5
	50	70	3
	50	80	3
	60	10	5
	60	20	5
	60	30	5
	60	40	5
	60	50	5
	60	60	5
	60	70	3
	60	80	3
	70	10	0
	70	20	0
	70	30	0
	70	40	0
	70	50	0
	70	60	0
	70	70	0
	70	80	0
	80	10	0
	80	20	0
	80	30	0
	80	40	0
	80	50	0
	80	60	0
	80	70	0
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Bangkalan	10	10	47
	10	20	47
	10	30	47
	10	40	46
	10	50	43
	10	60	19

	10	70	16
	10	80	15
	20	10	47
	20	20	47
	20	30	47
	20	40	46
	20	50	43
	20	60	19
	20	70	16
	20	80	15
	30	10	47
	30	20	47
	30	30	47
	30	40	46
	30	50	43
	30	60	19
	30	70	16
	30	80	15
	40	10	4
	40	20	4
	40	30	4
	40	40	4
	40	50	4
	40	60	4
	40	70	1
	40	80	0
	50	10	4
	50	20	4
	50	30	4
	50	40	4
	50	50	4
	50	60	4
	50	70	1
	50	80	0
	60	10	4
	60	20	4
	60	30	4
	60	40	4
	60	50	4
	60	60	4
	60	70	1
	60	80	0
	70	10	1

	70	20	1
	70	30	1
	70	40	1
	70	50	1
	70	60	1
	70	70	1
	70	80	0
	80	10	1
	80	20	1
	80	30	1
	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Sampang	10	10	38
	10	20	38
	10	30	38
	10	40	34
	10	50	34
	10	60	20
	10	70	16
	10	80	16
	20	10	38
	20	20	38
	20	30	38
	20	40	34
	20	50	34
	20	60	20
	20	70	16
	20	80	16
	30	10	38
	30	20	38
	30	30	38
	30	40	34
	30	50	34
	30	60	20
	30	70	16
	30	80	16

	40	10	10
	40	20	10
	40	30	10
	40	40	10
	40	50	10
	40	60	10
	40	70	6
	40	80	6
	50	10	10
	50	20	10
	50	30	10
	50	40	10
	50	50	10
	50	60	10
	50	70	6
	50	80	6
	60	10	10
	60	20	10
	60	30	10
	60	40	10
	60	50	10
	60	60	10
	60	70	6
	60	80	6
	70	10	1
	70	20	1
	70	30	1
	70	40	1
	70	50	1
	70	60	1
	70	70	1
	70	80	1
	80	10	1
	80	20	1
	80	30	1
	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	1

Daerah	Nilai *Minimum Support*	Nilai *Minimum*	Jumlah *Rule*

	(%)	*Confidence* (%)	yang Dihasilkan
Pamekasan	10	10	48
	10	20	48
	10	30	48
	10	40	44
	10	50	44
	10	60	22
	10	70	21
	10	80	21
	20	10	48
	20	20	48
	20	30	48
	20	40	44
	20	50	44
	20	60	22
	20	70	21
	20	80	21
	30	10	48
	30	20	48
	30	30	48
	30	40	44
	30	50	44
	30	60	22
	30	70	21
	30	80	21
	40	10	5
	40	20	5
	40	30	5
	40	40	5
	40	50	5
	40	60	5
	40	70	4
	40	80	4
	50	10	5
	50	20	5
	50	30	5
	50	40	5
	50	50	5
	50	60	5
	50	70	4
	50	80	4
	60	10	5
	60	20	5

	60	30	5
	60	40	5
	60	50	5
	60	60	5
	60	70	4
	60	80	4
	70	10	0
	70	20	0
	70	30	0
	70	40	0
	70	50	0
	70	60	0
	70	70	0
	70	80	0
	80	10	0
	80	20	0
	80	30	0
	80	40	0
	80	50	0
	80	60	0
	80	70	0
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Sidoarjo	10	10	41
	10	20	41
	10	30	40
	10	40	35
	10	50	35
	10	60	19
	10	70	17
	10	80	16
	20	10	41
	20	20	41
	20	30	40
	20	40	35
	20	50	35
	20	60	19
	20	70	17
	20	80	16
	30	10	41

