ALGORITMA *DECISION TREE* C4.5 UNTUK KLASIFIKASI KELUARGA PESERTA JAMKESMAS BERDASARKAN KEMISKINAN

SKRIPSI

Oleh:
SISCA YULIHARYANI
0510960057-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2011

ALGORITMA *DECISION TREE* C4.5 UNTUK KLASIFIKASI KELUARGA PESERTA JAMKESMAS BERDASARKAN KEMISKINAN

SKRIPSI

Sebagai salah satu syarat untuk meraih gelar Sarjana Komputer dalam bidang Ilmu Komputer

Oleh:
Sisca Yuliharyani
0510960057-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2011

LEMBAR PENGESAHAN SKRIPSI

ALGORITMA *DECISION TREE* C4.5 UNTUK KLASIFIKASI KELUARGA PESERTA JAMKESMAS BERDASARKAN KEMISKINAN

Oleh:
SISCA YULIHARYANI
0510960057-96

Setelah dipertahankan di depan Majelis Penguji
pada tanggal 18 Juli 2011
dan dinyatakan memenuhi syarat untuk memperoleh gelar
Sarjana Komputer dalam bidang Ilmu Komputer

Dosen Pembimbing I,	Dosen Pembimbing II,
Lailil Muflikhah, S.Kom, M.Sc.	Drs. Marji, MT
NIP. 197411132005012001	NIP. 196708011992031001

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas Brawijaya

Dr. Abdul Rouf Alghofari, MSc
NIP. 196709071992031001

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama **:** Sisca Yuliharyani
NIM **:** 0510960057-96
Jurusan **:** Matematika
Program Studi **:** Ilmu Komputer
Judul Tugas Akhir **:** Algoritma *Decision Tree* C4.5 Untuk Klasifikasi Keluarga Peserta Jamkesmas Berdasarkan Kemiskinan

Dengan ini menyatakan bahwa :

1. **Isi dari tugas akhir yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam Tugas Akhir ini.**
2. **Apabila dikemudian hari ternyata Tugas Akhir yang saya tulis terbukti hasil jiplakan, maka saya akan bersedia menanggung segala resiko yang akan saya terima.**

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang,
Yang menyatakan,

Sisca Yuliharyani
NIM. 0510960057-96

ALGORITMA *DECISION TREE* C4.5 UNTUK KLASIFIKASI KELUARGA PESERTA JAMKESMAS BERDASARKAN KEMISKINAN

ABSTRAK

Pertumbuhan yang pesat dari akumulasi data telah menciptakan kondisi kaya akan data tapi minim informasi. Data mining diharapkan mampu memanfaatkan kondisi tersebut, yaitu dari data yang jumlahnya sangat besar dapat menghasilkan informasi dari kumpulan data yang belum pernah dipelajari sebelumnya. Informasi ini nantinya dapat digunakan untuk menentukan kebijakan atau langkah selanjutnya pada suatu organisasi maupun institusi. Data mining merupakan suatu proses menemukan pengetahuan baru berdasarkan spesifikasi tertentu dari sekumpulan data. Penelitian ini memanfaatkan teknologi data mining dengan metode *decision tree* (pohon keputusan) menggunakan algoritma C4.5 untuk klasifikasi keluarga peserta Jaminan Kesehatan Masyarakat (Jamkesmas) berdasarkan kemiskinan karena Jamkesmas adalah salah satu program pemerintah yang saat ini mendapat sorotan dikarenakan program ini masih banyak yang salah sasaran. Seharusnya hanya keluarga miskin yang dapat memanfaatkan program itu, namun kenyataannya banyak dari mereka yang tergolong keluarga mampu dapat memanfaatkan program tersebut.

Penelitian ini akan menggunakan algoritma *decission tree* C4.5 dengan metode *pruning statistical bounds* untuk menentukan klasifikasi keluarga berdasarkan tingkat kemiskinan apakah memenuhi syarat menjadi peserta Jamkesmas atau tidak. Pada penelitian ini akan dibandingkan pula kinerja algoritma *decision tree* C4.5 antara tanpa *pruning* dan dengan *pruning* aturan *post pruning* menggunakan metode *statistical bounds* juga perbandingan kinerja *decision tree* C4.5 terhadap kombinasi atribut yang dipakai yang nantinya akan diperoleh kinerja terbaik dari kombinasi atribut tertentu. Sehingga dari penelitian ini diharapkan akan diperoleh hasil klasifikasi keluarga peserta Jamkesmas yang lebih akurat.

C4.5 DECISION TREE ALGORITHM TO CLASIFFY JAMKESMAS' FAMILIES PARTICIPANT BASED ON POVERTY LEVEL

ABSTRACT

The rapid growth of data accumulation has created a condition, rich of data but poor of information. Data mining is expected to take advantage from those condition which is large amounts of data can provide information about the database that has not been studied. Next, this information can be used to determine the next step or the next policy in an organization or institution. Data mining is a process to discover a new knowledge based on certain specification from a data sets. This research utilizing data mining technology with decision tree using C4.5 algorithm to classified the public health insurance's family member (Jamkesmas) based on poverty, because Jamkesmas is one of government's program which gets attention because that program still missed the target. In appropriately, only poor family can get the advantage of this program but in reality there is a lot of intermediate-rich family get the advantage from this program.

This research will use C4.5 algorithm decision tree with pruning statistical bounds method to determining the family classification based on poverty level whether appropriate to be a member of Jamkesmas or not . In this research will also compared the performance of the C4.5 decision tree algorithm with pruning or without pruning post pruning-rule using statistical bounds method, also the comparison of C4.5 decision tree used a combination of attributes that will be obtained the best performance of a particular attribute combination. So from this research, the researcher expect to get the classification results of Jamkesmas participating families more accurately.

KATA PENGANTAR

Alhamdulillah, segala puji bagi *Allah Subhanahu Wata'ala*, karena atas segala rahmat dan limpahan hidayahNya, skripsi yang berjudul "Algoritma *Decision Tree* C4.5 untuk Klasikfikasi Keluarga Peserta Jamkesmas berdasarkan Kemiskinan" ini dapat diselesaikan. Skripsi ini disusun dan diajukan sebagai syarat untuk memperoleh gelar sarjana pada program studi Ilmu Komputer, jurusan Matematika, fakultas MIPA, Universitas Brawijaya.

Dalam penyelesaian skripsi ini, penulis telah mendapat begitu banyak bantuan baik moral maupun material dari banyak pihak. Atas bantuan yang telah diberikan, penulis ingin menyampaikan penghargaan dan ucapan terima kasih yang sedalam-dalamnya kepada:

1. Lailil Muflikhah, S.Kom, MSc dan Drs. Marji, MT selaku dosen pembimbing, dan atas semua saran, bantuan, kritikan, waktu, dorongan semangat dalam bimbingannya
2. Drs. Marji, MT selaku Ketua Program Studi Ilmu Komputer Universitas Brawijaya Malang
3. Wayan Firdaus Mahmudy, SSi, MT selaku Penasihat Akademik
4. Putriku, Aisyah Nur Shabrina dan Akang Aries Syamsuddin atas segala dukungan dan semangatnya, serta kesabarannya menunggu ummi menyelesaikan studi
5. Ummi dan Abi atas bantuannya baik material maupun spiritual
6. Bapak dan ibu mertua atas dukungan dan doanya
7. Mbin dan Mban atas inspirasi yang telah diberikan
8. Segenap pimpinan dan staf di lingkungan Pemerintah Daerah Kabupaten Blitar atas kerjasama dan dukungannya terhadap penelitian penulis
9. Teman-teman Ilmu Komputer khususnya angkatan 2005 dan 2007 atas kebersamaannya.

Semoga penulisan skripsi ini bermanfaat bagi pembaca sekalian. Tak ada gading yang tak retak, sebagai manusia, penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan, dan mengandung banyak kekurangan, sehingga dengan segala kerendahan hati penulis mengharapkan kritik dan saran yang membangun dan melengkapi dari pembaca.

Malang, 18 Juli 2011

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR *SOURCE CODE*

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Seiring dengan kemajuan dalam pengumpulan data dan teknologi penyimpanan yang cepat, sekarang ini memungkinkan suatu organisasi menghimpun jumlah data yang sangat besar. Alat dan teknik analisis data yang tradisional tidak dapat lagi digunakan untuk mempermudah dalam memperoleh informasi dari data yang sangat besar ini. Untuk itu, diperlukan suatu metode baru yang dapat menjawab kebutuhan tersebut. Metode baru ini adalah data mining yang merupakan teknologi dimana terdapat penggabungan antara metode analisis tradisional dengan algoritma untuk memproses data berjumlah besar.

Data mining merupakan teknologi yang sangat berguna untuk membantu menemukan informasi yang sangat penting, melalui penggunaan data mining dapat diramalkan tren dan sifat-sifat perilaku yang sangat berguna untuk mendukung pengambilan keputusan penting. Analisis yang dilakukan oleh data mining melebihi yang dilakukan oleh sistem pendukung sistem pengambilan keputusan tradisional yang sudah banyak digunakan. Data mining dapat menjawab pertanyaan-pertanyaan yang apabila dilakukan dengan cara tradisional memerlukan banyak waktu dan biaya yang tinggi. Data mining mengeksplorasi basis data untuk menemukan pola-pola yang tersembunyi, mencari informasi untuk memprediksi hal-hal yang mungkin saja terlupakan oleh para pengguna disebabkan karena terletak di luar ekspektasi sehingga data mining sangat bermanfaat bagi manusia untuk mengambil keputusan dalam berbagai bidang.

Salah satu manfaat data mining dalam bidang pemerintahan, yaitu dapat digunakan sebagai bahan untuk mendukung pengambilan keputusan dan menunjang program kegiatan pemerintah. Salah satu program pemerintah yang saat ini mendapat sorotan adalah Jaminan Kesehatan Masyarakat (Jamkesmas). Masyarakat luas menilai program ini masih banyak yang salah sasaran. Seharusnya hanya keluarga miskin yang dapat memanfaatkan program itu, namun kenyataannya banyak dari mereka yang tergolong keluarga mampu dapat memanfaatkan program tersebut. Padahal di sisi lain, keluarga yang benar-benar miskin malah tidak mendapat fasilitas tersebut.

Dalam data mining, terdapat berbagai fungsi yang salah satunya merupakan fungsi klasifikasi. Teknik yang dapat digunakan untuk menentukan klasifikasi adalah algoritma *Decision Tree* C4.5 yang merupakan algoritma klasifikasi data dan prediksi dengan teknik pohon keputusan yang terkenal dan banyak digunakan karena memiliki kelebihan-kelebihan yaitu (Ruggieri, 2001):

1. Dapat mengolah data numerik dan diskrit
2. Dapat menangani nilai atribut yang hilang
3. Menghasilkan aturan-aturan yang mudah diinterpretasikan dan tercepat di antara algoritma-algoritma yang menggunakan memori utama di komputer.

Penelitian sebelumnya yang dilakukan oleh Bagus Hari Mahardika (2009) bahwa algoritma *decision tree* C4.5 dapat digunakan dalam melakukan klasifikasi data. Untuk menghasilkan *decision rule* yang lebih efektif dan sederhana harus dilakukan *pruning* (pemangkasan *tree*). Pada penelitian tersebut, pruning yang digunakan adalah aturan *post pruning* dengan metode *statistical bounds* dan *pessimistic error pruning*. Diperoleh hasil bahwa metode *statistical bounds* memiliki hasil yang lebih akurat (Bagus H M, 2009).

Pada penelitian ini, nantinya akan menggunakan algoritma *decission tree* C4.5 dengan metode *pruning statistical bounds* untuk menentukan klasifikasi keluarga berdasarkan tingkat kemiskinan apakah memenuhi syarat menjadi peserta Jamkesmas atau tidak. Perbedaan antara penelitian ini dengan sebelumnya yaitu:

1. Membandingkan kinerja algoritma *decision tree* C4.5 dengan *pruning* metode *statistical bounds* dan tanpa *pruning*
2. Membandingkan kinerja *decision tree* C4.5 terhadap kombinasi atribut yang dipakai sehingga diperoleh kinerja terbaik dari kombinasi atribut tertentu.

Sehingga dari penelitian ini diharapkan akan diperoleh hasil klasifikasi keluarga berdasarkan tingkat kemiskinan yang lebih akurat dan pada akhirnya salah sasaran dari program Jamkesmas dapat diminimalisir.

1.2 Perumusan Masalah

Berdasarkan dari uraian latar belakang, permasalahan yang dapat dirumuskan yaitu:

- Bagaimana menentukan klasifikasi keluarga peserta Jamkesmas berdasarkan kemiskinan di Kabupaten Blitar dengan algoritma *decision tree* C4.5?
- Bagaimana kinerja algoritma *decision tree* C4.5 untuk menentukan klasifikasi keluarga peserta Jamkesmas berdasarkan kemiskinan di Kabupaten Blitar?
- Kombinasi atribut mana saja yang menghasilkan kinerja paling baik untuk menentukan klasifikasi keluarga peserta jamkesmas berdasarkan kemiskinan di Kabupaten Blitar?

1.3 Batasan Masalah

Batasan masalah pada penulisan skripsi ini adalah sebagai berikut:

- Skripsi ini hanya mengulas algoritma *decision tree* C4.5 sebagai fungsi klasifikasi pada data mining
- Data yang digunakan adalah data kependudukan Desa Candirejo Kecamatan Ponggok Kabupaten Blitar bulan Desember tahun 2010
- Tidak mengatasi adanya data yang tidak ada nilainya (*missing value*)
- Semua atribut yang digunakan bertipe kategoris
- *Pruning* yang digunakan adalah aturan *post pruning* dengan metode *statistical bounds*
- Pengukuran kinerja *decision tree* C4.5 menggunakan data uji yang berbeda, penggunaan metode *pruning*, dan kombinasi atribut yang digunakan.

1.4 Tujuan

Dengan mengacu pada perumusan dan batasan masalah, maka tujuan yang ingin dicapai pada skripsi ini yaitu:

- Mengimplementasikan algoritma *decision tree* C4.5 untuk menentukan klasifikasi keluarga peserta Jamkesmas berdasarkan kemiskinan di Kabupaten Blitar sehingga hasilnya dapat digunakan sebagai bahan rekomendasi untuk program Jaminan Kesehatan Masyarakat
- Mengetahui kinerja algoritma *decision tree* C4.5 untuk menentukan klasifikasi keluarga peserta Jamkesmas berdasarkan kemiskinan di Kabupaten Blitar
- Mengetahui kombinasi atribut yang terbaik untuk

menentukan klasifikasi keluarga peserta Jamkesmas berdasarkan kemiskinan di Kabupaten Blitar.

1.5 Sistematika Penulisan

Sistematika penulisan yang digunakan dalam penyusunan skripsi ini sebagai berikut:

BAB I PENDAHULUAN

Bab ini membahas tentang latar belakang permasalahan, perumusan masalah, batasan masalah, tujuan, dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini membahas tentang teori-teori yang mendasari perancangan dan pembuatan tulisan ini.

BAB III METODOLOGI DAN PERANCANGAN

Bab ini membahas tahapan-tahapan yang dilalui dalam pembuatan skripsi ini, mulai dari perancangan, ujicoba, dan pengambilan data hasil.

BAB IV

Bab ini membahas implementasi dari data mining, evaluasi hasil ujicoba, dan analisa hasil ujicoba.

BAB V PENUTUP

Bab ini membahas uraian kesimpulan atas analisa hasil ujicoba dan saran-saran yang dapat berguna untuk penyempurnaan skripsi ini.

BABII
LANDASAN TEORI

2.1 Jamkesmas

Undang-undang Dasar 1945 pasal 28H dan Undang-undang No.23 Tahun 1992 tentang kesehatan, menetapkan bahwa setiap orang berhak mendapat pelayanan kesehatan. Karena itu, setiap individu, keluarga, dan masyarakat berhak memperoleh perlindungan terhadap kesehatannya dan Negara bertanggung jawab mengatur agar terpenuhi hak hidup sehat bagi penduduknya termasuk bagi masyarakat miskin dan tidak mampu. Untuk menjamin akses penduduk miskin terhadap pelayanan kesehatan sebagaimana diamanatkan dalam Undang-undang Dasar 1945 tersebut pemerintah telah berupaya untuk membuat kebijakan melalui Program Pemeliharaan Kesehatan Masyarakat Miskin. Program ini berupa bantuan sosial untuk pelayanan kesehatan bagi masyarakat miskin dan tidak mampu yang diselenggarakan secara nasional agar terjadi subsidi silang dalam rangka mewujudkan pelayanan kesehatan yang menyeluruh bagi masyarakat miskin, selanjutnya disebut Program Jaminan Kesehatan Masyarakat (Jamkesmas).

Rendahnya derajat kesehatan masyarakat menjadi persoalan mendasar bangsa ini, di tengah terpaan beban ekonomi serta mahalnya biaya untuk mendapatkan pelayanan kesehatan yang bermutu, membuat masyarakat miskin semakin sulit menjangkau layanan kesehatan. Hal tersebut membuat pemerintah terus berupaya untuk mengambil langkah kebijakan untuk mengatasinya.

Pada awal Tahun 2008, pemerintah melalui Departemen Kesehatan RI telah mengeluarkan program kebijakan di bidang kesehatan, yakni kebijakan Jaminan Kesehatan Masyarakat. Kebijakan tersebut dituangkan dalam surat Keputusan Menteri Kesehatan RI Nomor 125/MENKES/SK/II/2008. Program Jamkesmas ini disuguhkan untuk memberikan akses pelayanan kesehatan secara gratis baik di tingkat Puskesmas maupu Rumah Sakit. Kebijakan tersebut bersifat nasional, sehingga kebijakan ini akan diselenggarakan di semua daerah di tanah air.Kebijakan tersebut disuguhkan untuk warga miskin dan tidak mampu. Namun, untuk menentukan sasaran tersebut, Departemen Kesehatan RI memberikan kuota yang berbeda-beda pada setiap kabupaten dan kota berdasarkan angka kemiskinannya.

Tujuan penyelenggaraan Program Jamkesmas secara umum Jamkesmas dibangun untuk memberikan akselerasi dalam peningkatan akses dan mutu pelayanan kesehatan terhadap seluruh masyarakat miskin dan tidak mampu agar tercapai derajat kesehatan masyarakat yang optimal secara efektif dan efisien. Secara khusus program ini ditujukan untuk meningkatkan cakupan masyarakat miskin dan tidak mampu guna mendapat pelayanan kesehatan di Puskesmas serta jaringannya dan di Rumah Sakit. Melalui program ini pula diharapkan akan terjadi proses penyelenggaraan pengelolaan keuangan yang transparan dan akuntabel yang pada akhirnya akan berdampak kepada peningkatan kualitas pelayanan kesehatan bagi masyarakat miskin (Budihardja, 2009).

2.2 Data Mining

Data mining adalah serangkaian proses untuk menggali nilai tambah berupa informasi yang selama ini tidak diketahui secara manual dari suatu basis data. Informasi yang dihasilkan diperoleh dengan cara mengekstraksi dan mengenali pola yang penting atau menarik dari data yang terdapat dalam basis data. Data mining terutama digunakan untuk mencari pengetahuan yang terdapat dalam basis data yang besar sehingga sering disebut *Knowledge Discovery in Database* (KDD). Proses pencarian pengetahuan ini menggunakan berbagai teknik pembelajaran komputer (*machine learning*) untuk menganalisis dan mengekstraksikannya. Proses pencarian bersifat iteratif dan interaktif untuk menemukan pola atau model yang benar, baru, bermanfaat, dan dimengerti. Dalam penerapannya, data mining memerlukan berbagai perangkat lunak analisis data untuk menemukan pola dan relasi data agar dapat digunakan untuk membuat prediksi dan klasifikasi dengan akurat (Tan dkk, 2004).

Kehadiran data mining dilatarbelakangi oleh berlimpahnya data (*overload data*) yang dialami oleh berbagai institusi, perusahaan atau organisasi. Berlimpahnya data ini merupakan akumulasi data transaksi yang terekam bertahun-tahun. Data-data tersebut merupakan data transaksi yang umumnya diproses menggunakan aplikasi komputer yang biasa disebut dengan *On Line Transaction Processing* (OLTP). Data mining juga dilatarbelakangi oleh adanya ledakan informasi (*explosion information*) dari berbagai media terutama internet. Delapan puluh persen informasi yang disajikan media internet adalah dalam bentuk tak terstruktur. Media internet

menyajikan informasi dalam berbagai format file, bahasa, dan bentuk penyajian seperti teks, gambar, suara ataupun video. Pertumbuhan yang pesat dari akumulasi data atau informasi itu telah menciptakan kondisi dimana suatu intuisi memiliki bergunung-gunung data tetapi miskin informasi yang bermanfaat (*rich of data but poor of information*) (Tan dkk, 2004).