	30	20	41
	30	30	40
	30	40	35
	30	50	35
	30	60	19
	30	70	17
	30	80	16
	40	10	5
	40	20	5
	40	30	5
	40	40	5
	40	50	5
	40	60	5
	40	70	3
	40	80	2
	50	10	5
	50	20	5
	50	30	5
	50	40	5
	50	50	5
	50	60	5
	50	70	3
	50	80	2
	60	10	5
	60	20	5
	60	30	5
	60	40	5
	60	50	5
	60	60	5
	60	70	3
	60	80	2
	70	10	2
	70	20	2
	70	30	2
	70	40	2
	70	50	2
	70	60	2
	70	70	2
	70	80	1
	80	10	2
	80	20	2
	80	30	2
	80	40	2

	80	50	2
	80	60	2
	80	70	2
	80	80	1

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Kab. Pasuruan	10	10	45
	10	20	45
	10	30	44
	10	40	39
	10	50	39
	10	60	19
	10	70	17
	10	80	16
	20	10	45
	20	20	45
	20	30	44
	20	40	39
	20	50	39
	20	60	19
	20	70	17
	20	80	16
	30	10	45
	30	20	45
	30	30	44
	30	40	39
	30	50	39
	30	60	19
	30	70	17
	30	80	16
	40	10	6
	40	20	6
	40	30	6
	40	40	6
	40	50	6
	40	60	6
	40	70	4
	40	80	3
	50	10	6
	50	20	6
	50	30	6

	50	40	6
	50	50	6
	50	60	6
	50	70	4
	50	80	3
	60	10	6
	60	20	6
	60	30	6
	60	40	6
	60	50	6
	60	60	6
	60	70	4
	60	80	3
	70	10	1
	70	20	1
	70	30	1
	70	40	1
	70	50	1
	70	60	1
	70	70	1
	70	80	0
	80	10	1
	80	20	1
	80	30	1
	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Kota Pasuruan	10	10	45
	10	20	45
	10	30	45
	10	40	42
	10	50	42
	10	60	20
	10	70	20
	10	80	20
	20	10	45
	20	20	45

	20	30	45
	20	40	42
	20	50	42
	20	60	20
	20	70	20
	20	80	20
	30	10	45
	30	20	45
	30	30	45
	30	40	42
	30	50	42
	30	60	20
	30	70	20
	30	80	20
	40	10	3
	40	20	3
	40	30	3
	40	40	3
	40	50	3
	40	60	3
	40	70	3
	40	80	3
	50	10	3
	50	20	3
	50	30	3
	50	40	3
	50	50	3
	50	60	3
	50	70	3
	50	80	3
	60	10	3
	60	20	3
	60	30	3
	60	40	3
	60	50	3
	60	60	3
	60	70	3
	60	80	3
	70	10	1
	70	20	1
	70	30	1
	70	40	1
	70	50	1

	70	60	1
	70	70	1
	70	80	1
	80	10	1
	80	20	1
	80	30	1
	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	1

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Kab. Probolinggo	10	10	47
	10	20	47
	10	30	47
	10	40	42
	10	50	42
	10	60	20
	10	70	18
	10	80	18
	20	10	47
	20	20	47
	20	30	47
	20	40	42
	20	50	42
	20	60	20
	20	70	18
	20	80	18
	30	10	47
	30	20	47
	30	30	47
	30	40	42
	30	50	42
	30	60	20
	30	70	18
	30	80	18
	40	10	3
	40	20	3
	40	30	3
	40	40	3

	40	50	3
	40	60	3
	40	70	1
	40	80	1
	50	10	3
	50	20	3
	50	30	3
	50	40	3
	50	50	3
	50	60	3
	50	70	1
	50	80	1
	60	10	3
	60	20	3
	60	30	3
	60	40	3
	60	50	3
	60	60	3
	60	70	1
	60	80	1
	70	10	0
	70	20	0
	70	30	0
	70	40	0
	70	50	0
	70	60	0
	70	70	0
	70	80	0
	80	10	0
	80	20	0
	80	30	0
	80	40	0
	80	50	0
	80	60	0
	80	70	0
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Kota Probolinggo	10	10	45
	10	20	45
	10	30	44

	10	40	41
	10	50	41
	10	60	17
	10	70	14
	10	80	13
	20	10	45
	20	20	45
	20	30	44
	20	40	41
	20	50	41
	20	60	17
	20	70	14
	20	80	13
	30	10	45
	30	20	45
	30	30	44
	30	40	41
	30	50	41
	30	60	17
	30	70	14
	30	80	13
	40	10	9
	40	20	9
	40	30	9
	40	40	9
	40	50	9
	40	60	9
	40	70	6
	40	80	5
	50	10	9
	50	20	9
	50	30	9
	50	40	9
	50	50	9
	50	60	9
	50	70	6
	50	80	5
	60	10	9
	60	20	9
	60	30	9
	60	40	9
	60	50	9
	60	60	9

	60	70	6
	60	80	5
	70	10	2
	70	20	2
	70	30	2
	70	40	2
	70	50	2
	70	60	2
	70	70	2
	70	80	1
	80	10	2
	80	20	2
	80	30	2
	80	40	2
	80	50	2
	80	60	2
	80	70	2
	80	80	1

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Situbondo	10	10	37
	10	20	37
	10	30	36
	10	40	31
	10	50	31
	10	60	19
	10	70	17
	10	80	16
	20	10	37
	20	20	37
	20	30	36
	20	40	31
	20	50	31
	20	60	19
	20	70	17
	20	80	16
	30	10	37
	30	20	37
	30	30	36
	30	40	31
	30	50	31

	30	60	19
	30	70	17
	30	80	16
	40	10	6
	40	20	6
	40	30	6
	40	40	6
	40	50	6
	40	60	6
	40	70	4
	40	80	3
	50	10	6
	50	20	6
	50	30	6
	50	40	6
	50	50	6
	50	60	6
	50	70	4
	50	80	3
	60	10	6
	60	20	6
	60	30	6
	60	40	6
	60	50	6
	60	60	6
	60	70	4
	60	80	3
	70	10	3
	70	20	3
	70	30	3
	70	40	3
	70	50	3
	70	60	3
	70	70	3
	70	80	2
	80	10	3
	80	20	3
	80	30	3
	80	40	3
	80	50	3
	80	60	3
	80	70	3
	80	80	2

Daerah	Nilai *Minimum Support* (%)	Nilai *Minimum Confidence* (%)	Jumlah *Rule* yang Dihasilkan
Banyuwangi	10	10	44
	10	20	44
	10	30	44
	10	40	39
	10	50	39
	10	60	15
	10	70	13
	10	80	13
	20	10	44
	20	20	44
	20	30	44
	20	40	39
	20	50	39
	20	60	15
	20	70	13
	20	80	13
	30	10	44
	30	20	44
	30	30	44
	30	40	39
	30	50	39
	30	60	15
	30	70	13
	30	80	13
	40	10	3
	40	20	3
	40	30	3
	40	40	3
	40	50	3
	40	60	3
	40	70	1
	40	80	1
	50	10	3
	50	20	3
	50	30	3
	50	40	3
	50	50	3
	50	60	3
	50	70	1

	50	80	1
	60	10	3
	60	20	3
	60	30	3
	60	40	3
	60	50	3
	60	60	3
	60	70	1
	60	80	1
	70	10	0
	70	20	0
	70	30	0
	70	40	0
	70	50	0
	70	60	0
	70	70	0
	70	80	0
	80	10	0
	80	20	0
	80	30	0
	80	40	0
	80	50	0
	80	60	0
	80	70	0
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Jember	10	10	43
	10	20	43
	10	30	43
	10	40	41
	10	50	41
	10	60	17
	10	70	15
	10	80	15
	20	10	43
	20	20	43
	20	30	43
	20	40	41
	20	50	41
	20	60	17