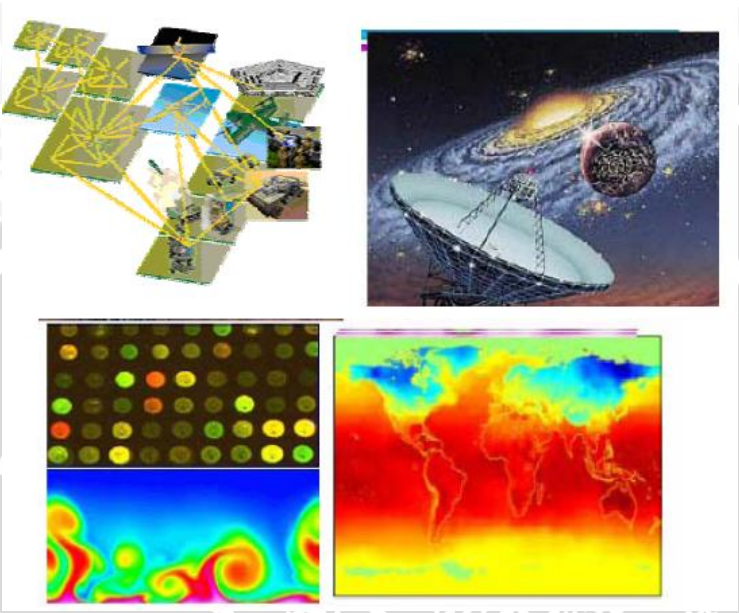
Pemanfaatan data mining diperlukan untuk menangani tumpukan data yang besar, namun sering kali tumpukan data yang besar ini dibiarkan saja tanpa dilakukan upaya untuk menggali informasi lebih jauh. Seakan-akan tumpukan data dalam jumlah yang besar tersebut tidak memiliki manfaat sama sekali. Pemanfaatan data itu dapat dilihat dalam dua sudut pandang, yaitu sudut pandang komersial dan sudut pandang keilmuan. Dari sudut pandang komersial, pemanfaatan data mining dapat digunakan untuk menangani meledaknya volume data. Terkait dengan cara menyimpannya, mengestraknya serta memanfaaatkannya. Berbagai teknik komputasi dapat digunakan untuk menghasilkan informasi yang dibutuhkan. Informasi yang dihasilkan menjadi asset untuk meningkatkan daya saing suatu intuisi. Data mining tidak hanya digunakan untuk menangani persoalan menumpuknya data atau informasi dan bagaimana menggudangkannya tanpa kehilangan informasi yang penting. Data mining juga diperlukan untuk menyelesaikan permasalahan atau menjawab kebutuhan bisnis itu sendiri, misalnya (Tan dkk, 2004):

a. Bagaimana mengetahui hilangnya pelanggan karena pesaing
b. Bagaimana mengetahui item produk atau konsumen yang memiliki kesamaan karakteristik
c. Bagaimana mengidentifikasi produk-produk yang terjual bersamaan dengan produk lain
d. Bagaimana memprediksi tingkat penjualan
e. Bagaimana menilai tingkat resiko dalam menentukan jumlah produksi suatu item
f. Bagaimana memprediksi perilaku bisnis di masa yang akan datang.

Dari sudut pandang keilmuan, data mining dapat digunakan untuk meng*capture*, menganalisis, serta menyimpan data yang bersifat *real time* dan sangat besar, misalnya:

a. *Remote* sensor yang ditempatkan pada suatu satelit
b. Teleskop yang digunakan untuk memindai langit

c. Simulasi saintifik yang membangkitkan data dalam ukuran *terabytes.*



Gambar 2.1: Contoh pemanfaatan data mining di bidang keilmuan (Tan dkk, 2004)

Data mining merupakan salah satu metode alternatif yang dapat digunakan untuk mengolah data mentah, ketika metode konvensional tidak mungkin untuk dilakukan karena besarnya volume data yang diolah. Hal ini dapat terjadi karena data mining memiliki kemampuan mereduksi data baik melalui teknik katalogisasi, klasifikiasi, maupun segmentasi (Tan dkk, 2004).

Data mining sesungguhnya merupakan salah satu rangkaian dari proses pencarian pengetahuan pada database atau *Knowledge Discovery in Database* (KDD). KDD berhubungan dengan teknik integrasi dan penemuan ilmiah, interpretasi, dan visualisasi dari pola-pola sejumlah kumpulan data. KDD adalah keseluruhan proses non trivial untuk mencari dan mengidentifikasi pola (*pattern*) dalam data, dimana pola yang ditemukan bersifat sah, baru, dapat bermanfaat dan dapat dimengerti. Serangkaian proses tersebut yang memiliki tahap sebagai berikut (Tan dkk, 2004):

1. Pembersihan data dan integrasi data

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

2. Seleksi dan transformasi data

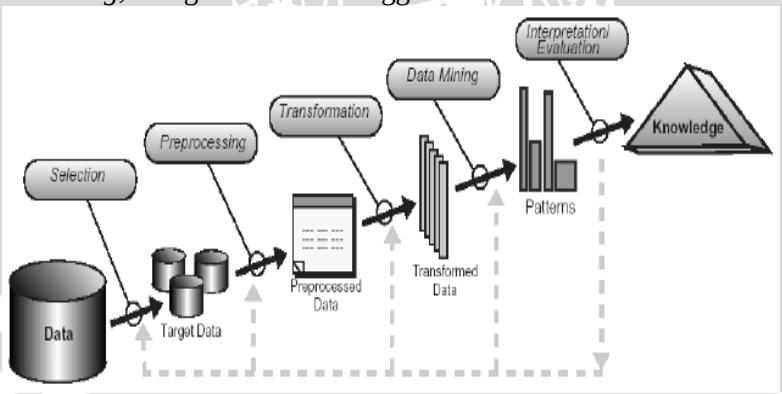
Data yang terdapat dalam basis data kemudian direduksi dengan berbagai teknik. Proses reduksi diperlukan untuk mendapatkan hasil yang lebih akurat dan mengurangi waktu komputasi terutama untuk masalah dengan skala besar

Beberapa cara seleksi, antara lain :

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

Transformasi data diperlukan sebagai tahap *preprocessing*, dimana data yang diolah siap untuk ditambang. Beberapa cara transformasi, antara lain (Santosa, 2007):

a. *Centering*, mengurangi setiap data dengan rata-rata dari setiap atribut yang ada.
b. *Normalization*, membagi setiap data yang di*centering* dengan standar deviasi dari atribut bersangkutan.
c. *Scaling*, mengubah data sehingga berada dalam skala tertentu.



Gambar 2.2 : Tahap-Tahap *Data Mining* (Han, 2006)

3. Penambangan data (*data mining*)

Data-data yang telah diseleksi dan ditransformasi ditambang

dengan berbagai teknik. Proses data mining adalah proses mencari pola atau informasi menarik dalam data terpilih dengan menggunakan fungsi-fungsi tertentu. Fungsi atau algoritma dalam data mining sangat bervariasi. Pemilihan fungsi atau algoritma yang tepat sangat bergantung pada tujuan dan proses pencaraian pengetahuan secara keseluruhan

4. Evaluasi pola dan presentasi pengetahuan

Tahap ini merupakan bagian dari proses pencarian pengetahuan yang mencakup pemeriksaan apakah pola atau informasi yang ditemukan bertentangan dengan fakta atau hipotesa yang ada sebelumnya. Langkah terakhir KDD adalah mempresentasikan pengetahuan dalam bentuk yang mudah dipahami oleh pengguna.

Fungsi-fungsi yang umum diterapkan dalam data mining (Haskett, 2000):

a. *Association*, adalah proses untuk menemukan aturan assosiatif antara suatu kombinasi item dalam suatu waktu
b. *Sequence*, hampir sama dengan *association* bedanya *sequence* diterapkan lebih dari satu periode
c. *Clustering*, adalah proses pengelompokan sejumlah data atau obyek ke dalam kelompok-kelompok data (klaster) sehingga setiap klaster akan berisi data yang saling mirip
d. *Classification*, adalah proses penemuan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data dengan tujuan untuk dapat memperkirakan kelas dari suatu objek yang labelnya tidak diketahui
e. *Regretion*, adalah proses pemetaan data dalam suatu nilai prediksi
f. *Forecasting*, adalah proses pengestimasian nilai prediksi berdasarkan pola-pola di dalam sekumpulan data
g. *Solution*, adalah proses penemuan akar masalah dan *problem solving* dari persoalan bisnis yang dihadapi atau paling tidak sebagai informasi pendukung dalam pengambilan keputusan.

2. 3 Algoritma *Decision Tree* C4.5

Pohon Keputusan (*Decision Tree*) merupakan metode klasifikasi dan prediksi yang sangat kuat dan terkenal. Metode pohon keputusan mengubah fakta yang sangat besar menjadi pohon keputusan yang merepresentasikan aturan. Aturan dapat dengan mudah dipahami dengan bahasa alami. Aturan ini juga dapat

diekspresikan dalam bentuk bahasa basis data seperti SQL untuk mencari *record* pada kategori tertentu. Pohon keputusan juga berguna untuk mengeksplorasi data, menemukan hubungan tersembunyi antara sejumlah calon variabel input dengan sebuah variabel target. Karena pohon keputusan memadukan antara eksplorasi data dan pemodelan, pohon keputusan ini sangat bagus sebagai langkah awal dalam proses pemodelan bahkan ketika dijadikan sebagai model akhir dari beberapa teknik lain(J R Quinlan, 1993).

Dalam situasi lain kemampuan untuk menjelaskan alasan pengambilan keputusan adalah sesuatu yang sangat penting. Misalnya pada perusahaan asuransi ada larangan resmi untuk mendeskriminasi berdasarkan variabel-variabel tertentu. Perusahaan asuransi dapat mencari sendiri keadaan yang mencerminkan bahwa mereka tidak menggunakan deskriminasi yang ilegal dalam memutuskan seseorang diterima atau ditolak. Sebuah pohon keputusan adalah sebuah struktur yang dapat digunakan untuk membagi kumpulan data yang besar menjadi himpunan-himpunan *record* yang lebih kecil dengan menerapkan serangkaian aturan keputusan. Anggota himpunan hasil menjadi mirip satu dengan yang lain dengan masing-masing rangkaian pembagian. Sebuah model pohon keputusan terdiri dari sekumpulan aturan untuk membagi sejumlah populasi yang heterogen menjadi lebih kecil, lebih homogen dengan memperhatikan pada variabel tujuannya. Sebuah pohon keputusan mungkin dibangun dengan seksama secara manual, atau dapat tumbuh secara otomatis dengan menerapkan salah satu atau beberapa algoritma pohon keputusan untuk memodelkan himpunan data yang belum terklasifikasi (Tan dkk, 2004).

Variabel tujuan biasanya dikelompokkan dengan pasti dan model pohon keputusan lebih mengarah pada perhitungan probabilitas dari masing-masing *record* terhadap kategori-kategori tersebut, atau untuk mengklasifikasi *record* dengan mengelompokkannya dalam satu kelas. Pohon keputusan juga dapat digunakan untuk mengestimasi nilai dari variabel kontinyu, meskipun ada beberapa teknik yang lebih sesuai untuk kasus ini.

Kelebihan dari metode pohon keputusan adalah:

a. Daerah pengambilan keputusan yang sebelumnya kompleks dan sangat global, dapat diubah menjadi lebih simpel dan spesifik
b. Eliminasi perhitungan-perhitungan yang tidak diperlukan, karena

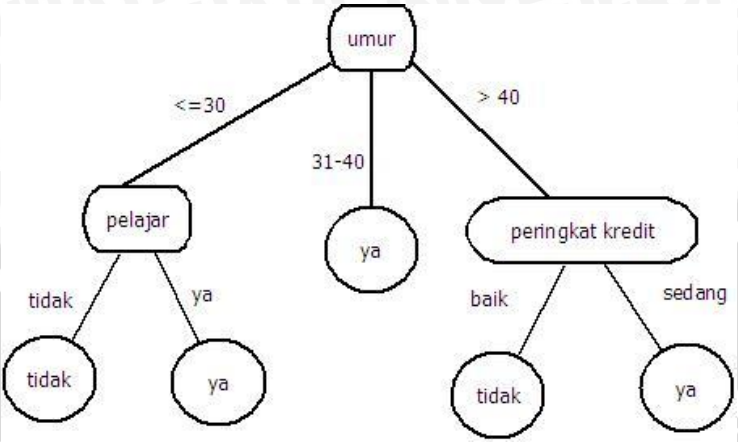
ketika menggunakan metode pohon keputusan maka sampel diuji hanya berdasarkan kriteria atau kelas tertentu

c. Fleksibel untuk memilih fitur dari *node* internal yang berbeda, fitur yang terpilih akan membedakan suatu kriteria dibandingkan kriteria yang lain dalam *node* yang sama. Kefleksibelan metode pohon keputusan ini meningkatkan kualitas keputusan yang dihasilkan jika dibandingkan ketika menggunakan metode penghitungan satu tahap yang lebih konvensional
d. Dalam analisis multivarian, dengan kriteria dan kelas yang jumlahnya sangat banyak, seorang penguji biasanya perlu mengestimasikan baik itu distribusi dimensi tinggi ataupun parameter tertentu dari distribusi kelas tersebut. Metode pohon keputusan dapat menghindari munculnya permasalahan ini dengan menggunakan kriteria yang jumlahnya lebih sedikit pada setiap *node* internal tanpa banyak mengurangi kualitas keputusan yang dihasilkan.

Kekurangan pada pohon keputusan adalah:

a. Terjadi *overlapping* terutama ketika kelas-kelas dan kriteria yang digunakan jumlahnya sangat banyak. Hal tersebut juga dapat menyebabkan meningkatnya waktu pengambilan keputusan dan jumlah memori yang diperlukan
b. Pengakumulasian jumlah kesalahan dari setiap tingkat dalam sebuah pohon keputusan yang besar
c. Kesulitan dalam mendesain pohon keputusan yang optimal
d. Hasil kualitas keputusan yang didapatkan dari metode pohon keputusan sangat tergantung pada bagaimana pohon tersebut didesain.

Pohon keputusan adalah model prediksi menggunakan struktur pohon atau struktur berhirarki. Contoh dari pohon keputusan dapat dilihat pada Gambar 2.3



Gambar 2.3 Model Pohon Keputusan (Pramudiono,2008)

Setiap percabangan menyatakan kondisi yang harus dipenuhi dan tiap ujung pohon menyatakan kelas data. Contoh pada Gambar 2.3 adalah identifikasi pembeli komputer. Dari pohon keputusan tersebut diketahui bahwa salah satu kelompok yang potensial membeli komputer adalah orang yang berusia di bawah 30 tahun dan juga pelajar. Setelah sebuah pohon keputusan dibangun maka dapat digunakan untuk mengklasifikasikan *record* yang belum ada kelasnya. Dimulai dari *node root*, menggunakan tes terhadap atribut dari *record* yang belum ada kelasnya ini lalu mengikuti cabang yang sesuai dengan hasil dari tes tersebut, yang akan membawa kepada *internal node* (*node* yang memiliki satu cabang masuk dan dua atau lebih cabang yang keluar), dengan cara harus melakukan tes lagi terhadap atribut atau *node leaf*. *Record* yang kelasnya tidak diketahui kemudian diberikan kelas yang sesuai dengan kelas yang ada pada *node leaf*. Pada pohon keputusan setiap simpul *leaf* menandai label kelas. Proses dalam pohon keputusan yaitu mengubah bentuk data (tabel) menjadi model pohon (*tree*) kemudian mengubah model pohon tersebut menjadi aturan (*rule*) (J R Quinlan, 1993).

Salah satu algoritma induksi pohon keputusan yaitu ID3 (*Iterative Dichotomiser* 3). ID3 dikembangkan oleh J. Ross Quinlan. Dalam prosedur algoritma ID3, input berupa sampel training, label training dan atribut. Algoritma *Decision Tree* C4.5 merupakan

pengembangan dari ID3. Sedangkan pada perangkat lunak *open source* WEKA mempunyai versi sendiri dari C4.5 yang dikenal sebagai J48.

Berikut ini adalah dasar algoritma C4.5 untuk proses pembentukan *decision tree* (Han dan Khamber, 2001) :

Input : *Training* samples, *Atribute*
Output : *Decision tree*
Generate_decision_tree (Training samples, Atribute) // decision tree function
Method :
(1) Create node N;
(2) **If** samples are all of the same class C **then**
(3) **Return** N as a leaf node labeled with the class C;
(4) **if** atribute-list is empty **then**
(5) **Return** N as a leaf node labeled with the most common class in samples; // majority voting
(6) **else**
(7) select test-atribute, atribute among atribute-list with the highest information gain;
(8) label node N with test-atribute;
(9) **for** each known value $\mathbf{a}_i$ of test-atribute // partition the samples
(10) grow a branch from node N for the condition test-atribute = $\mathbf{a}_i$;
(11) let $\mathbf{s}_i$ be the set of samples in samples **for** which test-atribute = $\mathbf{a}_i$; // a partition
(12) **if** $\mathbf{s}_i$ is empty **then**
(13) attach a leaf labeled with the most common class in samples;
else attach the node returned by **Generate_decision_tree**($\mathbf{s}_i$, atribute-list-test-atribute);

Gambar 2.4 Algoritma *Decision Tree* C4.5 (Han dan Khamber, 2001)

Secara umum algoritma *Decision Tree* C4.5 untuk membangun pohon keputusan adalah sebagai berikut (Kusrini, 2009) :

a. Pilih atribut sebagai *root*
b. Buat cabang untuk masing-masing nilai
c. Bagi atribut terpilih dalam cabang
d. Ulangi proses untuk masing-masing cabang sampai semua

atribut terpilih pada cabang memiliki kelas yang sama.

Untuk menghitung gain diberikan rumus sebagai berikut:

$$Gain(S,A) = Entropi(S) - \sum_{i=1}^{n} \frac{|P_i|}{|S|} * Entropi(Si) \ldots\ldots (2.1)$$

Dimana
S = himpunan kasus
A = Atribut
n = jumlah partisi
|Si| = jumlah kasus pada partisi ke-i
|S| = jumlah kasus dalam S

Sedangkan untuk perhitungan nilai entropi adalah sebagai berikut:

$$Entropy(S) = \sum_{i=1}^{n} - pi * log_2 pi \qquad \ldots\ldots (2.2)$$

Dimana,
S = himpunan kasus
n = jumlah pastisi S
pi = proposi dari Si terhadap S

2.4 Pemangkasan *Tree* (pruning)

Pruning merupakan bagian dari proses pembentukan *Decision Tree*. Saat pembentukan *Decision Tree*, beberapa *node* merupakan *outlier* maupun hasil dari *noise* data. Penerapan *pruning* pada *Decision Tree*, dapat mengurangi *outlier* maupun *noise* data pada *Decision Tree* awal sehingga dapat mempercepat proses klasifikasi data (Han & Kamber 2006). Oleh sebab itu pemilihan algoritma *pruning* yang tepat perlu dilakukan untuk mendapat hasil klasifikasi yang maksimal.

Post-Pruning merupakan pemangkasan yang menumbuhkan pohon sampai pohon terbentuk secara lengkap lalu memangkasnya setelah pohon terbangun semuanya dengan menggunakan dua cara pemangkasan yaitu pemangkasan mulai dari *subtree* atau percabangan paling atas (*top-down*) dan pemangkasan mulai dari *subtree* terbawah (*bottom-up*). Memangkas suatu *subtree* atau

beberapa *subtree* dan menggantikannya dengan sebuah *leaf* lalu menyederhanakan sebuah pohon keputusan atau *Decision Tree*

Statistical bounds merupakan metode perhitungan *error* yang menerapkan aturan *post-pruning*, dimana perhitungannya berdasarkan jumlah label data pada suatu *node* dan batasan statistika (*upper bound)*. Dalam statistika, *upper bound* digunakan untuk menentukan batas yang dimiliki suatu nilai karena jika akan menentukan suatu nilai maka nilai tersebut harus memiliki batas maksimal begitu juga nilai *error* pada *node* sebuah *tree*. Hal ini yang ingin direpresentasikan oleh metode *statistical bound* untuk menghitung jumlah *error* yang dimiliki oleh setiap *node leaf* dan *subtree*. Batas atas atau *upper bound* akan digunakan sebagai standar *error* pada masing-masing *node*. Karakteristik dari s*tatistical bounds* (Kantardzic, 2003):

1. Aturan pemangkasan yang digunakan adalah *post-pruning*.
2. Pemangkasan dilakukan mulai dari percabangan terbawah atau *bottom-up*.
3. Standar *error* digunakan pada saat menentukan *error* di sebuah *subtree* dan *node leaf*.

Untuk lebih jelasnya dapat dilihat pada persamaan 2.3.

$$E = N \times e_{Upper} \quad \ldots\ldots\ldots\ldots (2.3)$$

Dimana E adalah *error* keseluruhan atau *error* total, N adalah jumlah semua kelas pada suatu *node* dan e_{Upper} adalah batas atas atau *upper bound* untuk *error rate* di suatu *node* yang digunakan sebagai standar *error*. Sedangkan rumus *upper bound* ditunjukkan pada persamaan 2.4

$$e_{upper}(N, e, \alpha) = \frac{e + \frac{z^2}{2N} + z\sqrt{\frac{e(1-e)}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}} \quad \ldots (2.4)$$

Dimana nilai z = nilai standarisasi yang didapat dari distribusi binomial (untuk C4.5 dengan nilai *confidence* atau α = 25% maka z = 0.69), e = *error rate* pada suatu *node*, dan N = jumlah *instance* pada suatu *node* (Kantardzic, 2003). Untuk rumus mencari *error rate* suatu *node* ditunjukkan pada Persamaan 2.5

$$e = \frac{e(t)}{N} \qquad \ldots\ldots(2.5)$$

Dimana $e(t)$ adalah jumlah kelas dalam suatu *node* yang tidak terklasifikasi dan N adalah jumlah semua kelas pada suatu *node*.

2.5 Pengukuran Kinerja

A. *Precision, Recall* dan *Accuracy*

Untuk permasalahan dalam klasifikasi, pengukuran yang biasa digunakan adalah *precision, recall* dan *accuracy* (Jyh-Jian Sheu, 2008). Hal ini dapat dilihat pada tabel berikut :

Tabel 2.1 Pengukuran kinerja

	Diidentifikasi sesuai (relevan)	Diidentifikasi tidak sesuai(tidak relevan)
Data yang diharapkan	A	B
Data tidak diharapkan	C	D

Dimana A adalah data yang diharapkan dan diidentifikasi sesuai dengan yang diharapkan (relevan), B adalah Data yang diharapkan tetapi diidentifikasi tidak sesuai dengan yang diharapkan (tidak relevan), C adalah Data yang tidak diharapkan tetapi diidentifikasi sesuai dengan yang diharapkan (relevan), dan D adalah Data yang tidak diharapkan dan diidentifikasi tidak sesuai dengan yang diharapkan (tidak relevan).

a. Precision

Precision adalah bagian data yang di ambil sesuai dengan informasi yang dibutuhkan. *Precision* pada permasalahan ini dapat juga disebut sebagai *positive predictive value* atau nilai prediksi yang positif. Rumus *precision* adalah :

$$Precision = (A / (A + B)) \times 100\ \% \qquad \ldots\ldots(2.6)$$

b. *Recall*

Recall adalah pengambilan data yang berhasil dilakukan terhadap bagian data yang relevan dengan *query*. *Recall* disebut

juga dengan *sensitivity*. Peluang munculnya data relevan yang diambil sesuai dengan query dapat dilihat dengan *recall.*
Rumus *Recall* adalah :

Recall = (A / (A + C)) x 100 % .....(2.7)

c. *Accuracy*

Accuracy adalah persentase dari total klasifikasi yang benar diidentifikasi.
Rumus *Accuracy* adalah :

Accuracy = ((A + D) / Total Data) * 100 % .....(2.8)

B. Confidence

Confidence adalah tingkat kepercayaan. Hal ini merupakan sebuah tingkat kepercayaan untuk *decision tree* yang terbentuk dari proses *training*.
Rumus *confidence* adalah :

Confidence = Jumlah anggota kelas / total data .....(2.9)

Rata-rata *confidence* adalah jumlah nilai *confidence* dibagi banyaknya data pada suatu proses *testing*.