	20	70	15
	20	80	15
	30	10	43
	30	20	43
	30	30	43
	30	40	41
	30	50	41
	30	60	17
	30	70	15
	30	80	15
	40	10	5
	40	20	5
	40	30	5
	40	40	5
	40	50	5
	40	60	5
	40	70	3
	40	80	3
	50	10	5
	50	20	5
	50	30	5
	50	40	5
	50	50	5
	50	60	5
	50	70	3
	50	80	3
	60	10	5
	60	20	5
	60	30	5
	60	40	5
	60	50	5
	60	60	5
	60	70	3
	60	80	3
	70	10	0
	70	20	0
	70	30	0
	70	40	0
	70	50	0
	70	60	0
	70	70	0
	70	80	0
	80	10	0

	80	20	0
	80	30	0
	80	40	0
	80	50	0
	80	60	0
	80	70	0
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Lumajang	10	10	52
	10	20	52
	10	30	52
	10	40	48
	10	50	48
	10	60	16
	10	70	15
	10	80	15
	20	10	52
	20	20	52
	20	30	52
	20	40	48
	20	50	48
	20	60	16
	20	70	15
	20	80	15
	30	10	52
	30	20	52
	30	30	52
	30	40	48
	30	50	48
	30	60	16
	30	70	15
	30	80	15
	40	10	2
	40	20	2
	40	30	2
	40	40	2
	40	50	2
	40	60	2
	40	70	1
	40	80	1

	50	10	2
	50	20	2
	50	30	2
	50	40	2
	50	50	2
	50	60	2
	50	70	1
	50	80	1
	60	10	2
	60	20	2
	60	30	2
	60	40	2
	60	50	2
	60	60	2
	60	70	1
	60	80	1
	70	10	0
	70	20	0
	70	30	0
	70	40	0
	70	50	0
	70	60	0
	70	70	0
	70	80	0
	80	10	0
	80	20	0
	80	30	0
	80	40	0
	80	50	0
	80	60	0
	80	70	0
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Malang	10	10	49
	10	20	49
	10	30	48
	10	40	45
	10	50	45
	10	60	14
	10	70	11

	10	80	10
	20	10	49
	20	20	49
	20	30	48
	20	40	45
	20	50	45
	20	60	14
	20	70	11
	20	80	10
	30	10	49
	30	20	49
	30	30	48
	30	40	45
	30	50	45
	30	60	14
	30	70	11
	30	80	10
	40	10	7
	40	20	7
	40	30	7
	40	40	7
	40	50	7
	40	60	7
	40	70	4
	40	80	3
	50	10	7
	50	20	7
	50	30	7
	50	40	7
	50	50	7
	50	60	7
	50	70	4
	50	80	3
	60	10	7
	60	20	7
	60	30	7
	60	40	7
	60	50	7
	60	60	7
	60	70	4
	60	80	3
	70	10	2
	70	20	2

	70	30	2
	70	40	2
	70	50	2
	70	60	2
	70	70	2
	70	80	1
	80	10	2
	80	20	2
	80	30	2
	80	40	2
	80	50	2
	80	60	2
	80	70	2
	80	80	1

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Blitar	10	10	48
	10	20	48
	10	30	48
	10	40	44
	10	50	44
	10	60	18
	10	70	14
	10	80	14
	20	10	48
	20	20	48
	20	30	48
	20	40	44
	20	50	44
	20	60	18
	20	70	14
	20	80	14
	30	10	48
	30	20	48
	30	30	48
	30	40	44
	30	50	44
	30	60	18
	30	70	14

	30	80	14
	40	10	6
	40	20	6
	40	30	6
	40	40	6
	40	50	6
	40	60	6
	40	70	2
	40	80	2
	50	10	6
	50	20	6
	50	30	6
	50	40	6
	50	50	6
	50	60	6
	50	70	2
	50	80	2
	60	10	6
	60	20	6
	60	30	6
	60	40	6
	60	50	6
	60	60	6
	60	70	2
	60	80	2
	70	10	1
	70	20	1
	70	30	1
	70	40	1
	70	50	1
	70	60	1
	70	70	1
	70	80	1
	80	10	1
	80	20	1
	80	30	1
	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	1

Daerah	Nilai *Minimum Support* (%)	Nilai *Minimum Confidence* (%)	Jumlah *Rule* yang Dihasilkan
Tulungagung	10	10	50
	10	20	50
	10	30	49
	10	40	42
	10	50	42
	10	60	24
	10	70	20
	10	80	19
	20	10	50
	20	20	50
	20	30	49
	20	40	42
	20	50	42
	20	60	24
	20	70	20
	20	80	19
	30	10	50
	30	20	50
	30	30	49
	30	40	42
	30	50	42
	30	60	24
	30	70	20
	30	80	19
	40	10	9
	40	20	9
	40	30	9
	40	40	9
	40	50	9
	40	60	9
	40	70	5
	40	80	4
	50	10	9
	50	20	9
	50	30	9
	50	40	9
	50	50	9
	50	60	9
	50	70	5
	50	80	4
	60	10	9

	60	20	9
	60	30	9
	60	40	9
	60	50	9
	60	60	9
	60	70	5
	60	80	4
	70	10	1
	70	20	1
	70	30	1
	70	40	1
	70	50	1
	70	60	1
	70	70	1
	70	80	0
	80	10	1
	80	20	1
	80	30	1
	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Trenggalek	10	10	50
	10	20	50
	10	30	50
	10	40	44
	10	50	44
	10	60	20
	10	70	17
	10	80	17
	20	10	50
	20	20	50
	20	30	50
	20	40	44
	20	50	44
	20	60	20
	20	70	17
	20	80	17

	30	10	50
	30	20	50
	30	30	50
	30	40	44
	30	50	44
	30	60	20
	30	70	17
	30	80	17
	40	10	5
	40	20	5
	40	30	5
	40	40	5
	40	50	5
	40	60	5
	40	70	2
	40	80	2
	50	10	5
	50	20	5
	50	30	5
	50	40	5
	50	50	5
	50	60	5
	50	70	2
	50	80	2
	60	10	5
	60	20	5
	60	30	5
	60	40	5
	60	50	5
	60	60	5
	60	70	2
	60	80	2
	70	10	1
	70	20	1
	70	30	1
	70	40	1
	70	50	1
	70	60	1
	70	70	1
	70	80	1
	80	10	1
	80	20	1
	80	30	1

	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	1

Daerah	Nilai *Minimum Support* (%)	Nilai *Minimum Confidence* (%)	Jumlah *Rule* yang Dihasilkan
Pacitan	10	10	50
	10	20	50
	10	30	50
	10	40	45
	10	50	45
	10	60	25
	10	70	23
	10	80	23
	20	10	50
	20	20	50
	20	30	50
	20	40	45
	20	50	45
	20	60	25
	20	70	23
	20	80	23
	30	10	50
	30	20	50
	30	30	50
	30	40	45
	30	50	45
	30	60	25
	30	70	23
	30	80	23
	40	10	8
	40	20	8
	40	30	8
	40	40	8
	40	50	8
	40	60	8
	40	70	6
	40	80	6
	50	10	8
	50	20	8

	50	30	8
	50	40	8
	50	50	8
	50	60	8
	50	70	6
	50	80	6
	60	10	8
	60	20	8
	60	30	8
	60	40	8
	60	50	8
	60	60	8
	60	70	6
	60	80	6
	70	10	0
	70	20	0
	70	30	0
	70	40	0
	70	50	0
	70	60	0
	70	70	0
	70	80	0
	80	10	0
	80	20	0
	80	30	0
	80	40	0
	80	50	0
	80	60	0
	80	70	0
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Muncar	10	10	46
	10	20	46
	10	30	45
	10	40	42
	10	50	42
	10	60	18
	10	70	15
	10	80	14
	20	10	46

	20	20	46
	20	30	45
	20	40	42
	20	50	42
	20	60	18
	20	70	15
	20	80	14
	30	10	46
	30	20	46
	30	30	45
	30	40	42
	30	50	42
	30	60	18
	30	70	15
	30	80	14
	40	10	4
	40	20	4
	40	30	4
	40	40	4
	40	50	4
	40	60	4
	40	70	1
	40	80	0
	50	10	4
	50	20	4
	50	30	4
	50	40	4
	50	50	4
	50	60	4
	50	70	1
	50	80	0
	60	10	4
	60	20	4
	60	30	4
	60	40	4
	60	50	4
	60	60	4
	60	70	1
	60	80	0
	70	10	1
	70	20	1
	70	30	1
	70	40	1