C. Waktu Eksekusi *Testing*

Waktu eksekusi *testing* adalah lama proses terjadinya *testing* yaitu, ketika data *testing* diambil sampai proses *testing* selesai dan didapatkan hasil kinerja.

BAB III
METODOLOGI DAN PERANCANGAN

Bab ini membahas tentang metodologi dan perancangan perangkat lunak yang digunakan untuk penelitian ini. Langkah-langkah yang dilakukan dalam penelitian ini sebagai berikut :

a. Mempelajari dan memahami beberapa jurnal dan literatur mengenai data mining, algoritma *decision tree* C4.5, klasifikasi dengan *decision tree* dan *pruning* dalam *decision tree*
b. Menetukan permasalahan yang akan dibahas dalam klasifikasi data yaitu bagaimana mengimplementasikan algoritma *decision tree* C4.5 untuk menentukan klasifikasi keluarga peserta jamkesmas berdasarkan kemiskinan. Studi kasus dilakukan pada program jamkesmas Kabupaten Blitar
c. Menetukan kombinasi atribut yang dipakai dalam klasifikasi keluarga peserta jamkesmas berdasarkan kemiskinan
d. Menentukan metode pemangkasan pohon keputusan yaitu menggunakan aturan *post-pruning* dengan metode *statistical bounds*
e. Merancang dan membangun perangkat lunak yang digunakan untuk mengimplementasikan algoritma *Decision Tree* C4.5 untuk menentukan klasifikasi keluarga peserta jamkesmas berdasarkan kemiskinan
f. Mengimplementasikan perangkat lunak yang dibangun untuk proses klasifikasi keluarga peserta jamkesmas berdasarkan kemiskinan
g. Mencari kombinasi atribut yang menghasilkan hasil terbaik
h. Mengevaluasi dan menganalisa hasil implementasi, yaitu kinerja proses klasifikasi. Pengukuran yang digunakan adalah *precision, recall* dan *accuracy.*

Langkah-langkah yang dilakukan dalam penelitian dapat ditunjukkan pada Gambar 3.1.

Gambar 3.1 Alur proses penelitian

3. 1 Pengambilan Data *Training*

Data yang digunakan untuk penelitian adalah data kependudukan di Kabupaten Blitar. Data diperoleh dari Pemerintah Daerah Kabupaten Blitar yang berasal dari lintas sektoral yaitu Dinas Kesehatan, Badan Pemberdayaan Perempuan dan Keluarga Berencana (BPPKB), Dinas Kependudukan dan Catatan Sipil serta Badan Pusat Statistik (BPS) Kabupaten Blitar. Data tersebut sebelumnya sudah digunakan untuk program jamkesmas pada tahun 2010 dimana data diperoleh dari hasil pemutakhiran data penduduk tahun 2007 oleh Dinas Kependudukan dan Catatan Sipil digabung dengan data yang dimiliki oleh BPS pada tahun 2008. Data tersebut telah dilakukan pengecekan di lapangan oleh petugas penyuluh BPPKB pada tahun 2008. Sedangkan pada pertengahan tahun 2010 telah dilaksanakan dua program pemerintah yaitu pemutakhiran data penduduk oleh Dinas Kependudukan dan Catatan Sipil dan Sensus Tahun 2010 oleh BPS, sehingga database kependudukan mengalami perubahan dan berakibat pula berubahnya data penduduk untuk program Jamkesmas. Dengan adanya perubahan data tersebut, Dinas Kesehatan selaku instansi yang menggunakan secara langsung data dari peserta jamkesmas ingin melakukan klasifikasi kembali terhadap data keluarga peserta Jamkesmas berdasarkan kemiskinan.

Data yang digunakan untuk traning dan testing adalah data penduduk Desa Candirejo Kecamatan Ponggok Kabupaten Blitar. Jumlah keluarga yang tercatat sebanyak 2592 keluarga. Bulan desember tahun 2010 Petugas Penyuluh BPPKB telah melakukan pengecekan langsung ke lapangan untuk data Desa Candirejo, terkait kebenaran data terutama status miskin atau tidak miskin sehingga dapat masuk kategori peserta Jamkesmas atau tidak.

3.2 Analisa Data

Data yang digunakan berupa data nyata yang diambil dari data keluarga penduduk Kabupaten Blitar untuk pertengahan tahun 2010 yang telah mengalami perubahan data melalui pemutakhiran penduduk tahun 2010 dan hasil Sensus Tahun 2010. Data terdiri dari 8 atribut dengan jumlah data 2592 data keluarga.

Untuk uji coba aplikasi akan digunakan lima buah tipe data berdasar jumlah banyaknya data yaitu 100 data, 500 data, 1000 data, 1500 data dan 2000 data dimana rincian lebih jelas sebagai berikut :

Table 3.1 Data penelitian

No	Total Data Training	Total Data Testing	Status Ya	Status Tidak
1	100	300	47	53
2	500	300	230	270
3	1000	300	462	538
4	1500	300	711	789
5	2000	300	981	1019

3.3 Analisis Perangkat Lunak

Pada Sub Bab ini akan dijelaskan mengenai kebutuhan-kebutuhan dari perangkat lunak yang akan dibangun (*system requierments*).

3.3.1 Deskripsi Sistem

Pada penelitian ini akan dibangun sebuah sistem yang memiliki fungsi dan peran sebagai berikut :

a. Menentukan klasifikasi keluarga peserta Jamkesmas berdasarkan kemiskinan di Kabupaten Blitar
b. Menggunakan Algoritma *decision tree* C4.5 untuk fungsi klasifikasi pada data mining
c. Pengukuran kinerja dengan mengunakan parameter *precision*, *recall* dan *accuracy*
d. Mengetahui kombinasi atribut terbaik untuk menentukan klasifikasi keluarga peserta Jamkesmas berdasarkan kemiskinan di Kabupaten Blitar

3.3.2 Batasan Sistem

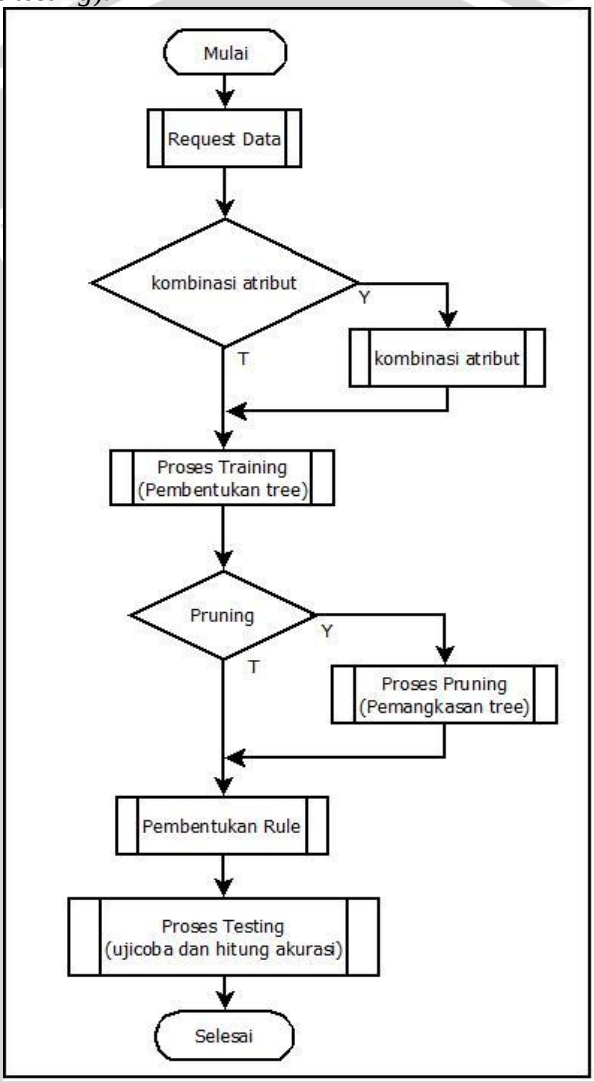
Adapun batasan pada sistem yang dibangun :

a. Data training dan testing harus memiliki nilai, tidak *missing value.*
b. Menggunakan aturan *post-pruning* dengan metode *statistical bounds*

3.4 Perancangan Perangkat Lunak

Pada sub bab ini akan dibahas mengenai berbagai proses yang terjadi pada implementasi pengklasifikasian data dengan algoritma *decision tree* C4.5 dan algoritma pemangkasan *post-pruning* dengan

metode *statistical bounds*. Proses-proses penting yang terjadi selama pengklasifikasian data, antara lain: proses *request* data, proses pembangunan model (proses *learning*), dan proses pengklasifikasian data (proses *testing*).

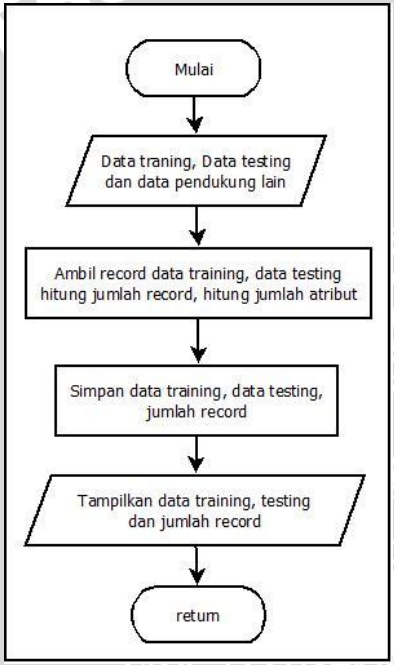


Gambar 3.2 Alur proses

3.4.1 Perancangan Proses *Request* Data

1. *Request* Data

Proses *request* data merupakan proses yang pertama dilakukan sistem, dimana sistem akan mendapatkan data untuk diolah, yaitu data *training* dan data *testing* berserta informasinya. Adapun informasi data *training* yang didapatkan yaitu berupa jumlah *record* tabel data *training.* Sedangkan informasi tabel data *testing* yang didapatkan yaitu nama tabel *testing* dan jumlah *record*.



Gambar 3.3 Proses *request* data

Pada saat aplikasi dijalankan, user akan memilih nama tabel data *training* dan tabel data *testing*. Kemudian sistem melakukan koneksi ke DBMS agar aplikasi dapat mengakses tabel yang disimpan dalam *database*. Lalu sistem menghitung jumlah *record* dari tabel data *training* dan tabel data *testing.* Kemudian sistem menyimpan informasi tabel data *training* dan informasi tabel data *testing* tersebut. Proses terakhir pada *request* data yaitu, sistem

menampilkan semua informasi data *training* dan data *testing* yang telah disimpan kepada *user*.

2. Pemilihan Variabel

Data penduduk Kabupaten Blitar diambil variabel keputusan yang digunakan sebagai variabel penentu dalam pembentukan pohon keputusan, variabel keputusan tersebut yaitu :

1. Pekerjaan Kepala Keluarga
2. Nilai Pajak Bumi dan Bangunan
3. Jenis Lantai Rumah
4. Jenis Dinding Rumah
5. Sumber Air Minum
6. Pendapatan
7. Jumlah Daya Listrik
8. Status Jamkesmas

Setelah dilakukan pemilihan variabel , selanjutnya dilakukan pra-proses yaitu

a. Menerjemahkan kategori pendapatan

Menerjemahkan kategori pendapatan keluarga tiap bulan sebagai berikut :

Tabel 3.2 Kategori pendapatan

Kategori Pendapatan	Pendapatan perbulan (Rp.)
Kategori A	Kurang dari 500.000
Kategori B	500.000 – 1.000.000
Kategori C	1.000.000 – 5.000.000
Kategori D	5.000.000 – 10.000.000
Kategori E	Lebih dari 10.000.0000

b. Menerjemahkan nilai pajak bumi dan bangunan

Menerjemahkan kategori besar pajak bumi dan bangunan yang harus dibayar setiap tahun sebagai berikut :

Tabel 3.3 Kategori PBB

Kategori PBB	PBB per Tahun (Rp.)
Kategori A	Kurang dari 25.000
Kategori B	25.000 – 50.000
Kategori C	50.000 – 100.000
Kategori D	100.000 – 500.000
Kategori E	Lebih dari 500.0000

c. Menerjemahkan Daya Listrik
Menerjemahkan kategori besar langganan daya listrik yang harus dibayar setiap tahun sebagai berikut :

Tabel 3.4 Kategori daya listrik

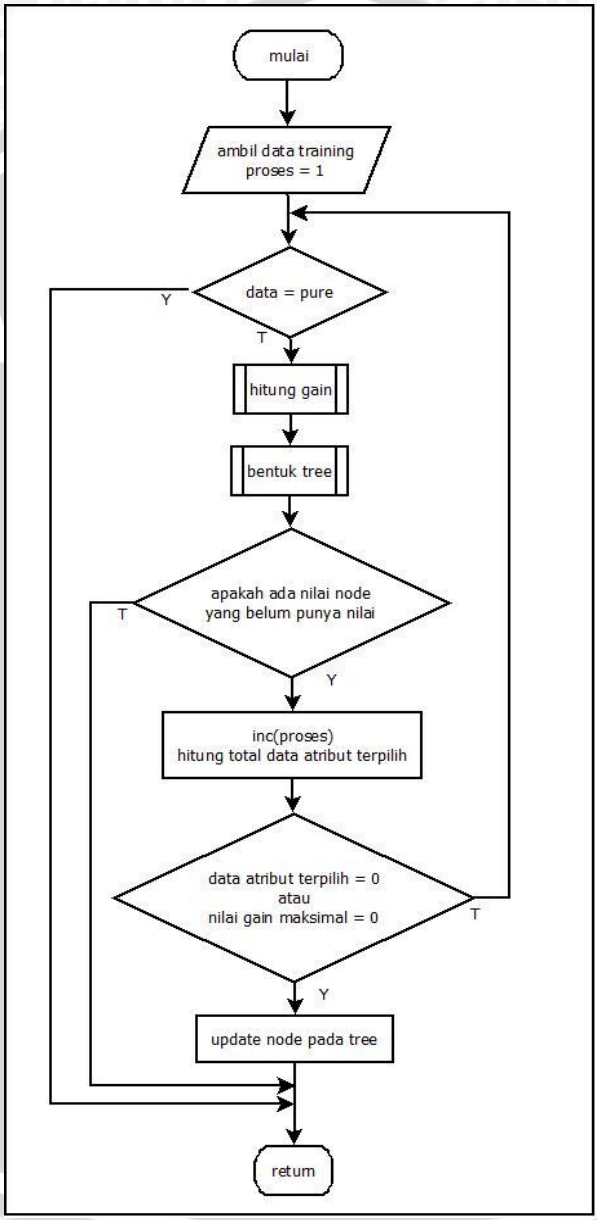
Kategori Listrik	Daya Listrik (VA)
Kategori A	450
Kategori B	900
Kategori C	1300
Kategori D	2200
Kategori E	Lebih dari 2200

3.4.2 Perancangan Proses *Learning*

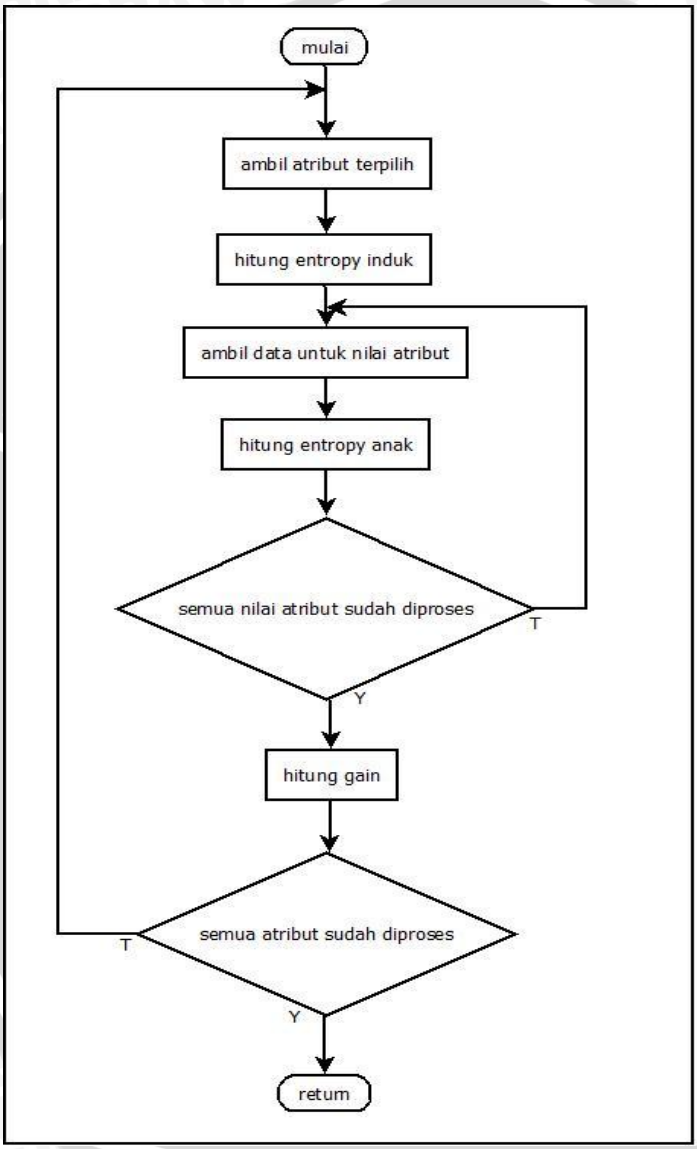
Proses *learning* merupakan implementasi dari proses pembuatan model klasifikasi pada pengklasifikasian data. Pengolahan data *training* dilakukan sehingga terbentuk sebuah model klasifikasi. Pada proses *learning* dibagi menjadi dua yaitu pembentukan *tree* dan perubahan *tree* menjadi *rule*. Selanjutnya dari hasil proses *learning* ini akan dilanjutkan dengan *pruning* atau tanpa *pruning* sehingga diperoleh *tree* yang lebih baik.

A. Pembentukan *Tree*

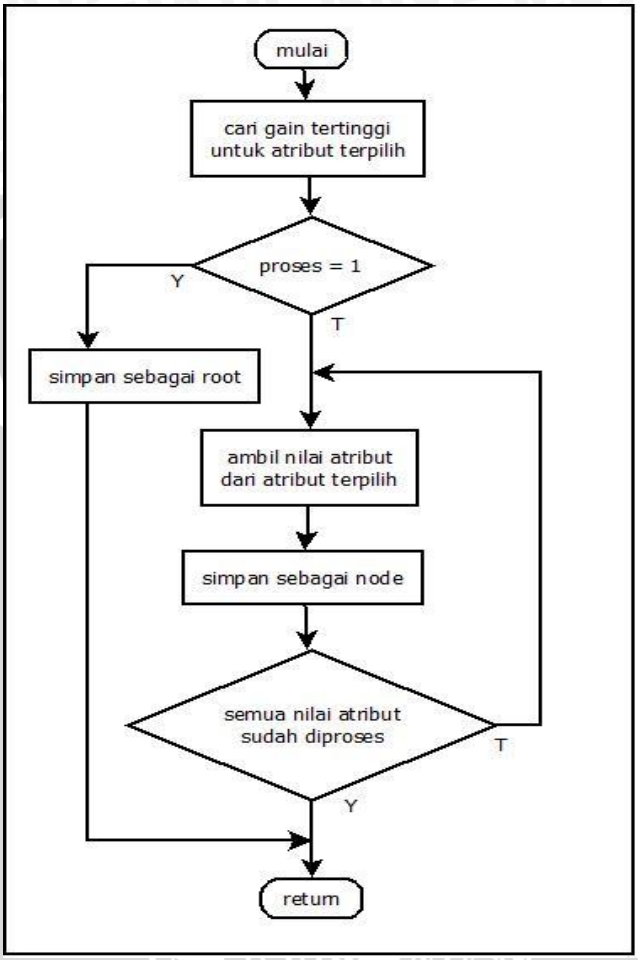
Proses pada pohon keputusan adalah mengubah bentuk data (tabel) menjadi model pohon, mengubah model pohon menjadi *rule*. Dalam kasus ini pada tabel akan dibuat pohon keputusan untuk menentukan status apakah sesorang merupakan peserta Jamkesmas atau tidak.



Gambar 3.4 Proses *Training*



Gambar 3.5 Proses Hitung Gain



Gambar 3.6 Proses Pembentukan *Tree*

Adapun proses-proses yang terjadi selama pada proses *training* adalah:

a. Sistem mengambil data *training*.
b. Sistem mengecek apakah setiap data sudah *pure* atau belum. *Pure* disini artinya kelas yang dimiliki oleh *node* sudah memiliki nilai yang sama
c. Jika data sudah *pure*, maka pada data tersebut tidak ada model *tree* yang terbentuk dan proses pembentukan *tree* berakhir.

Namun jika *node* belum *pure*, maka dilakukan pemanggilan prosedur hitung gain dan pembentukan *tree*

d. Dilakukan pengecekan apakah ada nilai *node* pada *tree* yang belum memiliki nilai
e. Jika data atribut terpilih = 0 atau nilai gain tertinggi = 0 maka dilakukan proses update nilai *node* tersebut dengan pertimbangan dari nilai *confidence*. Jika tidak maka lakukan proses perhitungan pada *node* yang belum memiliki nilai sebagai atribut terpilih.

Untuk proses yang terjadi pada proses hitung gain adalah :

a. Mengambil data atribut terpilih untuk tiap atribut
b. Dilakukan perhitungan entropi untuk induk
c. Mengambil data nilai atribut dari atribut terpilih
d. Menghitung entropi untuk setiap nilai atribut
e. Dilakukan pengecekan apakah semua data nilai atribut sudah diproses, jika sudah maka hitung nilai gain, namun jika tidak maka lakukan perhitungan untuk nilai atribut berikutnya
f. Dilakukan pengecekan apakah semua atribut sudah diproses, jika sudah maka proses selesai, namun jika tidak maka lakukan proses untuk atribut berikutnya.

Untuk proses yang terjadi pada proses pembentukan *tree* adalah:

a. Dilakukan pencarian untuk nilai gain tertinggi, atribut untuk gain tertinggi digunakan sebagai atribut terpilih
b. Jika proses = 1 maka simpan sebagai *root*
c. Jika proses < > 1 maka ambil nilai dari atribut terpilih dan simpan sebagai *node* (anak dari induk atribut terpilih)
d. Dilakukan pengecekan apakah semua nilai atribut sudah diproses, jika sudah semua maka proses selesai, namun jika tidak maka lanjutkan untuk proses pada nilai atribut berikutnya.

Adapun langkah-langkah perhitungan gain dapat dilihat pada contoh data berikut :

Tabel 3.5 Keputusan Status Peserta Jamkesmas

No	Pekerjaan	PBB	Lantai	Dinding	Sumber Air	Daya PLN	Penda-patan	Status
1	Swasta	B	keramik	tembok	PDAM	B	C	Tidak
2	Swasta	B	keramik	tembok	sumur	B	C	Tidak
3	Swasta	B	plester	kayu	sumur	A	A	Ya
4	Tani	A	plester	bata	sumur	A	A	Ya
5	Tani	A	plester	bata	sumur	A	B	Tidak
6	Tani	A	plester	tembok	sumur	A	B	Tidak
7	Tani	A	plester	tembok	sungai	A	B	Ya
8	Buruh	A	tanah	bata	sumur	A	B	Ya
9	Buruh	A	tanah	kayu	sungai	A	A	Ya
10	Buruh	A	tanah	kayu	sungai	A	A	Ya
11	Tani	A	tanah	kayu	sungai	A	A	Ya
12	Tani	A	tanah	kayu	sungai	A	A	Ya
13	Swasta	B	ubin	bata	sumur	A	B	Tidak
14	Swasta	B	ubin	tembok	PDAM	A	C	Tidak
15	Tani	B	ubin	bata	sumur	B	B	Tidak
16	Tani	A	ubin	tembok	sumur	B	B	Tidak
17	Tani	B	ubin	tembok	PDAM	A	C	Tidak
18	Tani	B	ubin	tembok	sumur	B	B	Tidak

Untuk memilih atribut sebagai *node*, didasarkan pada nilai gain tertinggi dari atribut-atribut yang ada. Maka dilakukan perhitungan gain dan entropi sesuai dengan rumus yang berlaku bagi algoritma *decision tree* C4.5

Tabel 3.6 Perhitungan *node* 1

Node			jml(S)	Tdk(s1)	Ya(s2)	Entropy	Gain
1	**Total**		18	10	8	0.99108	
	Pekerjaan						0.251
		Buruh	3	0	3	0	
		Tani	10	6	4	0.97095	
		Swasta	5	4	1	0.72193	
	PBB						0.379
		A	12	4	8	0.9183	
		B	6	6	0	0	
	lantai					0	**0.721**
		Tanah	5	0	5	0	
		Plester	5	2	3	0.97095	
		Ubin	6	6	0	0	
		Keramik	2	2	0	0	
	Air					0	0.413
		Sungai	4	0	4	0	
		Sumur	11	7	4	0.94566	
		Pdam	3	3	0	0	
	Daya PLN					0	0.297
		A	13	5	8	0.96124	
		B	5	5	0	0	
	Pendapatan						0.631
		A	6	0	6	0	
		B	8	6	2	0.81128	
		C	4	4	0	0	
	Dinding					0	0.48
		Kayu	5	0	5	0	
		Bata	5	3	2	0.97095	
		Tembok	8	7	1	0.54356	

Baris Total kolom entropi pada table di atas dihitung dengan persamaan (2.2) sebagai berikut :

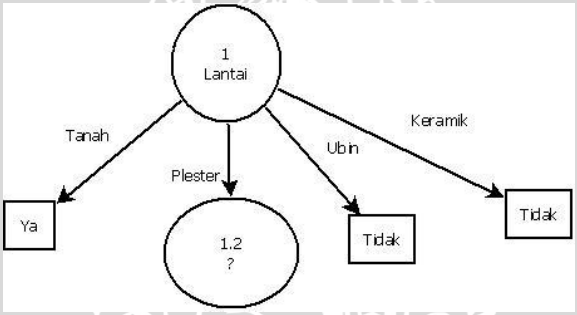
$$\textit{Entropy (Total)} = \left(-\frac{8}{18} * log_2\left(\frac{8}{18}\right)\right) + \left(-\frac{10}{18} * log_2\left(\frac{10}{18}\right)\right)$$

$$= 0{,}99108$$

Untuk nilai Gain pada baris Pekerjaan dihitung dengan menggunakan persamaan(2.1) sebagai berikut :

$$Gain(Total, Pekerjaan)$$
$$= 0{,}99108 - \left(\left(\frac{3}{18} * 0\right) + \left(\frac{10}{18} * 0{,}97\right) + \left(\frac{5}{18} * 9{,}72\right)\right)$$
$$= 0{,}251$$

Selanjutnya dilakukan perhitungan untuk entropi dan gain pada masing-masing atribut. Setelah perhitungan selesai dilakukan akan diketahui nilai gain yang terbesar yaitu pada atribut lantai dengan nilai 0,721 maka atribut lantai dapat dijadikan sebagai *root*. Pohon keputusan yang terbentuk dapat digambarkan sebagai berikut :



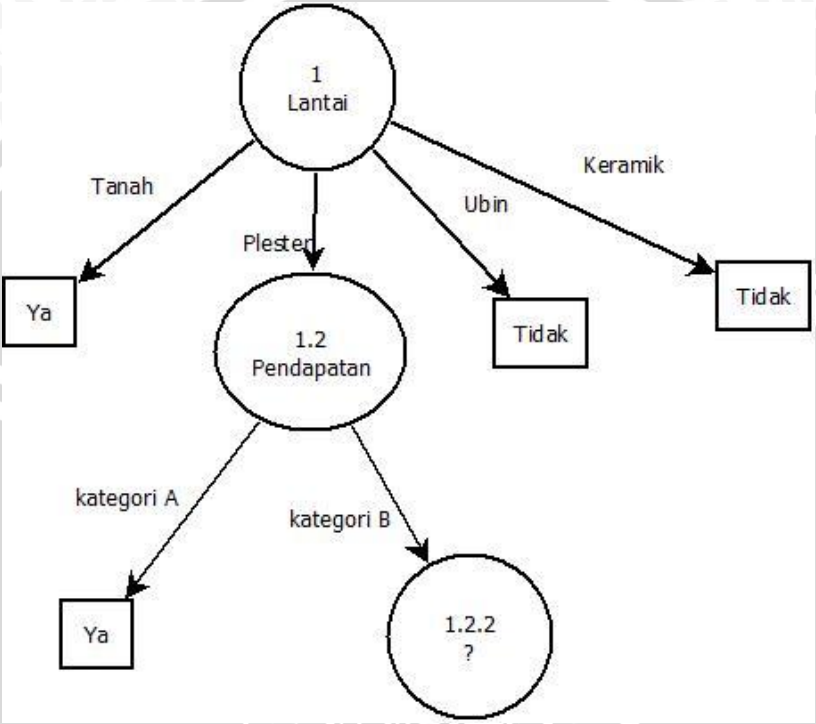
Gambar 3.7 Pohon keputusan *node* 1

Pada node 1.2 masih belum diketahui , sehingga untuk nilai atribut plester harus dilakukan perhitungan lagi

Tabel 3.7 Perhitungan *node* 1.2

Node			jml(S)	Tdk(s1)	Ya(s2)	Entropy	Gain
1.2	**Lantai-Plester**		5	2	3	0.97095	
	Pekerjaan						0.17
		Buruh	0	0	0	0	
		Tani	4	2	2	1	
		Swasta	1	0	1	0	
	PBB						0
		A	5	2	3	0.97095	
		B	0	0	0	0	
	Air					0	0
		Sungai	0	0	0	0	
		Sumur	5	2	3	0.97095	
		Pdam	0	0	0	0	
	Daya PLN					0	0
		A	5	2	3	0.97095	
		B	0	0	0	0	
	Pendapatan						**0.42**
		A	2	0	2	0	
		B	3	2	1	0.9183	
		C	0	0	0	0	
	Dinding						0.17
		Kayu	1	0	1	0	
		Bata	2	1	1	1	
		Tembok	2	1	1	1	

Diperoleh nilai gain yang terbesar pada atribut pendapatan sehingga *node* 1.2 ditempati oleh atribut pendapatan. Adapun pohon keputusan yang terbentuk sebagai berikut :



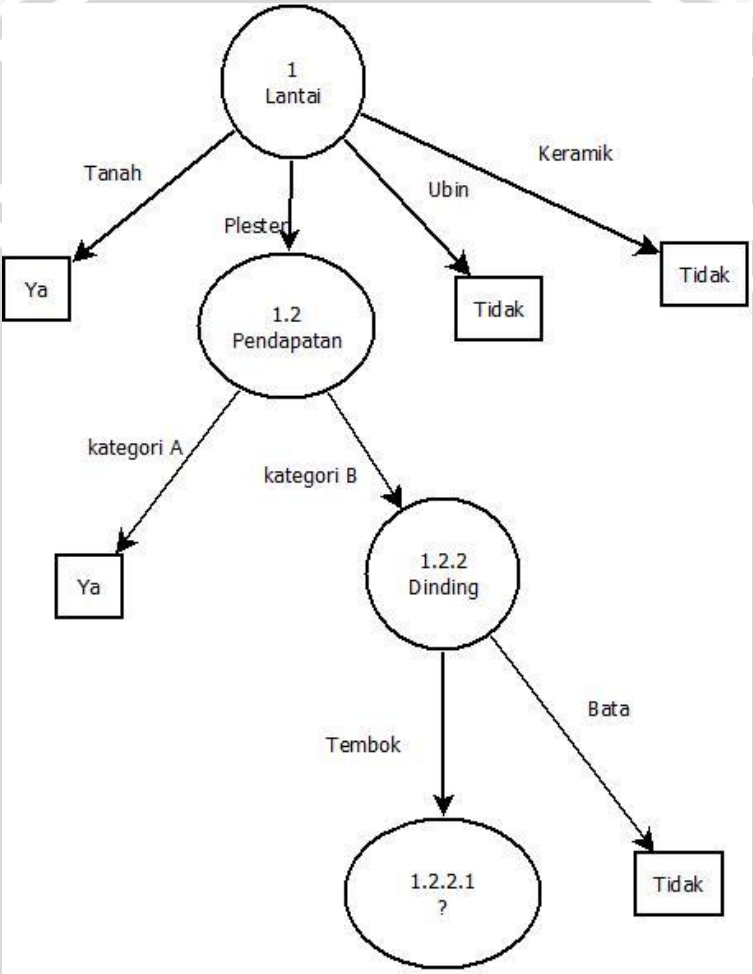
Gambar 3.8 Pohon keputusan *node* 1.2

Untuk node 1.2.2 masih belum diketahui maka harus dilakukan perhitungan lagi seperti sebelumnya.

Tabel 3.8 Perhitungan Node 1.2.2

Node			jml(S)	Tdk(s1)	Ya(s2)	Entropy	Gain
1.2.2	Lantai-Plester & Pendapatan-B		3	2	1	0.9183	
	Pekerjaan						0
		Buruh	0	0	0	0	
		Tani	3	2	1	0.9183	
		Swasta	0	0	0	0	
	PBB						0
		A	3	2	1	0.9183	
		B	0	0	0	0	
	Air					0	0
		Sungai	0	0	0	0	
		Sumur	3	2	1	0.9183	
		Pdam	0	0	0	0	
	Daya PLN					0	0
		A	3	2	1	0.9183	
		B	0	0	0	0	
	Dinding						**0.25**
		Kayu	0	0	0	0	
		Bata	1	1	0	0	
		Tembok	2	1	1	1	

Dari hasil perhitungan diperoleh nilai gain terbesar untuk atribut dinding senilai 0,25 maka *node* 1.2.2 ditempati oleh atribut dinding. Sedangkan pohon keputusan yang terbentuk dapat digambarkan sebagai berikut :



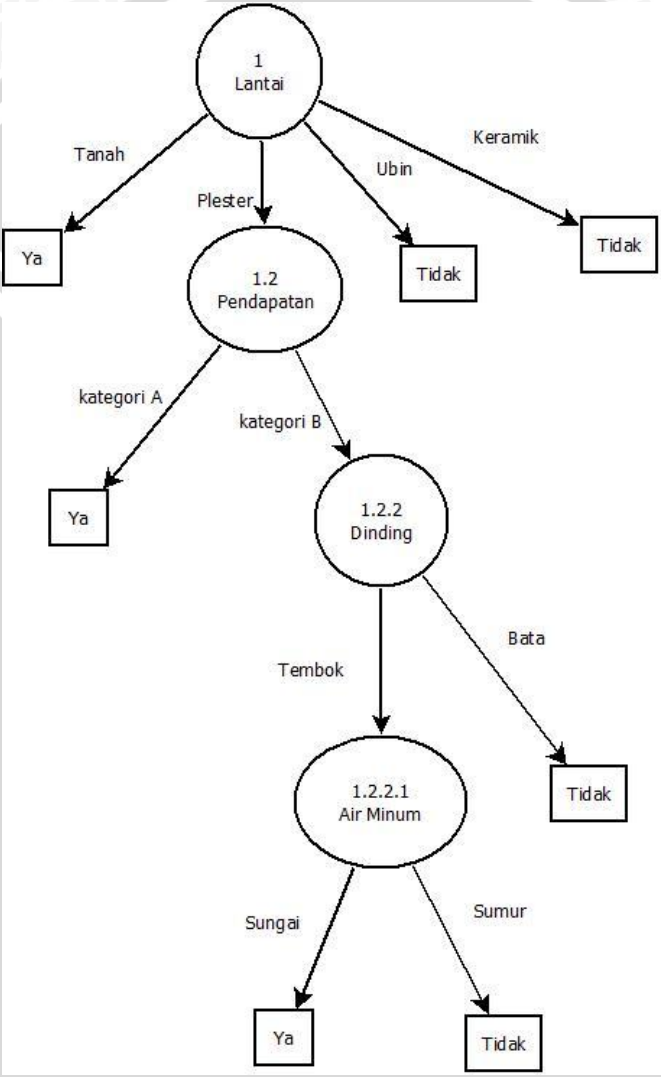
Gambar 3.9 Pohon keputusan *node* 1.2.2

Untuk *node* 1.2.2.1 masih belum diketahui sehingga perlu dilakukan perhitungan seperti sebelumnya dan nilai gain yang terbesar nantinya dipilih untuk menempati posisi *node* tersebut.

Tabel 3.9 Perhitungan *node* 1.2.2.1

Node			jml(S)	Tdk(s1)	Ya(s2)	Entropy	Gain
1.2.2.1	**Lantai-Plester & Pendapatan-B & Dinding-tembok**		2	1	1	1	
	Pekerjaan						
		Buruh	0	0	0	0	
		Tani	2	1	1	1	
		Swasta	0	0	0	0	
	PBB						0
		A	2	1	1	1	
		B	0	0	0	0	
	Air						**1**
		Sungai	1	0	1	0	
		Sumur	1	1	0	0	
		PDAM	0	0	0	0	
	Daya PLN						0
		A	2	1	1	1	
		B	0	0	0	0	

Dari hasil perhitungan yang didapat nilai gain terbesar pada atribut air minum sehingga dipilih untuk menempati *node* 1.2.2.1. Bentuk pohon keputusan yang terbentuk sebagai berikut :



Gambar 3.10 Pohon keputusan

B. Perubahan *tree* menjadi *rule* (*decision tree*)

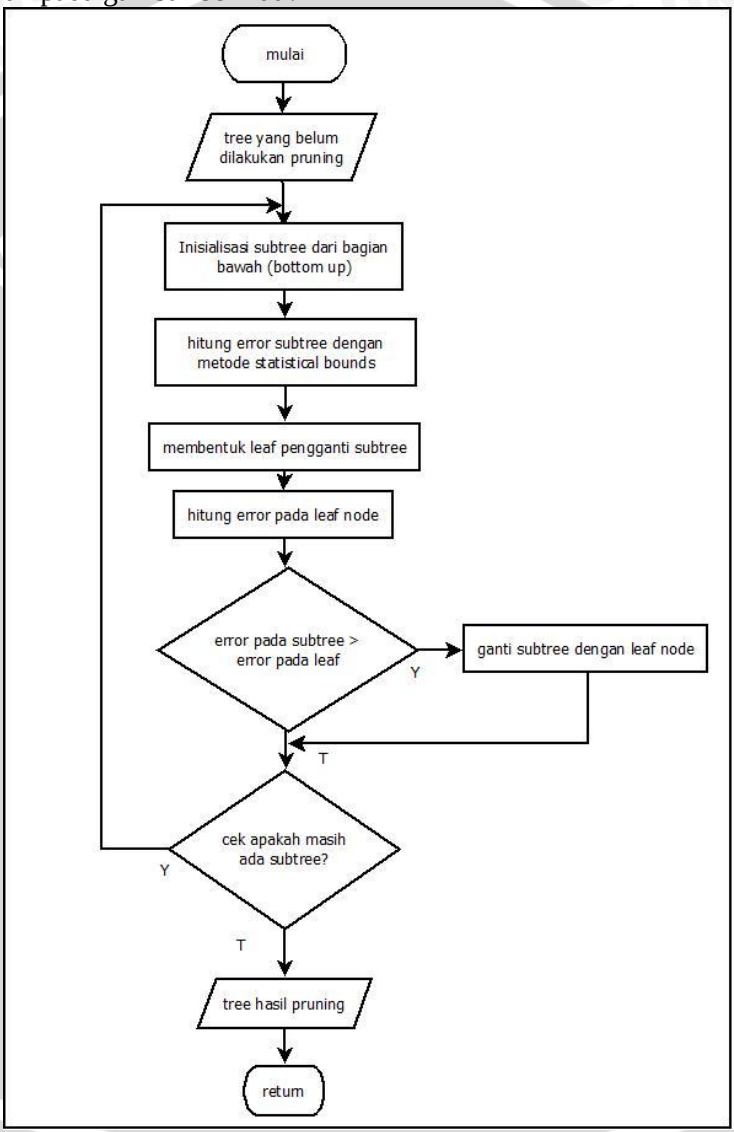
Setelah *tree* terbentuk maka selanjutnya dilakukan perubahan menjadi *rule*. Berikut ini merupakan *rule* sebelum di-*pruning* :

IF lantai = "tanah" **then** status_jamkesmas = Ya
ELSE IF lantai = "ubin" **then** status_jamkesmas = Tidak
ELSE IF lantai = "keramik" **then** status_jamkesmas = Tidak
ELSE IF lantai = "plester" **and** pendapatan = kategori A **then** status_jamkesmas = Ya
ELSE IF lantai = "plester" **and** pendapatan = kategori B **and** dinding="bata" **then** status_jamkesmas = Tidak
ELSE IF lantai = "plester" **and** pendapatan = kategori B and dinding="Tembok" and air_minum= "sungai" **then** status_jamkesmas = Ya
ELSE IF lantai = "plester" **and** pendapatan = kategori B **and** dinding="Tembok" **and** air_minum= "sumur" **then** status_jamkesmas = Tidak

Gambar 3.11 *Rule* sebelum dilakukan *pruning*

3.4.3 Perancangan Proses Pruning

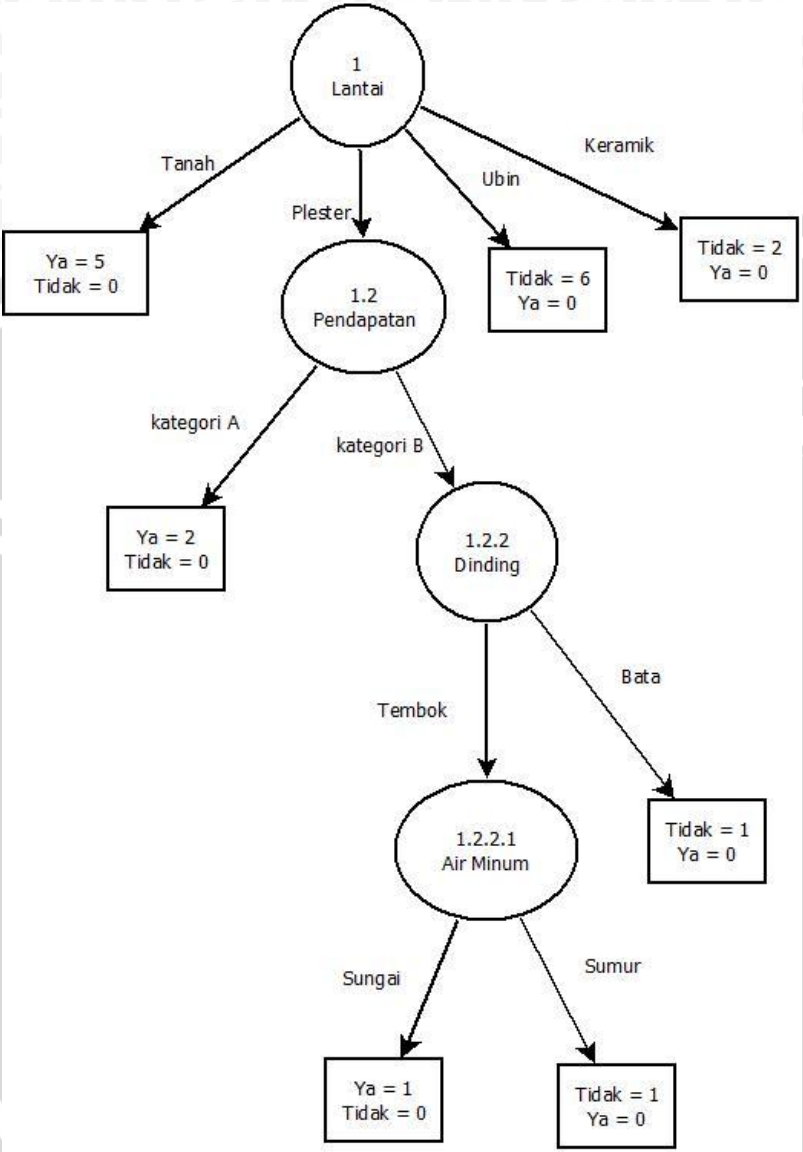
Proses yang terjadi dalam pruning metode *statistical bounds* dapat ditunjukan pada gambar berikut :



Gambar 3.12 Pruning dengan metode *Statistical Bound*

Tahapan proses pemangkasan metode *statistical bounds* adalah sebagai berikut :

a. Melakukan proses pengecekan pada *tree* dengan menginisialisasi setiap *subtree* dimulai dari paling bawah (*bottom up*)
b. Melakukan perhitungan *error* pada *subtree* dengan rumus *statistical bounds* (e_{Upper}). Untuk mengetahui besar *error* yang dihasilkan oleh sebuah *subtree* sebelum dipangkas
c. Membentuk sebuah *node* baru yaitu *leaf node*, yang mana merepresentasikan *subtree* dengan tujuan sebagai pengganti *subtree* apabila dalam perhitungan *error*nya didapatkan *error* pada *subtree* (kondisi sebelum dipangkas) lebih besar daripada *error* pada *node* baru atau *leaf node* (kondisi sesudah pemangkasan)
d. Melakukan proses perhitungan *error* pada *node baru* atau *leaf node*. Selanjutnya akan didapatkan nilai *error* pada *subtree* dan *leaf node*
e. Melakukan pengecekan nilai *error* pada *subtree* dan *leaf node*. Jika penghitungan *error* pada *leaf node* (kondisi sesudah pemangkasan) menghasilkan *error* yang lebih kecil dari *error* yang dihasilkan *subtree* (kondisi sebelum dipangkas), maka *subtree* diganti dengan *node leaf* atau *subtree* dipangkas. Jika sebaliknya maka tidak terjadi pemangkasan
f. Mengecek apakah masih ada percabangan yang masih bisa diproses atau tidak. Proses pemangkasan akan berhenti sampai tidak ditemukan lagi percabangan atau *node* yang ditemui adalah *root*.