	70	50	1
	70	60	1
	70	70	1
	70	80	0
	80	10	1
	80	20	1
	80	30	1
	80	40	1
	80	50	1
	80	60	1
	80	70	1
	80	80	0

Daerah	**Nilai *Minimum Support* (%)**	**Nilai *Minimum Confidence* (%)**	**Jumlah *Rule* yang Dihasilkan**
Sumenep	10	10	53
	10	20	53
	10	30	52
	10	40	48
	10	50	48
	10	60	18
	10	70	17
	10	80	16
	20	10	53
	20	20	53
	20	30	52
	20	40	48
	20	50	48
	20	60	18
	20	70	17
	20	80	16
	30	10	53
	30	20	53
	30	30	52
	30	40	48
	30	50	48
	30	60	18
	30	70	17
	30	80	16
	40	10	8
	40	20	8
	40	30	8

	40	40	8
	40	50	8
	40	60	8
	40	70	7
	40	80	6
	50	10	8
	50	20	8
	50	30	8
	50	40	8
	50	50	8
	50	60	8
	50	70	7
	50	80	6
	60	10	8
	60	20	8
	60	30	8
	60	40	8
	60	50	8
	60	60	8
	60	70	7
	60	80	6
	70	10	3
	70	20	3
	70	30	3
	70	40	3
	70	50	3
	70	60	3
	70	70	3
	70	80	2
	80	10	3
	80	20	3
	80	30	3
	80	40	3
	80	50	3
	80	60	3
	80	70	3
	80	80	2

Tabel Uji Coba Tingkat Kekuatan (*Lift Ratio*) terhadap *Rule* yang Dihasilkan

Daerah	*Rule* yang Dihasilkan	Nilai *Confidence*	Nilai *Lift Ratio*
Tuban	11→15	0.67	1.33
	14 →11	0.75	1.00
	15 → 17	1.00	1.33
	17→ 44	1.00	1.33
	23 → 26	1.00	2.00
	37 → 38	1.00	2.00
	38→ 41	1.00	2.00
	44 → 5	0.67	1.33
	5 → 6	1.00	2.00
	Rata-rata Nilai Lift Ratio		**1.59**
Lamongan	12 → 26	1.00	2.00
	13 → 16	0.67	1.33
	4 → 7	1.00	2.00
	Rata-rata Nilai Lift Ratio		**1.77**
Bangkalan	11 → 8	0.75	1.00
	24 → 41	0.67	1.33
	41 → 10	0.67	1.33
	8 → 24	0.67	1.33
	Rata-rata Nilai Lift Ratio		**1.24**
Sampang	1 → 2	1.00	2.00
	10 → 15	0.67	1.33
	15 → 42	0.67	1.33
	22 → 32	1.00	2.00
	34 → 35	1.00	2.00
	4 → 8	1.00	2.00
	42 → 43	1.00	1.33
	43 → 1	0.67	1.33
	7→ 10	0.67	1.33
	8→ 9	1.00	2.00
	Rata-rata Nilai Lift Ratio		**1.66**
Pamekasan	10 → 12	1.00	2.00
	12 → 13	1.00	2.00
	19 → 33	0.67	1.33
	32 → 35	1.00	2.00
	41 → 42	1.00	2.00
	Rata-rata Nilai Lift Ratio		**1.86**
Sidoarjo	12 → 41	1.00	1.00
	25 → 36	0.67	1.33
	34 → 40	1.00	2.00