Gambar 3.13 *Tree* sebelum dilakukan *pruning*

Untuk perhitungan *pruning* dengan *statistical bounds* dapat dilakukan sebagaimana berikut (dimulai dari percabangan paling bawah, lihat gambar 3.13) :

1. Sub*tree* Air Minum

a. Untuk *subtree* Air Minum, pada kondisi sebelum dipangkas perhitungannya adalah :

➢ *leaf* Air Minum sebelah kiri (Air Minum = Sungai):

diketahui :

N = 1

e(t) = 0

e = 0/1 = 0

dan z = 0,69

maka berdasarkan persamaan (2.4):

$$e_{Upper} = \frac{0 + \frac{0.69^2}{2*1} + 0.69\sqrt{\frac{0(1-0)}{1} + \frac{0.69^2}{4*1^2}}}{1 + \frac{0.69^2}{1}}$$

= 0.322539

➢ *leaf* Air Minum sebelah kanan (Air Minum = Sumur):

diketahui :

N = 1

e(t) = 0

e = 0/1 = 0

dan z = 0,69

maka berdasarkan persamaan (2.4):

$$e_{Upper} = \frac{0 + \frac{0.69^2}{2*1} + 0.69\sqrt{\frac{0(1-0)}{1} + \frac{0.69^2}{4*1^2}}}{1 + \frac{0.69^2}{1}}$$

= 0,322539

Sehingga *error* keseluruhan pada kedua *leaf* Air Minum adalah

(1 x 0,322539)+(1 x 0,322539) = **0,645078 *error***

b. Untuk *subtree* Air Minum, pada kondisi setelah dipangkas perhitungannya adalah :

subtree Air Minum, diketahui :

N = 2

e(t) = 1
e = 1/2 = 0,5
dan z = 0,69
maka berdasarkan persamaan (2.4):

$$e_{Upper} = \frac{0{,}5 + \frac{0.69^2}{2*2} + 0.69\sqrt{\frac{0{,}5(1-0{,}5)}{20} + \frac{0.69^2}{4*2^2}}}{1 + \frac{0.69^2}{2}}$$

= 0,719248

Sehingga *error* keseluruhan pada *subtree* Air Minum adalah

$N \times e_{Upper} = 2$ x 0,719248 = **1,438495 *error*.**

Berdasarkan perhitungan *error* pada *subtree* Air Minum dan *error* pada *leaf* Air Minum maka *subtree* Air Minum tidak mengalami pemangkasan, karena *error* pada *subtree* Air Minum lebih besar dari *leaf* "Air Minum"

2. Sub*tree* dinding

a. Untuk *subtree* dinding, pada kondisi sebelum dipangkas perhitungannya adalah :

- *leaf* dinding (dinding= bata) :

diketahui :
N = 1
e(t) = 0
e = 0/1 = 0
dan z = 0,69
maka berdasarkan persamaan (2.4):

$$e_{Upper} = \frac{0 + \frac{0.69^2}{2*1} + 0.69\sqrt{\frac{0(1-0)}{1} + \frac{0.69^2}{4*1^2}}}{1 + \frac{0.69^2}{1}}$$

= 0.322539

- untuk *leaf* dinding (dinding = tembok), telah didapat pada perhitungan sebelumnya yaitu (1 x 0,322539) + (1 x 0,322539) = **0,645078 *error***

Sehingga *error* keseluruhan pada *leaf* dinding adalah (1 x 0,322539) + (1 x 0,322539) + (1 x 0,322539) = **0,96761737 *error*.**

b. Untuk *subtree* dinding, pada kondisi setelah dipangkas perhitungannya adalah :

Subtree dinding, diketahui :

N = 3

e(t) = 1

e = 1/3 = 0,333

dan z = 0,69

maka berdasarkan persamaan (2.4):

$$e_{Upper} = \frac{0{,}33 + \frac{0{,}69}{2*3} + 0{,}69\sqrt{\frac{0{,}33(1-0{,}33)}{3} + \frac{0{,}69^2}{4*3^2}}}{1 + \frac{0{,}69^2}{3}}$$

= 0,83992429

Sehingga *error* keseluruhan pada *subtree* dinding adalah

$N \times e_{Upper}$ = 3 x 0,83992429 = **2,519772871 *error***.

karena *error* pada *subtree* dinding lebih besar dari *leaf* dinding, maka *subtree* dinding tidak mengalami pemangkasan.

3. Sub*tree* pendapatan

a. Untuk *subtree* pendapatan, pada kondisi sebelum dipangkas perhitungannya adalah :

- *leaf* pendapatan (pendapatan = kategori A) :

diketahui :

N = 2

e(t) = 0

e = 0/2 = 0

dan z = 0,69

maka berdasarkan persamaan (2.4):

$$e_{Upper} = \frac{0 + \frac{0.69^2}{2*2} + 0.69\sqrt{\frac{0(1-0)}{2} + \frac{0.69^2}{4*2^2}}}{1 + \frac{0.69^2}{2}}$$

= 0,749606236

- untuk *leaf* pendapatan (pendapatan = kategori B), telah didapat pada perhitungan sebelumnya yaitu (1 x 0,322539) + (1 x 0,322539) + (1 x 0,322539) = **0,96761737 *error*.**

Sehingga *error* keseluruhan pada *leaf* pendapatan adalah (1 x 0,322539) + (1 x 0,322539) + (1 x 0,322539) + (2 x 0,749606236) = **2,466829841 *error*.**

b. Untuk *subtree* pendapatan, pada kondisi setelah dipangkas perhitungannya adalah :
   Subtree pendapatan, diketahui :
   N = 5
   e(t) = 2
   e = 2/5 = 0,4
   dan z = 0,69
   maka berdasarkan persamaan (2.4):

$$e_{Upper} = \frac{0{,}4 + \frac{0{,}69}{2*5} + 0{,}69\sqrt{\frac{0{,}4(1-0{,}4)}{5} + \frac{0{,}69^2}{4*5^2}}}{1 + \frac{0{,}69^2}{5}}$$

= 0,818193023

Sehingga *error* keseluruhan pada *subtree* pendapatan adalah

$N \times e_{Upper} = 5$ x 0,818193023 = **4,090965116 *error***.

karena *error* pada *subtree* pendapatan lebih besar dari *leaf* pendapatan, maka *subtree* pendapatan tidak mengalami pemangkasan.

Setelah dilakukan *pruning* ternyata *tree* tidak mengalamai perubahan atau pemangkasan sehingga *tree* dan *rule* tetap sama seperti sebelum dilakukan *pruning*

3.4.4 Perancangan Proses Pengujian Model Klasifikasi

Pada tahap ini dilakukan pengujian pada model klasifikasi yang didapat dengan data *testing*, sehingga diperoleh nilai untuk pengukuran kinerja klasifikasi. Data *testing* yang digunakan untuk perhitungan adalah Tabel data *testing* seperti pada Tabel 3.10.

Tabel 3.10 Data *testing*

No	Kerja	PBB	Lantai	Dinding	Sumber Air	Daya PLN	Pendapatan	Status
1	tani	B	ubin	kayu	sungai	A	A	tdk
2	tani	A	plester	bata	sumur	A	A	ya
3	buruh	A	tanah	kayu	sumur	A	B	ya
4	swasta	B	ubin	tembok	sungai	A	A	tdk
5	tani	B	keramik	bata	sumur	A	B	tdk
6	swasta	B	keramik	tembok	sumur	A	A	tdk
7	buruh	B	ubin	tembok	PDAM	B	C	tdk
8	buruh	B	keramik	tembok	sumur	B	A	tdk
9	buruh	A	tanah	kayu	sumur	A	B	ya
10	petani	A	plester	bata	sungai	A	B	ya
11	petani	A	ubin	kayu	sumur	A	A	tdk
12	petani	B	plester	bata	sumur	A	A	ya
13	petani	B	plester	bata	sumur	A	A	ya
14	petani	A	ubin	bata	sumur	A	B	tdk
15	petani	B	keramik	tembok	PDAM	B	B	tdk
16	petani	B	plester	tembok	sumur	A	A	tdk
17	swasta	B	ubin	tembok	PDAM	B	B	tdk
18	swasta	B	ubin	tembok	sumur	A	A	tdk
19	tani	A	tanah	bata	sungai	A	A	tdk
20	buruh	A	plester	tembok	sumur	A	B	ya

Rule model klasifikasi untuk uji coba menggunakan *rule* yang dihasilkan dari proses *pruning*.

IF lantai = "tanah" **then** status_jamkesmas = Ya
ELSE IF lantai = "ubin" **then** status_jamkesmas = Tidak
ELSE IF lantai = "keramik" **then** status_jamkesmas = Tidak
ELSE IF lantai = "plester" **and** pendapatan = kategori A **then** status_jamkesmas = Ya
ELSE IF lantai = "plester" **and** pendapatan = kategori B **and** dinding="bata" **then** status_jamkesmas = Tidak
ELSE IF lantai = "plester" **and** pendapatan = kategori B and dinding="Tembok" and air_minum= "sungai" **then** status_jamkesmas = Ya
ELSE IF lantai = "plester" **and** pendapatan = kategori B **and** dinding="Tembok" **and** air_minum= "sumur" **then** status_jamkesmas = Tidak

Gambar 3.14 *Rule* hasil proses *pruning*

Berikut ini merupakan hasil dan perbandingan dengan data sebelum dilakukan proses uji coba :

Tabel 3.11 Hasil uji coba

No	Sebelum ujicoba	Setelah ujicoba	Hasil
1	tidak	tidak	cocok
2	ya	ya	cocok
3	ya	ya	cocok
4	tidak	tidak	cocok
5	tidak	tidak	cocok
6	tidak	tidak	cocok
7	tidak	tidak	cocok
8	tidak	tidak	cocok
9	ya	ya	cocok
10	**ya**	**tidak**	**tidak**
11	tidak	tidak	cocok
12	ya	ya	cocok
13	ya	ya	cocok
14	tidak	tidak	cocok

15	tidak	tidak	cocok
16	**tidak**	**ya**	**tidak**
17	tidak	tidak	cocok
18	tidak	tidak	cocok
19	**tidak**	**ya**	**tidak**
20	**ya**	**tidak**	**tidak**

Dari hasil tersebut dapat diperoleh informasi sebagai berikut:

Tabel 3.12 Pengukuran kinerja

	Diidentifikasi sebagai miskin	Diidentifikasi sebagai tidak miskin
Miskin	5	2
Tidak Miskin	2	11

Precision = (a / (a + b)) x 100 %
= (5 / (5 + 2)) x 100 %
= (5/7) x 100%
= 71,43 %

Recall = (a / (a + c)) x 100 %
= (5 / (5 + 2)) x 100 %
= (5/7) x 100%
= 71,43 %

Accuracy = ((a + d) / total) * 100 %
= ((5+11) / 20) x 100 %
=(16/20)x100%
= 80%

3.4.5 Perancangan Proses *Testing*

Pada penelitian ini, proses *testing* dilakukan pada *rule* yang telah mengalami proses *pruning* dan tanpa *pruning*. Proses *testing* dilakukan dengan cara melakukan klasifikasi data *testing* terhadap *rule* yang dihasilkan. Menghitung akurasi yang dihasilkan oleh *rule,* Jumlah salah dan jumlah benar dalam mengklasifikasikan *rule,* dan *error* yang dihasilkan oleh *rule*. Proses *testing rul*e yang telah

mengalami *pruning* maupun tidak mengalami pruning dilakukan terhadap data *testing*, dengan tujuan untuk mendapatkan akurasi dari *rule*. Proses *testing* tersebut menggunakan data yang digunakan khusus untuk *testing*.

3.5 Perancangan Basis Data

Tabel Basis Data yang digunakan untuk menjalankan aplikasi ini terdapat beberapa tabel, antara lain :

- Tabel atribut

  Tabel atribut berisikan daftar variabel yang digunakan oleh sistem. Berikut *field* untuk tabel atribut :

Tabel 3.13 Tabel atribut

No	Field	Tipe Data	Keterangan
1	atribut	varchar(50)	Nama atribut
2	aktif	char(1)	Atribut sedang dipakai atau tidak
3	tujuan	char(1)	Atribut merupakan tujuan
4	ket	varchar(30)	Keterangan tentang atribut tersebut

Field atribut berisi nama atribut yang digunakan aplikasi. *Field* aktif berisi 'Y' atau 'T' merupakan pernyataan apakah atribut digunakan atau tidak. *Field* hasil berisi 'Y' atau 'T' merupakan pernyataan apakah atribut sebagai tujuan atau bukan.

- Tabel *training*

  Tabel *training* berisikan nilai-nilai atribut yang ada pada tabel atribut yang digunakan untuk proses *training*, berikut *field* dari tabel *training*:

Tabel 3.14 Tabel *training*

No	Field	Tipe Data	Keterangan
1	no_urut	int(11)	No urut
2	pekerjaan_kk	varchar(30)	Pekerjaan dari kepala keluarga

3	nilai_pbb	varchar(1)	Nilai PBB yang dibayar tiap tahun
4	jenis_lantai	varchar(30)	Jenis lantai rumah
5	jenis_dinding	varchar(30)	Jenis dinding rumah
6	sumber_air	varchar(30)	Sumber air bersih
7	pendapatan	char(1)	Jumlah pendapatan tiap bulan
8	daya_listrik	char(1)	Besar langganan daya listrik
9	status_jamkesmas	char(1)	Status kategori ya atau tidak

- Tabel *testing*

Tabel *testing* berisikan data yang akan digunakan untuk *testing*, berikut *field* dari tabel *testing* :

Tabel 3.15 Tabel *testing*

No	Field	Tipe Data	Keterangan
1	no_urut	int(11)	No Urut
2	pekerjaan_kk	varchar(30)	Pekerjaan dari kepala keluarga
3	nilai_pbb	varchar(1)	Nilai PBB yang dibayar tiap tahun
4	jenis_lantai	varchar(30)	Jenis lantai rumah
5	jenis_dinding	varchar(30)	Jenis dinding
6	sumber_air	varchar(30)	Sumber air bersih
7	pendapatan	char(1)	Jumlah pendapatan tiap bulan
8	daya_listrik	char(1)	Besar langganan daya listrik
9	status_jamkesmas	char(1)	Status kategori ya atau tidak
10	status_prediksi	char(1)	Status hasil proses testing
11	confidence	float	Nilai tingkat kepercayaan

- Tabel *tree*

Tabel *tree* merupakan tabel yang berisi struktur *tree* yang terbentuk. Data *field* untuk tabel *tree* sebagai berikut :

Tabel 3.16 Tabel *tree*

No	Field	Tipe Data	Keterangan
1	id_node	int(11)	No urut *node*
2	Node	varchar(50)	Nama *node*
3	Nilai	varchar(20)	Nilai atribut
4	Induk	int(11)	*Node* induk
5	internal_node	char(1)	Merupakan internal *node* atau *leaf*
6	Level	int(11)	Tingkat level proses
7	s1	int(11)	Jumlah data yang terklasifikasi
8	s2	int(11)	Jumlah data yang tidak terklasifikasi
9	Confidence	float	Nilai tingkat kepercayaan
10	Ket	text	Atribut yang terpilih
11	History	text	Atribut yang sudah dilakukan proses perhitungan
12	cek_prune	varchar(1)	Apakah sudah dilakukan pruning
13	error_subtree	double	Nilai *error subtree*
14	error_leaf	double	Nilai *error leaf*

Field id_*node* berisi nomor urut *node*. *Field node* merupakan atribut yang terpilih sebagai *node*. *Field* nilai berisi nilai atribut induk dari *node* pada *record* tersebut. *Field* internal_node merupakan pernyataan apakah *record* tersebut merupakan *leaf* (ujung dari *tree*) atau node biasa (*internal node*).

- Tabel Gain

Tabel gain berisi data hasil perhitungan nilai gain. Berikut *field* dari tabel gain :

Tabel 3.17 Tabel Gain

No	Field	Tipe Data	Keterangan
1	Proses	int(11)	Nomer proses
2	Node	varchar(255)	Nomer *node*
3	Atribut	varchar(50)	Nama atribut
4	Gain	double	Nilai gain

- Tabel *entropy*

  Tabel *entropy* berisi data hasil perhitungan nilai *entropy*. Berikut field dari tabel *entropy* :

Tabel 3.18 Tabel *Entropy*

No	Field	Tipe Data	Keterangan
1	Proses	int(11)	Nomer proses
2	Node	int(11)	Nomer node
3	Atribut	varchar(50)	Nama atribut
4	Nilai	varchar(255)	Nilai dari atribut
5	Entropy	double	Nilai entropy
6	s1	int(11)	Jumlah data 'Ya'
7	s2	int(11)	Jumlah data 'Tidak'
8	S	int(11)	Total data

- Tabel Hasil

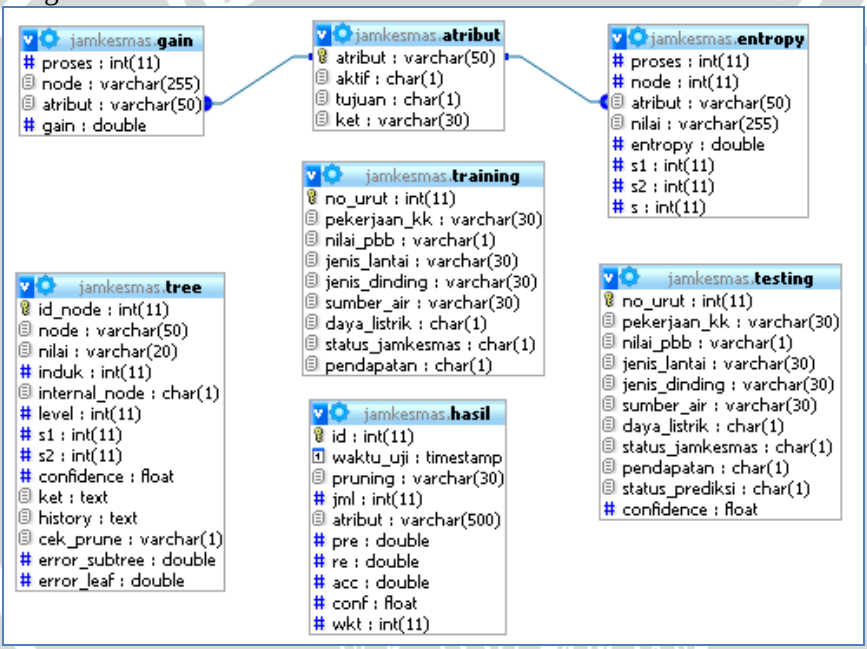
  Tabel hasil berisi data hasil kinerja untuk pengujian dari proses *testing*. Berikut *field* dari tabel hasil :

Tabel 3.19 Tabel Hasil

No	Field	Tipe Data	Keterangan
1	Id	int(11)	No urut
2	waktu_uji	timestamp	Tanggal dan jam pengujian
3	Pruning	varchar(30)	Apakah dengan *pruning* atau tidak
4	Jml	int(11)	Jumlah data training

5	atribut	varchar(500)	Kombinasi atribut yang dipakai
6	Pre	double	Nilai precision
7	Re	double	Nilai recall
8	Acc	double	Nilai accuracy
9	conf	float	Nilai confidence
10	wkt	int(11)	Waktu eksekusi testing

Untuk hubungan antar tabel dapat dilihat pada Relasi Entitas Diagram berikut :



Gambar 3.15 Relasi tabel

Pada gambar 3.15 relasi tabel terlihat bahwa tabel atribut merupakan tabel induk dari tabel yang lain yaitu tabel gain dan entropi. Sedangkan untuk tabel yang lain tidak memiliki hubungan antar tabel atau berdiri sendiri.