	36 → 39	0.67	1.33
	41 → 25	0.75	1.00
	Rata-rata Nilai Lift Ratio		**1.33**
Kab. Pasuruan	18 → 19	1.00	2.00
	19 → 22	1.00	2.00
	2 → 6	0.75	1.00
	35 → 39	1.00	2.00
	6 → 9	0.67	1.33
	9 → 24	0.67	1.33
	Rata-rata Nilai Lift Ratio		**1.61**
Kota Pasuruan	3 → 19	1.00	2.00
	8 → 17	1.00	2.00
	9 → 18	1.00	1.33
	Rata-rata Nilai Lift Ratio		**1.77**
Kab. Probolinggo	22 → 27	1.00	2.00
	7→ 9	0.67	1.33
	9 → 13	0.67	1.33
	Rata-rata Nilai Lift Ratio		**1.77**
Kota Probolinggo	1 → 21	1.00	1.00
	14 → 29	0.67	1.33
	16 → 25	1.00	2.00
	21 → 14	0.75	1.00
	25 → 31	1.00	2.00
	29 → 37	0.67	1.33
	31 → 33	1.00	2.00
	37 → 4	0.67	1.33
	6→ 12	1.00	2.00
	Rata-rata Nilai Lift Ratio		**1.55**
Situbondo	11 → 19	1.00	1.33
	18 → 41	1.00	1.00
	19 → 42	0.67	1.33
	3 → 11	0.67	1.33
	39 → 43	1.00	2.00
	41 → 3	0.75	1.00
	Rata-rata Nilai Lift Ratio		**1.33**
Banyuwangi	16 → 26	0.67	1.33
	26 → 28	0.67	1.33
	34 → 35	1.00	2.00
	Rata-rata Nilai Lift Ratio		**1.55**
Jember	11 → 44	0.67	1.33
	20 → 26	1.00	2.00
	29 → 32	1.00	2.00

	39 → 42	1.00	2.00
	44 → 4	0.67	1.33
	Rata-rata Nilai Lift Ratio		**1.73**
Lumajang	13 → 30	0.67	1.33
	25 → 26	1.00	2.00
	Rata-rata Nilai Lift Ratio		**1.66**
Malang	1 → 7	0.75	1.00
	10 → 13	1.00	2.00
	27 → 28	0.67	1.33
	28 → 43	0.67	1.33
	42 → 44	1.00	2.00
	43 → 8	0.67	1.33
	7 → 27	1.00	1.33
	Rata-rata Nilai Lift Ratio		**1.47**
Blitar	10 → 19	0.67	1.33
	19 → 29	1.00	1.33
	27 → 30	1.00	2.00
	29→ 34	0.67	1.33
	34 → 37	0.67	1.33
	37 → 1	0.67	1.33
	Rata-rata Nilai Lift Ratio		**1.44**
Tulungagung	13 → 23	0.67	1.33
	17 → 19	1.00	2.00
	19 → 22	1.00	2.00
	22 → 33	1.00	2.00
	23 → 30	0.67	1.33
	38 → 4	0.75	1.00
	39 → 41	1.00	2.00
	4 → 7	0.67	1.33
	7 → 13	0.67	1.33
	Rata-rata Nilai Lift Ratio		**1.59**
Trenggalek	11 → 12	0.67	1.33
	12 → 20	0.67	1.33
	3 → 4	1.00	1.33
	36 → 39	1.00	2.00
	4 → 11	0.67	1.33
	Rata-rata Nilai Lift Ratio		**1.46**
Pacitan	1 → 4	1.00	2.00
	13 → 18	1.00	2.00
	16 → 26	0.67	1.33
	18 → 20	1.00	2.00
	2 → 16	0.67	1.33

	21 → 23	1.00	2.00
	23 → 24	1.00	2.00
	34 → 39	1.00	2.00
	Rata-rata Nilai Lift Ratio		**1.83**
Muncar	15 → 24	0.75	1.00
	24 → 38	0.67	1.33
	38 → 43	0.67	1.33
	43 → 2	0.67	1.33
	Rata-rata Nilai Lift Ratio		**1.24**
Sumenep	10 → 20	1.00	1.00
	2 → 7	1.00	1.33
	20 → 2	0.75	1.00
	25 → 27	1.00	2.00
	27 → 30	1.00	2.00
	31 → 32	1.00	2.00
	4 → 5	1.00	2.00
	7 → 26	0.67	1.33