3.6 Rancangan ujicoba

Pengukuran kinerja dapat dilihat dari *precision* , *recall* dan *accuracy* serta waktu eksekusi aplikasi yaitu untuk waktu eksekusi total antara proses *training* dan *testing*. Pengukuran kinerja berdasarkan jumlah data yang digunakan untuk proses training. Tabel yang digunakan untuk pengambilan data tersebut ditunjukkan pada Tabel 3.20

Tabel 3.20 Pengambilan data hasil pengujian C4.5 tanpa *pruning* dan dengan *pruning*

Jumlah Data	*Precision*	*Recall*	*Accuracy*	*Confidence*	Waktu (ms)
100					
500					
1000					
1500					
2000					

Sedangkan untuk data hasil pengujian dengan menggunakan kombinasi baik untuk proses tanpa *pruning* dan dengan *pruning* ditunjukkan pada Tabel 3.21. Jumlah data *training* yang digunakan adalah 2000 data. Tabel dibuat bagi posisi terbaik untuk *precision*, *recall*, *accuracy* dan waktu eksekusi *testing* pada masing-masing jumlah data *training*. Sedangkan untuk data secara keseluruhan dapat dilihat pada halaman lampiran.

Tabel 3.21 Pengambilan data hasil pengujian C4.5 dengan kombinasi atribut tanpa *pruning* dan dengan *pruning*

Kombinasi	Atribut	*Kinerja yang dihitung*
1 atribut		
2 atribut		
3 atribut		
4 atribut		
5 atribut		
6 atribut		
7 atribut		

BAB IV
HASIL DAN PEMBAHASAN

4.1 Perangkat Sistem

Perangkat sistem terdiri dari perangkat lunak dan perangkat keras yang digunakan dalam penelitian.

4.1.1 Perangkat Lunak

Perangkat Lunak yang digunakan dalam penelitian ini adalah :

1. Sistem operasi Microsoft Windows XP Profesional *Version* 2007-2008 *Service Pack* 2
2. Borland Delphi 7 sebagai *software development* dalam pembuatan aplikasi
3. MySQL 5 sebagai DBMS *(Database Management System)*.

4.1.2 Perangkat Keras

Perangkat keras yang digunakan dalam penelitian ini adalah sebuah komputer *desktop* dengan spesifikasi sebagai berikut :

1. *Processor* Intel Core2 Duo T5300 @ 1,73 GHz
2. *Memory* 2 GB RAM
3. VGA NVIDIA GeForce Go 7300
4. *Space Hardisk* 120 GB.

4.2 Implementasi

Pada Sub Bab ini akan dijelaskan mengenai implementasi dari rancangan perangkat lunak yang dijelaskan sebelumnya pada Bab 3. Implementasi terdiri dari beberapa proses yaitu Proses *training*, Proses *Pruning*, Proses *Rule*, Proses *Testing*, dan Proses Kombinsi Atribut.

4.2.1 Proses *Training*

Pada proses *training* terdapat tiga buah prosedur yaitu prosedur *training* , prosedur perhitungan nilai gain dan prosedur pembentukan *tree*. Ditunjukkan pada *source code* 4.1 untuk prosedur *training*

```
1   procedure Tftraining.training(var datatran:integer);
2   var his,pilih,tujuan,isnode:string;
3   proses,id:integer;pure,kosong:boolean;
4   begin
5     pure:=false;
6     kosong:=false;
7     with fdb.Qproses do
8     begin
9       Close;
10      SQL.Text := 'delete from tree';
11      Execute;
12      Close;
13      SQL.Text := 'ALTER TABLE tree AUTO_INCREMENT =1';
14      Execute;
15      Close;
16      SQL.Text := 'delete from entropy';
17      Execute;
18      Close;
19      SQL.Text := 'delete from gain';
20      Execute;
21      Close;
22      SQL.Text := 'select atribut from atribut '+
23        ' where tujuan = ''Y'' and Aktif = ''Y''';
24      Open;
25      tujuan := Fields[0].AsString;
26     end;
27    pilih:='';
28    proses:=1;
29   while pure=false do
30    begin
31      if kosong=false then
32      begin
33        hitung_gain(pilih,tujuan,his,proses,datatran);
34        tree(proses,id,datatran,tujuan,pilih,his);
35      end;
36      with fdb.qpure do
37      begin
38          Close;
39          SQL.Text := 'select id_node,ket,history,s1,s2 from
40  tree where node=:node order by id_node';
41          ParamByName('node').AsString := 'belum ketemu';
42          Open;
43      end;
44      if fdb.qpure.RecordCount=0 then
45       begin
46        pure:=true;
47       end
48      else
49      begin
50        inc(proses);
51        id:=fdb.qpure.Fields[0].Asinteger;
52        pilih:=fdb.qpure.Fields[1].AsString;
53        his:=fdb.qpure.Fields[2].AsString;
```

```
54          with fdb.qproses2 do
55          begin
56            SQL.Text := 'SELECT count(*) as tot from atribut
57  where aktif=:y and tujuan=:t '+his;
58            ParamByName('t').Asstring := 'T';
59            ParamByName('y').Asstring := 'Y';
60                  Open;
61            if (fdb.qproses2.Fields[0].Asinteger = 0) or
62  (cgain=1) then
63              begin
64              kosong:=true;
65              if fdb.qpure.Fields[3].Asinteger >
66  fdb.qpure.Fields[4].Asinteger then
67              begin
68                  isnode:='Y';
69                  confidence:= fdb.qpure.Fields[3].Asinteger /
70  (fdb.qpure.Fields[3].Asinteger +
71  fdb.qpure.Fields[4].Asinteger);
72              end
73              else
74              begin
75                  isnode:='T';
76                  confidence:= fdb.qpure.Fields[4].Asinteger /
77  (fdb.qpure.Fields[3].Asinteger +
78  fdb.qpure.Fields[4].Asinteger);
79              end;
80                with fdb.qin do
81                begin
82                  Close;
83                  SQL.Text := 'update tree set node=:node,
84  internal_node=:t, confidence=:conf where id_node=:id';
85                  ParamByName('node').asstring := isnode;
86                  ParamByName('id').asinteger := id;
87                  ParamByName('conf').asfloat := confidence;
88                  ParamByName('t').asstring := 'T';
89                  Execute;
90                end;
91              end;
92          end;
93          pure:=false;
94      end;
95    end;
96  end;
```

Source code 4.1 Prosedur *training*

Pada prosedur *training* terdapat dua buah prosedur yang harus dipanggil yaitu prosedur hitung gain dan prosedur *tree*. Dimana prosedur hitung gain merupakan prosedur untuk menghitung nilai entropi dan gain, sedangkan prosedur *tree* digunakan untuk membentuk sebuah *decision tree*.

Proses yang dilakukan pada prosedur *training* diawali dengan menghapus tabel *tree*, tabel entropi dan tabel *gain* di *database* yang merupakan hasil dari proses pengujian sebelumnya. Selanjutnya proses mencari atribut yang merupakan atribut tujuan. Untuk pengujian kali ini yang menjadi atribut tujuan adalah status_jamkesmas. Prosedur ini akan melihat apakah kondisi variabel *pure* sedang *false* atau *true*. Jika kondisi variable *pure* masih *false* maka dilakukan proses pembentukan *tree*. Hal ini akan terus dilakukan sampai kondisi variabel *pure* menjadi *true*. Ditunjukkan pada *source code* 4.2 untuk prosedur hitung gain.

```
1   procedure Tftraining.hitung_gain(var
2   pilih,tujuan,history:string;proses,datatran:integer);
3   begin
4     with fdb.Qproses do
5    begin
6     Close;
7   SQL.Text := 'SELECT count(*) as tot from Training where
8   '+pilih+''+tujuan+'=:y and no_urut <= :tran';
9     ParamByName('y').Asstring := 'Y';
10    ParamByName('tran').asinteger := datatran;
11    Open;
12    s1 := Fields[0].asinteger;
13    Close;
14  SQL.Text := 'SELECT count(*) as tot from Training where
15  '+pilih+''+tujuan+'=:t and no_urut <= :tran';
16    ParamByName('t').Asstring := 'T';
17    ParamByName('tran').asinteger := datatran;
18    Open;
19    s2 := Fields[0].asinteger;
20    s:=s1+s2;
21    stot:=s;
22    node:=0;
23    nilai:='induk';
24    entropy:=(-(s1/s)*log2(s1/s))+(-(s2/s)*log2(s2/s));
25    entropy_induk:=entropy;
26    Close;
27    SQL.Text := 'insert into entropy (proses,
28  node,atribut,nilai,s,s1,s2,entropy) values (:proses,
29  :node,:atribut,:nilai,:s,:s1,:s2,:entropy)';
30    ParamByName('proses').AsInteger := proses;
31    ParamByName('node').Asinteger := node;
32    ParamByName('atribut').AsString := pilih;
33    ParamByName('nilai').asstring := nilai;
34    ParamByName('s').AsInteger := s;
35    ParamByName('s1').AsInteger := s1;
36    ParamByName('s2').AsInteger := s2;
37    ParamByName('entropy').Asfloat := entropy;
38    Execute;
39    s1:=0;
40    s2:=0;
```

```
41      s:=0;
42      Close;
43    SQL.Text := 'SELECT atribut from atribut where aktif=:y and
44    tujuan=:t '+history;
45      ParamByName('t').Asstring := 'T';
46      ParamByName('y').Asstring := 'Y';
47      Open;
48      while not fdb.qproses.eof do
49      begin
50        atribut := fdb.qproses.Fields[0].AsString;
51        inc(node);
52        entropy_anak:=0;
53        with  fdb.qproses1 do
54        begin
55         Close;
56    SQL.Text := 'SELECT distinct '+atribut+' from training
57    where no_urut <= :tran';
58            ParamByName('tran').asinteger := datatran;
59            Open;
60            while not fdb.qproses1.eof do
61               begin
62                nilai:= fdb.qproses1.Fields[0].AsString;
63                with fdb.qproses2 do
64                  begin
65                   Close;
66                   SQL.Text := 'SELECT count(*) as tot from
67    Training where '+pilih+atribut+' = '+quotedstr(nilai)+' and
68    '+tujuan+'=:y and no_urut <= :tran';
69                   ParamByName('y').Asstring := 'Y';
70                   ParamByName('tran').asinteger := datatran;
71                   Open;
72                   s1 := Fields[0].asinteger;
73                   Close;
74                   SQL.Text := 'SELECT count(*) as tot from
75    Training where '+pilih+atribut+' = '+quotedstr(nilai)+' and
76    '+tujuan+'=:t and no_urut <= :tran';
77                   ParamByName('t').Asstring := 'T';
78                   ParamByName('tran').asinteger := datatran;
79                   Open;
80                   s2 := Fields[0].asinteger;
81                 end;
82              s:=s1+s2;
83              if s>0 then
84              begin
85                if (s1=0) or (s2=0) then
86                    entropy:=0
87                else
88                    entropy:=(-(s1/s)*log2(s1/s))+(-
89    (s2/s)*log2(s2/s));
90
91                    entropy_anak:=entropy_anak+((s/stot) *
92    entropy);
93                 with fdb.qin do
94                  begin
95                   Close;
```

```
                    SQL.Text := 'insert into entropy (proses,
node,atribut,nilai,s,s1,s2,entropy) values (:proses,
:node,:atribut,:nilai,:s,:s1,:s2,:entropy)';
                    ParamByName('proses').AsInteger := proses;
                    ParamByName('node').Asinteger := node;
                    ParamByName('atribut').AsString :=
atribut;
                    ParamByName('nilai').asstring := nilai;
                    ParamByName('s').AsInteger := s;
                    ParamByName('s1').AsInteger := s1;
                    ParamByName('s2').AsInteger := s2;
                    ParamByName('entropy').Asfloat := entropy;
                    Execute;
                  end;
                end;
              fdb.qproses1.Next;
            end;
          end;
         gain:=entropy_induk - entropy_anak;
           with fdb.qin do
            begin
            Close;
            SQL.Text := 'insert into gain (proses,
node,atribut,gain) values (:proses, :node,:atribut,:gain)';
            ParamByName('proses').AsInteger := proses;
            ParamByName('node').Asinteger := node;
            ParamByName('atribut').AsString := atribut;
               ParamByName('gain').asfloat := gain;;
               Execute;
               end;
         fdb.qproses.Next;
    end;
   end;
end;
```

Source code 4.2 Prosedur perhitungan nilai gain

Setelah dilakukan perhitungan gain, kemudian program akan memanggil prosedur pembentukan *tree* dimana *node* dibuat dari atribut dengan nilai gain yang tertinggi. Ditunjukkan pada *source code* 4.3 untuk prosedur *tree*.

```
procedure Tftraining.tree(var
proses,id,datatran:integer;tujuan,pilih,his:string);
begin
if proses=1 then id:=1;
  cgain:=0;
   with  fdb.qproses do
  begin
    Close;
    SQL.Text := 'SELECT atribut,gain from gain where gain in
(select max(gain) from gain where proses=:proses)';
    ParamByName('proses').asinteger := proses;
```

```
12      Open;
13      atribut_pure := fdb.qproses.Fields[0].asstring;
14      if  fdb.qproses.Fields[1].asfloat=0 then
15      begin
16        cgain:=1;
17        exit;
18      end;
19      with fdb.qin do
20          begin
21             Close;
22              if proses=1 then
23               begin
24                  SQL.Text := 'insert into tree (node,
25  nilai,induk,internal_node,level,s1,s2) values (:node,
26  :nilai,:induk,:internal_node,:level,:s1,:s2)';
27                  ParamByName('node').AsString := atribut_pure;
28                  ParamByName('nilai').AsString := 'root';
29                  ParamByName('induk').asinteger := 0;
30                  ParamByName('internal_node').asstring := 'Y';
31                  ParamByName('level').asinteger := proses;
32                  ParamByName('s1').asinteger := 0;
33                  ParamByName('s2').asinteger := 0;
34                end
35                else
36                begin
37                  sql.text:='update tree set
38  node='+quotedstr(atribut_pure)+' where
39  id_node='+inttostr(id);
40                end;
41               Execute;
42          end;
43      Close;
44      SQL.Text := 'SELECT distinct '+atribut_pure+' from
45  training where no_urut <= :tran';
46      ParamByName('tran').asinteger := datatran;
47      Open;
48      while not fdb.qproses.Eof do
49      begin
50          nilai_pure := fdb.qproses.Fields[0].asstring;
51          s1:=0;
52          s2:=0;
53          with fdb.qproses2 do
54          begin
55            Close;
56           SQL.Text := 'SELECT count(*) as tot from Training
57  where '+pilih+atribut_pure+' = '+quotedstr(nilai_pure)+' and
58  '+tujuan+'=:y and no_urut <= :tran';
59            ParamByName('y').Asstring := 'Y';
60            ParamByName('tran').asinteger := datatran;
61           Open;
62            s1 := fdb.qproses2.Fields[0].asinteger;
63          Close;
64           SQL.Text := 'SELECT count(*) as tot from Training
65  where '+pilih+atribut_pure+' = '+quotedstr(nilai_pure)+' and
66  '+tujuan+'=:t and no_urut <= :tran';
```

```
67              ParamByName('t').Asstring := 'T';
68              ParamByName('tran').asinteger := datatran;
69               Open;
70              s2 := fdb.qproses2.Fields[0].asinteger;
71            end;
72            if (s1=0) and (s2=0) then
73            begin
74            end
75            else if s1=0 then
76            begin
77             with fdb.qin do
78              begin
79                Close;
80                SQL.Text := 'insert into tree (node,
81    nilai,induk,internal_node,level,s1,s2,confidence) values
82    (:node, :nilai,:induk,:internal_node,:level,:s1,:s2,:conf)';
83                ParamByName('node').AsString := 'T';
84                ParamByName('nilai').AsString := nilai_pure;
85                ParamByName('induk').asinteger := id;
86                ParamByName('internal_node').asstring := 'T';
87                ParamByName('level').asinteger := proses;
88                ParamByName('s1').asinteger := s1;
89                ParamByName('s2').asinteger := s2;
90                ParamByName('conf').asfloat := 1;
91                Execute;
92              end;
93            end
94            else if s2=0 then
95            begin
96               with fdb.qin do
97                begin
98                  Close;
99                  SQL.Text := 'insert into tree (node,
100   nilai,induk,internal_node,level,s1,s2,confidence) values
101   (:node, :nilai,:induk,:internal_node,:level,:s1,:s2,:conf)';
102                 ParamByName('node').AsString := 'Y';
103                 ParamByName('nilai').AsString := nilai_pure;
104                 ParamByName('induk').asinteger := id;
105                 ParamByName('internal_node').asstring := 'T';
106                 ParamByName('level').asinteger := proses;
107                 ParamByName('s1').asinteger := s1;
108                 ParamByName('s2').asinteger := s2;
109                 ParamByName('conf').asfloat := 1;
110                 Execute;
111             end;
112           end
113           else
114           begin
115            with fdb.qin do
116             begin
117               Close;
118               SQL.Text := 'insert into tree (node,
119   nilai,induk,internal_node,level,s1,s2,confidence,ket,history
120   ) values (:node,
121   :nilai,:induk,:internal_node,:level,:s1,:s2,:conf,:ket,:hist
```

```
ory)';
                ParamByName('node').AsString := 'belum ketemu';
                ParamByName('nilai').AsString := nilai_pure;
                ParamByName('induk').asinteger := id;
                ParamByName('internal_node').asstring := 'Y';
                ParamByName('level').asinteger := proses;
                ParamByName('s1').asinteger := s1;
                ParamByName('s2').asinteger := s2;
                if (s1 > s2) then
                   begin
                      ParamByName('conf').asfloat := s1/(s1+s2);
                   end
                   else
                   begin
                       ParamByName('conf').asfloat := s2/(s1+s2);
                    end;
                 ParamByName('ket').AsString :=
pilih+atribut_pure+'='+quotedstr(nilai_pure)+' and ';
                ParamByName('history').AsString :=his+' and
atribut <> '+quotedstr(atribut_pure);
                 Execute;
              end;
           end;
         fdb.qproses.Next;
      end;
   end;
end;
```

Source code 4.3 Prosedur pembentukan *tree*

Node yang pertama kali terbentuk akan dijadikan sebagai *root*, sedangkan *node* berikutnya akan menjadi *leaf* atau internal *node*. *Node* merupakan *leaf* jika *node* sudah memliliki hasil keputusan sehingga tidak memiliki anak atau cabang lagi. *Internal node* merupakan node yang masih memiliki cabang atau anak di bawahnya.

4.2.2 Proses Pruning

Proses *pruning* digunakan untuk melakukan pemotongan *tree* sehingga dihasilkan *tree* yang lebih optimal. Metode yang digunakan adalah *pruning* dengan metode *statistical bound.* Ditunjukkan pada *source code* 4.4 untuk prosedur *pruning*

```
procedure Tfpruning.pruning(var z:double);
var e,Eupper,error_leaf,error_subtree:double;
    n,et,s1,s2:integer;
    isi_node,inode: string;
begin
   with fdb.qproses do
```

```
7       begin
8           Close;
9       SQL.Text := 'select id_node, node ,s1,s2 from tree
10      where internal_node='+quotedstr('T')+' order by id_node
11      desc';
12            Open;
13             while not fdb.qproses.Eof do
14               begin
15                   id_node := Fields[0].Asinteger;
16                   isi_node :=Fields[1].AsString;
17                   s1:= Fields[2].asinteger;
18                   s2:=Fields[3].asinteger;
19                   n:=s1+s2;
20                   et:=0;
21                   if isi_node='Y' then
22                     et:=s1
23                   else if isi_node='T' then
24                     et:=s2;
25                   e:=et/n;
26      Eupper:= ((e+((z*z)/(2*n)))+(z*sqrt((e*(1-
27      e))/n)+((z*z)/(4*n*n)))) /(1+((z*z)/n));
28              error_leaf:=Eupper*n;
29               with fdb.qin do
30                begin
31                  Close;
32                  SQL.Text := 'update tree set
33      error_leaf=:etot where id_node=:node';
34                  ParamByName('etot').Asfloat := error_leaf;
35                  ParamByName('node').asinteger := id_node;
36                  Execute;
37                end;
38               fdb.qproses.Next;
39              end;
40          end;
41          with fdb.qproses1 do
42          begin
43             Close;
44      SQL.Text := 'select id_node, s1,s2 from tree where
45      nilai<>'+quotedstr('root')+' and
46      internal_node='+quotedstr('Y')+' order by id_node
47      desc';
48             Open;
49
50           while not fdb.qproses1.Eof do
51            begin
52               id_node := Fields[0].Asinteger;
53               s1:= Fields[1].asinteger;
54                s2:=Fields[2].asinteger;
55                n:=s1+s2;
56                    if s1>s2 then
57                    begin
58                      et:=s1;
59                      confidence:=s1/(s1+s2);
60                    end
61                    else
```

```
                begin
                  et:=s2;
                  confidence:=s2/(s1+s2);
                end;
                e:=et/n;
Eupper:= ((e+((z*z)/(2*n)))+(z*sqrt((e*(1-
e))/n)+((z*z)/(4*n*n)))) /(1+((z*z)/n));
                error_subtree:=Eupper*n;
              with fdb.qproses2 do
              begin
                Close;
                SQL.Text := 'select sum(error_leaf) as tot
from tree where induk=:induk';
                ParamByName('induk').asinteger := id_node;
                Open;
                error_leaf:=Fields[0].asfloat;
              end;
             if error_subtree < error_leaf then
              begin
                if s1>s2 then
                  inode:='Y'
                 else
                  inode:='T';
                with fdb.qin do
                 begin
                 Close;
SQL.Text := 'update tree set confidence=:conf,
error_subtree=:esub, error_leaf=:eleaf, node=:inode,
internal_node=:t, cek_prune=:y where id_node=:node';
             ParamByName('conf').Asfloat := confidence;
             ParamByName('esub').Asfloat := error_subtree;
             ParamByName('eleaf').Asfloat :=
error_subtree;
             ParamByName('node').asinteger := id_node;
             ParamByName('y').asstring := 'Y';
             ParamByName('t').asstring := 'T';
             ParamByName('inode').asstring := inode;
             Execute;
Close;
SQL.Text := 'delete from tree where induk=:node';
           ParamByName('node').asinteger := id_node;
           Execute;
         end;
              end
              else
              begin
                with fdb.qin do
                  begin
               Close;
               SQL.Text := 'update tree set
error_subtree=:esub, error_leaf=:eleaf, cek_prune=:y
where id_node=:node';
              ParamByName('esub').Asfloat :=
error_subtree;
              ParamByName('eleaf').Asfloat := error_leaf;
```

```
117                 ParamByName('node').asinteger := id_node;
118                 ParamByName('y').asstring := 'Y';
119                 Execute;
120                 end;
121               end;
122             fdb.qproses1.Next;
123           end;
124       end;
125   end;
```

Source code 4.4 Prosedur *pruning* dengan *Statistical Bound*

Proses *pruning* yang dilakukan adalah membandingkan nilai *error rate* sebuah *internal node* dengan nilai *error rate* seumpama *internal node* tersebut diubah menjadi sebuah *leaf*. Jika nilai *error rate leaf* lebih sedikit maka *leaf* tadi akan menggantikan *internal node* tersebut, jika hal ini terjadi, berarti proses *pruning* dijalankan.

4.2.3 Proses Pembentukan Rule

Proses pembentukan *rule* diperlukan untuk mengubah dari bentuk *tree* menjadi suatu *rule*. Ditunjukkan pada *source code* 4.5 untuk prosedur pembentukan *rule.*

```
1   procedure Tfrule.rule();
2   var root:boolean;  nilai,jawaban:string;
3   begin
4     teks_rule:='';
5     with fdb.qproses do
6     begin
7       close;
8       SQL.Text := 'SELECT ID_NODE, NODE, INDUK, nilai ,
9   confidence from tree where internal_node = :t ORDER BY
10  ID_NODE';
11      ParamByName('t').asstring := 'T';
12      Open;
13      while not fdb.qproses.Eof do
14        begin
15          jawaban:=fdb.qproses.Fields[1].asstring;
16          if jawaban='T' then jawaban:='Tidak'
17          else
18            jawaban:='Ya';
19          nilai:=fdb.qproses.Fields[3].asstring;
20          induk:= fdb.qproses.Fields[2].asinteger;
21          jawaban:=jawaban+'
22  (confidence:'+floattostr(fdb.qproses.Fields[4].asfloat)
23  +')';
24          root:=false;
25          while root=false do
26          begin
27            with fdb.qproses1 do
28             begin
```

```
29              close;
30              SQL.Text := 'SELECT ID_NODE, NODE, INDUK,
31    nilai from tree where id_node = :induk ';
32              ParamByName('induk').asinteger := induk;
33              Open;
34            end;
35            if fdb.qproses1.RecordCount = 0 then
36            begin
37               root:=true;
38            end
39            else
40            begin
41              induk:= fdb.qproses1.Fields[2].asinteger;
42              if induk = 0 then
43              begin
44                root:=true;
45                teks := 'JIKA
46    '+fdb.qproses1.Fields[1].asstring+teks;
47                teks := teks+' = '+nilai+' MAKA
48    '+jawaban+#13#10#13#10;
49                teks_rule:=teks_rule+teks;
50              end
51              else
52              begin
53                teks := ' =
54    '+fdb.qproses1.Fields[3].asstring+' DAN
55    '+fdb.qproses1.Fields[1].asstring+' '+teks;
56                root:=false;
57              end;
58            end;
59          end;
60          fdb.qproses.next;
61           teks:='';
62        end;
63    end;
64  end;
```

Source code 4.5 Pembentukan *rule*

Rule dibuat dari *tree* yang sudah terbentuk dari proses *training* yang dilakukan. *Rule* merupakan terjemahan *tree* kedalam kondisi *if..else..then* .., proses penerjemahan dilakukan dari posisi *root* sebagai induk kemudian turun ke anaknya (*subtree*) sampai ditemukan *leaf*, selanjutnya dilakukan proses penelusuran yang sama untuk *leaf-leaf* lainya. Proses ini akan berakhir sampai semua *leaf* berhasil ditelusuri.

4.2.4 Proses Testing

Pada proses *testing* terdapat dua buah prosedur yang digunakan yaitu prosedur *testing* dan prosedur hitung akurasi. Ditunjukkan pada *source code* 4.6 untuk prosedur *testing*

```
procedure Tftesting.testing();
var ketemu,root:boolean;
no_urut:integer;atribut,nilai_atribut,hasil:string;
begin
with fdb.qproses1 do
        begin
        Close;
SQL.Text := 'SELECT no_urut from testing';
        Open;
        while not fdb.qproses1.eof do
        begin
        ketemu:=false;
root:=true;
        no_urut:=Fields[0].Asinteger;
        while ketemu=false do
        begin
        if root = true then
        begin
        with fdb.qproses2 do
        begin
        Close;
        SQL.Text := 'select id_node, node from tree
where induk = 0';
 Open;
atribut := Fields[1].AsString;
        id_node := Fields[0].AsInteger;
 root:=false;
            end;
        end
        else
        begin
with fdb.qproses2 do
        begin
        close;
        SQL.Text := 'SELECT '+atribut+' from testing
where no_urut = :no ';
        ParamByName('no').AsInteger := no_urut;
        Open;
        nilai_atribut:= Fields[0].Asstring;
close;
        SQL.Text := 'select id_node, node,
internal_node, confidence '+
        ' from tree '+
' where induk  = :induk '+
' and nilai = :nilai ';
        ParamByName('induk').AsInteger := id_node;
        ParamByName('nilai').AsString := nilai_atribut;
Open;
```

```
49          if Fields[2].AsString = 'Y' then
50          begin
51          atribut := Fields[1].AsString;
52          id_node := Fields[0].AsInteger;
53          ketemu:=false;
54  end else
55          begin
56          hasil:=Fields[1].AsString;
57          confidence:= Fields[3].Asfloat;
58           SQL.Text := 'UPDATE testing SET status_prediksi
59  = :hasil , confidence= :conf where no_urut = :no ';
60    ParamByName('no').AsInteger := no_urut;
61           ParamByName('hasil').Asstring := hasil;
62           ParamByName('conf').asfloat := confidence;
63          execute;
64
65          ketemu:=true;
66          end;
67              end;
68          end;
69          end;
70          fdb.qproses1.Next;
71        end;
72    end;
73  end;
```

Source code 4.6 Prosedur *testing*

Prosedur *testing* digunakan untuk melakukan uji coba terhadap *rule* yang sudah terbentuk dan diterapkan pada data *testing*. Kemudian hasilnya akan dibandingkan apakah sama antara hasil *survey* dengan hasil dari proses *testing*. Prosedur hitung akurasi dilakukan untuk mengetahui kinerja dari hasil testing dengan menggunakan *tree* yang telah terbentuk sebelumnya, dimana kinerja ini terdiri dari nilai *precission*, *recall*, dan *accuracy*. Ditunjukkan pada *source code* 4.7 untuk prosedur hitung akurasi serta rata-rata *confidence*.

```
1   procedure Tftesting.Hitung_akurasi();
2   begin
3      with fdb.Qproses do
4      begin
5       Close;
6       SQL.Text := 'SELECT count(*) from testing where
7   status_jamkesmas=:j and status_prediksi=:p';
8       ParamByName('j').AsString := 'Y';
9       ParamByName('p').Asstring := 'Y';
10      Open;
11      a:= Fields[0].Asinteger;
12       Close;
13  SQL.Text := 'SELECT count(*) from testing where
```

```
14      status_jamkesmas=:j and status_prediksi=:p';
15          ParamByName('j').AsString := 'Y';
16          ParamByName('p').Asstring := 'T';
17          Open;
18          b:= Fields[0].Asinteger;
19          Close;
20      SQL.Text := 'SELECT count(*) from testing where
21      status_jamkesmas=:j and status_prediksi=:p';
22          ParamByName('j').AsString := 'T';
23          ParamByName('p').Asstring := 'Y';
24          Open;
25          c:= Fields[0].Asinteger;
26          Close;
27      SQL.Text := 'SELECT count(*) from testing where
28      status_jamkesmas=:j and status_prediksi=:p';
29          ParamByName('j').AsString := 'T';
30          ParamByName('p').Asstring := 'T';
31          Open;
32          d:= Fields[0].Asinteger;
33          Close;
34      SQL.Text := 'SELECT avg(confidence) from testing';
35          Open;
36          confidence:= Fields[0].asfloat;
37        end;
38        precision:=(d/(b+d))*100;
39         recall:=(d/(c+d))*100;
40         accuracy:=((a+d)/(a+b+c+d)) * 100;
41      end;
```

Source code 4.7 Prosedur hitung akurasi

4.2.5 Proses Kombinasi Atribut

Proses kombinasi atribut digunakan untuk membuat kombinasi dari atribut yang digunakan pada proses *training*. Hal ini dilakukan untuk melihat kombinasi atribut mana saja yang memiliki nilai akurasi paling baik. Ditunjukkan pada *source code* 4.8 untuk prosedur kombinasi

```
1       procedure Tfkombinasi.reset_atribut();
2       begin
3          with fdb.qin do
4          begin
5              close;
6              SQL.Text := 'UPDATE atribut SET
7       aktif='+quotedstr('T')+' where
8       atribut<>'+quotedstr('status_jamkesmas');
9               execute;
10         end;
11      end;
12
13      procedure Tfkombinasi.set_atribut(var nama:string);
14      begin
```

```
15      with fdb.qin do
16      begin
17          close;
18          SQL.Text := 'UPDATE atribut SET
19      aktif='+quotedstr('Y')+' where atribut =:nama';
20          ParamByName('nama').Asstring := nama;
21                              execute;
22      end;
23  end;
```

Source code 4.8 Proses kombinasi atribut

Pada proses kombinasi atribut terdapat beberapa prosedur dan fungsi antara lain :

a. Prosedur reset_atribut : untuk melakukan *reset* status atribut menjadi tidak aktif semua
b. Prosedur set_atribut : untuk melakukan perubahan status suatu atribut menjadi aktif

4.3 Penerapan Sistem

Sistem yang telah dibangun ini digunakan untuk mengimplentasikan algoritma *decision tree* C4.5 untuk menentukan klasifikasi keluarga peserta jamkesmas berdasarkan kemiskinan di Kabupaten Blitar. Beberapa tampilan sistem pada tiap-tiap menu akan ditunjukkan sebagai berikut :

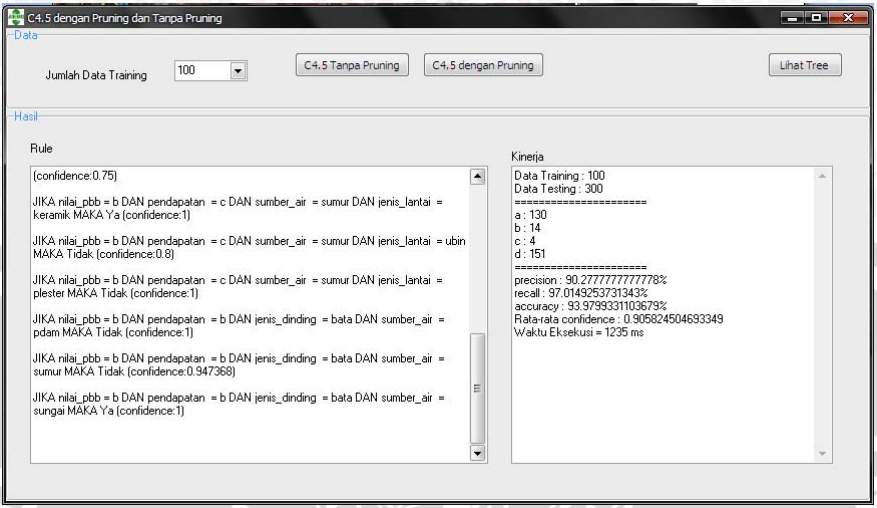
Gambar 4. 1 Tampilan menu utama

Gambar 4.1 merupakan tampilan utama ketika aplikasi dijalankan. Pada tampilan utama terdapat menu yang terdiri dari :

1. File, yang memiliki sub menu ’hasil pengujian’ dan sub menu ’Exit’
2. Menu, yang memiliki sub menu ’C4.5 tanpa pruning dan dengan pruning’ dan sub menu ’C4.5 dengan kombinasi atribut’
3. Setting, yang memiliki sub menu ’Setting database’.

4.3.1 Form Menu tanpa pruning dan dengan pruning

Pada menu ini dilakukan proses pembentukan *tree* baik tanpa *pruning* maupun dengan *pruning*. Jumlah atribut yang digunakan sebanyak 7 buah atribut. Sedangkan jumlah data *training* yang digunakan sebanyak 100, 500, 1000, 1500, dan 2000 data. Untuk data *testing* digunakan data tetap yaitu sebanyak 300 data. Berikut ini adalah tampilan menu C4.5 dengan *pruning* dan *tanpa pruning*.

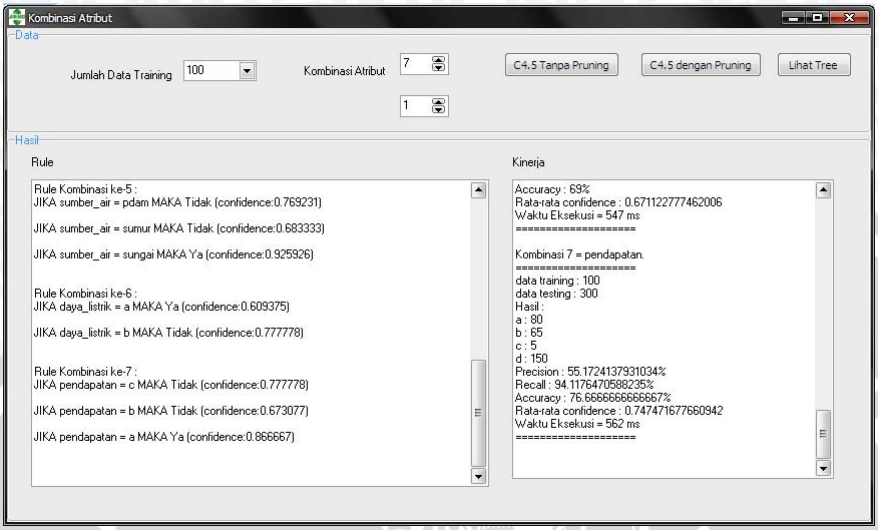


Gambar 4.2 Tampilan C4.5 dengan *pruning* dan tanpa *pruning*

Pada Gambar 4.2 terlihat form ini memiliki 2 buah bagian yaitu bagian atas sebagai tempat pemilihan jumlah data dan tombol perintah perhitungan apakah menggunakan *pruning* atau tanpa menggunakan proses *pruning*, sedangkan bagian bawah untuk menampilkan hasil yang berupa *rule* dan nilai kinerja dari proses perhitungan C4.5 dengan *pruning* atau tanpa *pruning*.

4.3.2 Form Menu dengan Kombinasi Atribut

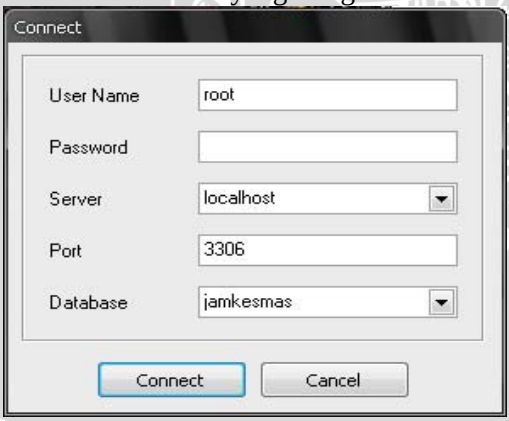
Pada menu ini hampir sama dengan menu yang pertama hanya saja terdapat proses kombinasi atribut. Sehingga dengan adanya proses kombinasi atribut ini, diharapkan akan diperoleh kombinasi atribut yang menghasilkan *tree* dengan hasil *testing* yang memiliki nilai akurasi terbaik. Berikut ini adalah tampilan menu Kombinasi Atribut



Gambar 4.3 Tampilan C4.5 dengan *pruning* dan tanpa *pruning* dengan kombinasi atribut

4.3.3 Form Menu Setting Database

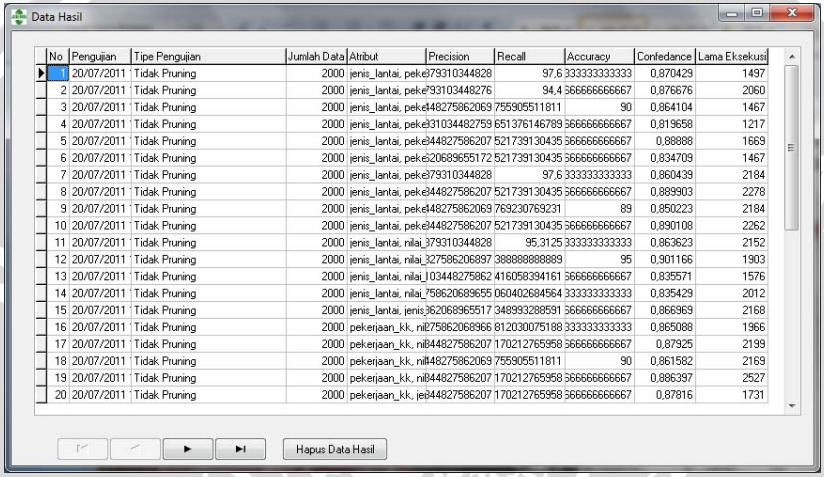
Form menu Setting Databse digunakan untuk melakukan setting koneksi database yang meliputi *username*, *password*, nama *server*, *port* yang digunakan dan nama Database. Jika sudah diisi sesuai dengan koneksi *database* yang diinginkan.



Gambar 4.4 *Setting* koneksi *database*

4.3.4 Form Hasil Pengujian

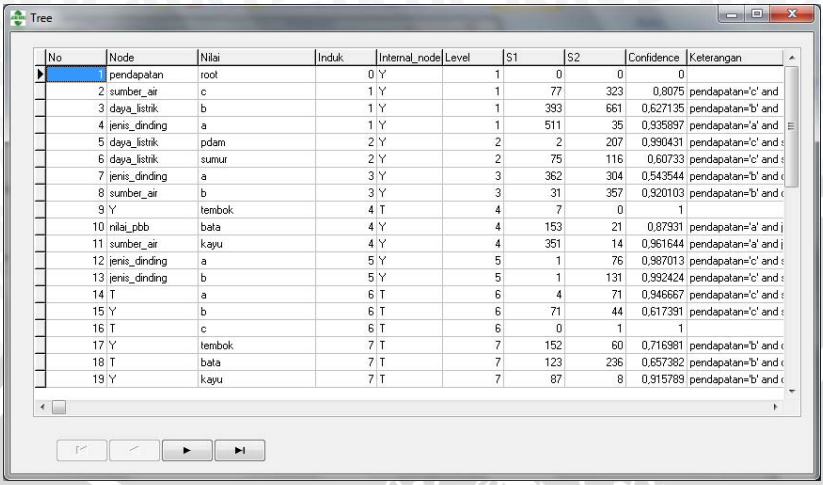
Form hasil pengujian digunakan untuk menyimpan hasil dari pengujian yang telah dilakukan. Data yang disimpan meliputi nilai *precision*, *recall*, *accuracy*, rata-rata *confidence* dan waktu eksekusi *testing*. Pada form ini juga terdapat tombol untuk menghapus data hasil pengujian.



Gambar 4.5 Form hasil pengujian

4.3.5 Form *Tree*

Form *tree* merupakan form untuk menampilkan hasil *tree* yang terbentuk dari proses *training*, ini dapat digunakan untuk pengecekan hasil *testing* yang diperoleh sudah sesuai dengan *tree* atau belum.



Gambar 4.6 Form *Tree*

4.4 Pengujian Sistem

Sistem ini juga digunakan untuk menguji kinerja dari hasil klasifikasi yang telah dibangun. Sistem akan memberikan hasil kinerja berupa prosentase jumlah benar dan salah. Sehingga dapat diketahui berapa nilai kinerjanya yang meliputi nilai *precision*, *recall* dan *Accuracy*. Selain itu besar rata-rata nilai *confidence* dan waktu eksekusi pada saat proses *testing* juga disimpan. Berikut ini adalah hasil dari pengujian aplikasi :

Tabel 4.1 Data hasil pengujian C4.5 tanpa *pruning*

Jumlah Data	*Precision*	*Recall*	*Accuracy*	*Confidence*	Waktu (ms)
100	90.278 %	97.015 %	93.980 %	0.90582	1201
500	98.621 %	89.937 %	94.000 %	0.91470	1591
1000	88.276 %	96.970 %	93.000 %	0.90301	1154
1500	88.276 %	96.970 %	93.000 %	0.88264	1201
2000	98.621 %	95.333 %	97.000 %	0.93073	1497

Pada Tabel 4.1 nilai *precision* yang tertinggi pada data *training* sebanyak 500 dan 2000 senilai 98,621% , sedangkan untuk nilai *recall* tertinggi pada data *training* sebanyak 100 senilai 97.015 %. Nilai *accuracy* tertinggi pada jumlah data 2000 senilai 97% dan nilai waktu eksekusi *testing* tercepat pada data 500 senilai 1154 milidetik.

Tabel 4.2 Data hasil pengujian C4.5 dengan *pruning*

Jumlah Data	*Precision*	*Recall*	*Accuracy*	*Confidence*	Waktu (ms)
100	75.000 %	92.308 %	84.950 %	0.80633	577
500	75.172 %	92.373 %	85.000 %	0.82252	420
1000	51.034 %	97.368 %	75.667 %	0.75710	405
1500	51.034 %	97.368 %	75.667 %	0.74811	421
2000	55.172 %	94.118 %	76.667 %	0.74435	515

Pada Tabel 4.2 nilai *precision* yang tertinggi pada data *training* sebanyak 500 data yaitu senilai 75,172% , sedangkan untuk nilai *recall* tertinggi pada data *training* sebanyak 1000 data dan 1500 data yaitu senilai 97,368%. *Accuracy* tertinggi pada jumlah data *training* 500 data yaitu senilai 85% dan nilai waktu eksekusi *testing* terbaik pada jumlah data *training* 1000 data yaitu senilai 405 milidetik.

Tabel 4.3 Data hasil pengujian C4.5 dengan kombinasi atribut tanpa *pruning* untuk *precision* terbaik

Kombinasi	Atribut	*Precision*
1 atribut	daya_listrik.	83,448 %
2 atribut	daya_listrik, pendapatan.	91,030 %
3 atribut	pekerjaan_kk, sumber_air, pendapatan.	91,035 %
4 atribut	jenis_lantai, jenis_dinding, sumber_air, pendapatan.	94,480 %
5 atribut	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	94,483 %
6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	98,621 %
7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	98,620 %

Pada Tabel 4.3 nilai *precision* terbaik pada data *training* dengan 6 atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan) yaitu senilai 98,621%.

Tabel 4.4 Data hasil pengujian kombinasi atribut tanpa *pruning* untuk *recall* terbaik

Kombinasi	Atribut	*Recall*
1 atribut	sumber_air.	97,370 %
2 atribut	jenis_lantai, jenis_dinding.	97,650 %
3 atribut	jenis_lantai, pekerjaan_kk, sumber_air.	98,170 %
4 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air.	98,164 %
5 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik.	98,165 %
6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	97,600 %
7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	95,330 %

Pada Tabel 4.4 nilai *recall* terbaik pada data *training* dengan 3 atribut (jenis_lantai, pekerjaan_kk, sumber_air) yaitu senilai 98,170 %.

Tabel 4.5 Data hasil pengujian kombinasi atribut tanpa *pruning* untuk *Accuracy* terbaik

Kombinasi	Atribut	*Accuracy*	*Confidence*
1 atribut	nilai_pbb.	85,00%	0,77239
2 atribut	nilai_pbb, jenis_dinding.	89,33%	0,83081
3 atribut	jenis_lantai, sumber_air, pendapatan.	93,67%	0,88465
4 atribut	jenis_lantai, jenis_dinding, sumber_air, pendapatan.	95,00%	0,90113
5 atribut	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	95,00 %	0,9012
6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	97,00%	0,93073
7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	97,00%	0,93073

Pada Tabel 4.5 nilai *accuracy* terbaik pada data *training* dengan 6 atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan) dan 7 atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan) yaitu senilai 97 %.

Tabel 4.6 Data hasil pengujian kombinasi atribut tanpa *pruning* untuk waktu eksekusi *testing* terbaik

Kombinasi	Atribut	Waktu(ms)
1 atribut	pendapatan.	389
2 atribut	nilai_pbb, pendapatan.	405
3 atribut	nilai_pbb, sumber_air, pendapatan.	764
4 atribut	pekerjaan_kk, nilai_pbb, jenis_dinding,	842

	pendapatan.	
5 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik.	917
6 atribut	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	1186
7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	1513

Pada Tabel 4.6 waktu eksekusi *testing* terbaik pada data *training* dengan atribut (pendapatan) yaitu senilai 389 milidetik.

Tabel 4.7 Data hasil pengujian kombinasi atribut dengan *pruning* untuk *precision* terbaik

Kombinasi	Atribut	*Precision*
1 atribut	daya_listrik.	83,448%
2 atribut	pekerjaan_kk, nilai_pbb.	75,172%
3 atribut	pekerjaan_kk, nilai_pbb, jenis_dinding.	75,182%
4 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, daya_listrik.	75,182%
5 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik.	75,172%
6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	51,034%
7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	55,172%

Pada Tabel 4.7 nilai *precision* terbaik pada data *training* dengan 1 atribut (daya_listrik) yaitu senilai 83,448 %.

Tabel 4.8 Data hasil pengujian kombinasi atribut dengan *pruning* untuk *recall* terbaik

Kombinasi	Atribut	*Recall*
1 atribut	nilai_pbb.	92,373%
2 atribut	sumber_air, daya_listrik.	97,377%
3 atribut	pekerjaan_kk, sumber_air, daya_listrik.	97,377%

4 atribut	jenis_lantai, jenis_dinding, sumber_air, daya_listrik.	97,368%
5 atribut	pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	97,377%
6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	97,367%
7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	94,118%

Pada Tabel 4.8 nilai *recall* terbaik pada data *training* dengan 2 atribut (sumber_air, daya_listrik) dan 3 atribut (pekerjaan_kk, sumber_air, daya_listrik) dan 5 atribut (pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik) yaitu senilai 97,377%.

Tabel 4.9 Data hasil pengujian kombinasi atribut dengan *pruning* untuk *Accuracy* terbaik

Kombinasi	Atribut	*Accuracy*	*Confidence*
1 atribut	nilai_pbb.	84,950%	0,77239
2 atribut	pekerjaan_kk, nilai_pbb.	84,960%	0,77239
3 atribut	jenis_lantai, nilai_pbb, daya_listrik.	85,000%	0,77239
4 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, daya_listrik.	84,956%	0,77439
5 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik.	84,950%	0,77239
6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	76,867%	0,74829
7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	76,667%	0,74435

Pada Tabel 4.9 nilai *accuracy* terbaik pada data *training* dengan 3 atribut (jenis_lantai, jenis_dinding, sumber_air, daya_listrik) yaitu senilai 85 %.

Tabel 4.10 Data hasil pengujian kombinasi atribut dengan *pruning* untuk waktu eksekusi *testing* terbaik

Kombinasi	Atribut	Waktu(ms)
1 atribut	pekerjaan_kk.	405
2 atribut	jenis_dinding, daya_listrik.	400
3 atribut	jenis_lantai, daya_listrik, pendapatan.	389
4 atribut	jenis_lantai, jenis_dinding, sumber_air, daya_listrik.	388
5 atribut	jenis_lantai, pekerjaan_kk, jenis_dinding, daya_listrik, pendapatan.	389
6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	403
7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	514

Pada Tabel 4.10 waktu eksekusi *testing* terbaik pada data *training* dengan 4 atribut (jenis_lantai, jenis_dinding, sumber_air, daya_listrik) yaitu senilai 388 milidetik.

Untuk pengujian dengan kombinasi atribut, tabel yang ditampilkan hanya kombinasi terbaik pada tiap jumlah data pengujian, sedangkan data hasil kombinasi secara lengkap dapat dilihat pada halaman lampiran.

4.5 Analisa hasil

Setelah diperoleh data hasil pengujian, diketahui untuk nilai *precision* yang terbaik terdapat pada pengujian tanpa *pruning* dengan 6 atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan) dan dengan 7 atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan) senilai 98,621%. Untuk nilai *recall* yang terbaik terdapat pada pengujian tanpa *pruning* dengan 5 atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik) dan 3 atribut (jenis_lantai, pekerjaan_kk, sumber_air) senilai 98,165%. Untuk nilai *accuracy* terbaik terdapat pada pengujian tanpa *pruning* dengan

6 atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan) dan 7 atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan) senilai 97 %. Untuk nilai waktu eksekusi testing terbaik terdapat pada pengujian menggunakan *pruning* dengan 4 atribut (jenis_lantai, jenis_dinding, sumber_air, daya_listrik) senilai 388 milidetik .Hasil pengujian terbaik ini ditunjukkan pada tabel 4.11

Tabel 4.11 Hasil Pengujian Terbaik

No	Kombinasi	Atribut	Pruning	Hasil Terbaik	Nilai
1	6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	Tanpa *Pruning*	*Precision*	98,621%
2	7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	Tanpa *Pruning*	*Precision*	98,621%
3	5 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik.	Tanpa *Pruning*	*Recall*	98,165%
4	3 atribut	jenis_lantai, pekerjaan_kk, sumber_air.	Tanpa *Pruning*	*Recall*	98,165%
4	6 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	Tanpa *Pruning*	*Accuracy*	97,000%

5	7 atribut	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	Tanpa *Pruning*	*Accuracy*	97,000%
6	4 atribut	jenis_lantai, jenis_dinding, sumber_air, daya_listrik.	*Pruning*	Waktu eksekusi *testing*	388 ms

Accuracy merupakan tingkat ketepatan hasil klasifikasi dengan data sebenarnya, Nilai *confidence* dipengaruhi oleh besarnya nilai *accuracy*, semakin tinggi nilai *accuracy* maka nilai *confidence* akan semakin tinggi. Nilai *confidence* diperoleh dari tingkat *pure* data *training* sehingga semakin tinggi tingkat *pure* maka semakin tinggi nilai *accuracy* yang diperoleh. Hal ini terjadi baik pada pengujian dengan *pruning* maupun tanpa *pruning*.

1. Pengaruh Jumlah Data *Training* terhadap *precision*, *recall* dan *accuracy*

Penambahan jumlah data secara umum mengakibatkan meningkatnya nilai *precision* dan *accuracy*, namun penambahan jumlah data ini menyebabkan nilai *recall* mengalami penurunan. Hal ini disebabkan karena data yang digunakan memiliki pola ketika data yang diharapkan diidentifikasi tidak sesuai atau tidak relevan mengalami penurunan sedangkan data yang tidak diharapkan diidentifikasi sesuai atau relevan mengalami kenaikan, hal ini yang menyebabkan ketika nilai *precision* mengalami kenaikan maka nilai *recall* mengalami penurunan. Nilai *precision* dan *accuracy* semakin tinggi ketika jumlah data bertambah karena *tree* yang terbentuk lebih baik hasil dari proses *training* yang dilakukan lebih banyak sesuai dengan banyaknya jumlah data *training*

2. Pengaruh *precision* dan *recall* terhadap *accuracy*

Jika nilai *precision* tinggi secara umum menyebabkan nilai *accuracy* menjadi tinggi, sebaliknya jika nilai *recall* tinggi secara umu menyebabkan nilai *accuracy* menjadi rendah. Dengan kata lain, nilai *accuracy* berbanding lurus dengan nilai *precision* dan berbanding terbalik dengan nilai *recall*. Hal ini menunjukkan bahwa hasil pengujian tersebut sudah baik, karena nilai *precision* tertinggi adalah 98,621 %

3. Pengaruh jumlah data *training*, *precision*, *recall* dan *accuracy* terhadap waktu eksekusi *testing*

Penambahan jumlah data *training* tidak selalu menyebabkan waktu eksekusi testing semakin lambat, hal ini disebabkan pada kondisi tertentu data training cenderung seragam menyebabkan *rule* yang terbentuk lebih ringkas sehingga waktu eksekusi *testing* lebih cepat. Bertambahnya nilai *precision* dan *accuracy* secara umum menyebabkan waktu eksekusi testing menjadi lebih lambat hal ini karena *tree* yang terbentuk menjadi semakin banyak sehingga waktu eksekusi *testing* menjadi lebih lambat. Sedangkan untuk kenaikan nilai *recall* secara umum menyebabkan waktu eksekusi menjadi lebih cepat. Hal ini dikarenakan *tree* yang terbentuk lebih sedikit sehingga waktu *testing* lebih cepat.

4. Pengaruh kombinasi atribut terhadap *precision*, *recall*, *accuracy* dan waktu eksekusi

Pengujian dengan kombinasi atribut memungkinkan untuk mendapatkan waktu eksekusi *testing* lebih cepat daripada pengujian tanpa kombinasi atribut, hal ini disebabkan pada pengujian dengan kombinasi atribut alternatif pengujian lebih banyak sehingga bisa jadi diperoleh waktu eksekusi *testing* lebih cepat pada kombinasi atribut tertentu yang memiliki *rule* yang terbentuk lebih ringkas. Semakin banyak jumlah atribut yang digunakan untuk kombinasi, besar waktu eksekusi *testing* akan semakin meningkat. Hal ini disebabkan jumlah *rule* yang terbentuk akan semakin banyak.

Penambahan jumlah atribut pada kombinasi atribut secara umum menyebabkan kenaikan pada nilai *precision* dan nilai *accuracy*, namun menyebabkan penurunan nilai pada *recall*. Hal ini

terjadi dikarenakan tree yang terbentuk dari proses training lebih sedikit

5. Pengaruh nilai *confidence* terhadap *precision*, *recall* dan *accuracy*

Nilai *precision* juga berpengaruh terhadap nilai *confidence*, semakin tinggi nilai *precision* maka nilai *confidence* akan semakin tinggi. Sedangkan nilai *recall* tidak berpengaruh secara langsung terhadap nilai *confidence* karena pada beberapa pengujian nilai *recall* mengalami kenaikan sedangkan nilai *confidence* mengalami penurunan namun pada pengujian yang lain nilai *recall* mengalami kenaikan, nilai *confidence* juga mengalami kenaikan. Nilai *accuracy* sangat dipengaruhi oleh besarnya nilai *confidence*, hal ini terjadi karena semakin besar tingkat *accuracy* suatu *tree* maka semakin tinggi pula tingkat *confidence* dimana ini sangat dipengaruhi oleh tingkat *pure* pada masing-masing *node*.

6. Perbandingan antara pengujian dengan *pruning* dan tanpa *pruning*

Pengujian dengan menggunakan *pruning* memiliki waktu eksekusi *testing* yang lebih cepat dibandingkan tanpa pruning hal ini disebabkan karena *rule* yang terbentuk lebih ringkas. Untuk nilai *precision* dan nilai *accuracy* pada pengujian tanpa *pruning* secara umum memiliki nilai lebih besar dibandingkan pada pengujian dengan *pruning*, sedangkan untuk nilai *recall* pada pengujian dengan pruning memiliki nilai yang lebih besar. Hal ini sangat dipengaruhi dengan bentuk dari *tree* yang terbentuk dari proses *training* dan dari proses *pruning* untuk pengujian dengan *pruning*. *Tree* yang terbentuk untuk pengujian dengan *pruning* memiliki bentuk *tree* yang lebih ringkas sehingga ketika proses *testing* waktu eksekusi yang didapat lebih cepat namun hal ini berakibat pada semakin rendahnya nilai *accuracy*.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Beberapa kesimpulan yang dapat diambil dari hasil penelitian ini adalah :

1. Pada penelitian ini telah berhasil dilakukan klasifikasi keluarga miskin dengan algoritma *decision tree* C4.5 dengan hasil akurasi terbaik senilai 97% dan akurasi terburuk senilai 66,667%
2. Kinerja algoritma *decision tree* C4.5 sebagai berikut :
   a. Kinerja algoritma *decision tree* C4.5 untuk pengujian tanpa kombinasi atribut adalah nilai *precision* terbaik senilai 98,621% untuk jumlah data *training* 500 data dan 2000 data pada pengujian tanpa *pruning*, nilai *recall* terbaik senilai 97,368% untuk jumlah data *training* 1000 data dan 1500 data pada pengujian dengan *pruning*, nilai *accuracy* terbaik senilai 97 % untuk jumlah data *training* 2000 data pada pengujian tanpa *pruning*, dan nilai waktu eksekusi *testing* terbaik senilai 405 milidetik untuk jumlah data *training* 100 data pada pengujian dengan *pruning*
   b. Kinerja algoritma *decision tree* C4.5 untuk pengujian dengan kombinasi atribut adalah nilai *precision* terbaik senilai 98,621%, nilai *recall* terbaik senilai 98,165% , nilai *accuracy* terbaik senilai 97 % , dan nilai waktu eksekusi *testing* terbaik senilai 388 milidetik
3. Atribut (jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan) memiliki hasil accuracy terbaik, sehingga semakin banyak atribut yang digunakan untuk kombinasi atribut maka nilai *precision* dan nilai *accuracy* menjadi lebih tinggi

5.2 Saran

Saran yang dapat diberikan setelah pengerjaan skripsi ini meliputi:

1. Kombinasi atribut dengan hasil *accuracy* terbaik dapat digunakan sebagai atribut dalam implementasi sistem ini pada Program Jamkesmas di Kabupaten Blitar

2. Sebaiknya dilakukan perbandingan dengan metode algoritma *decision tree* atau algoritma data mining yang lain sehingga dapat diperoleh hasil kinerja yang paling baik

DAFTAR PUSTAKA

Adriaans. P. and D. Zantinge, 1996, *Data Mining*, Addison-Wesley, New York

Budihardja, 2009, *Petunjuk Teknis Jamkesmas*, Direktorat Jendral Bina Masyarakat, Jakarta

Dwi widiastuti, 2008, *Analisa Perbandingan Algoritma Svm, Naive Bayes, Dan Decision Tree Dalam Mengklasifikasikan Serangan (Attacks) Pada Sistem Pendeteksi Intrusi*, Universitas Gunadarma, Jakarta

Frank, V. 2003. *Classification Trees : C4.5*.Universit Libre de Bruxelles.

George H. John, 1994, *Cross-validated C4.5:Using Error Estimation for Automatic Parameter Selection*, Stanford University

Han, J and Khamber, M. 2001. *Data Mining : Concepts and Techniques*, Morgan Kaufmann Publishers, San Francisco,USA.

Jose Martinez, Olac Fuentes, 2005, *Using C4.5 as Variable selection Criterion in Classification Tasks*, National Institute of Astrophysic, Optics and Electronics, Mexico

Kantardzic, M. 2003. *Data Mining: Concepts, Models, Methods, and Algorithms. Subbab Data-Mining Process*. John Wiley & Sons. New Jersey.

Kohavi, R and Quinlan, R. 1999. *Decision Tree Discovery. 1-16*. AAAI and The MIT Pres.

Kusrini & Emha Taufiq Luthfi, 2009, *Algoritma Data Mining*, Andi Offset

Larose, Daniel. 2005.*Data Mining methods and Model*, Wiley-Interscience, New Jersey.

Larose, D. 2005. *Discovering Knowledge in Data. An Introduction to Data Mining*. John Wiley & Sons. New Jersey.

Mannila, Smyth, and Hand, David.2001. *Principle of Data Mining*. MIT Press, Cambridge.

Quinlan, R. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann. San Mateo.

Tan, Pang-Ning, Michael Steinbach dan Vipin Kumar. 2004. *Introducing to data Mining*. New York.

Yudho Giri Sucahyo, 2003, *Data mining*, publikasi ilmukomputer.com

LAMPIRAN

1. **Data Hasil Untuk Pengujian Tanpa Kombinasi dan Tanpa Pruning**

Jumlah Data	*Precision*	*Recall*	*Accuracy*	*Confidence*	Waktu (ms)
100	90.278 %	97.015 %	93.980 %	0.90582	1201
500	98.621 %	89.937 %	94.000 %	0.91470	1591
1000	88.276 %	96.970 %	93.000 %	0.90301	1154
1500	88.276 %	96.970 %	93.000 %	0.88264	1201
2000	98.621 %	95.333 %	97.000 %	0.93073	1497

2. **Data Hasil Untuk Pengujian Tanpa Kombinasi dan dengan Pruning**

Jumlah Data	*Precision*	*Recall*	*Accuracy*	*Confidence*	Waktu (ms)
100	75.000 %	92.308 %	84.950 %	0.80633	577
500	75.172 %	92.373 %	85.000 %	0.82252	420
1000	51.034 %	97.368 %	75.667 %	0.75710	405
1500	51.034 %	97.368 %	75.667 %	0.74811	421
2000	55.172 %	94.118 %	76.667 %	0.74435	515

3. **Data Hasil Untuk Pengujian dengan Kombinasi dan Tanpa Pruning**

Kombinasi atribut		*Precision*	*Recall*	*Accuracy*	*Confidence*	Waktu (ms)
1	pendapatan.	55,172%	94,12%	76,67%	0,74829	389
1	pekerjaan_kk.	32,414%	95,92%	66,67%	0,66048	405
1	jenis_lantai.	68,966%	67,11%	68,67%	0,71505	390
1	daya_listrik.	83,448%	63,68%	69,00%	0,67133	390
1	jenis_dinding.	43,448%	96,92%	72,00%	0,71322	406
1	sumber_air.	51,034%	97,37%	75,67%	0,72965	405
1	nilai_pbb.	75,172%	92,37%	85,00%	0,77239	405
2	jenis_lantai, daya_listrik.	68,966%	67,11%	68,67%	0,70985	764

2	pekerjaan_kk, daya_listrik.	81,379%	69,01%	73,33%	0,73798	764
2	sumber_air, daya_listrik.	87,586%	67,20%	73,33%	0,74065	764
2	jenis_lantai, pekerjaan_kk.	54,483%	88,76%	74,67%	0,73959	779
2	jenis_dinding, daya_listrik.	73,793%	76,43%	76,33%	0,72422	764
2	nilai_pbb, pendapatan.	55,172%	94,12%	76,67%	0,74829	405
2	jenis_lantai, jenis_dinding.	57,241%	97,65%	78,67%	0,76495	780
2	daya_listrik, pendapatan.	91,034%	72,93%	79,33%	0,78354	780
2	jenis_dinding, pendapatan.	62,759%	93,81%	80,00%	0,77992	764
2	pekerjaan_kk, jenis_dinding.	61,379%	96,74%	80,33%	0,79422	780
2	jenis_dinding, sumber_air.	62,759%	96,81%	81,00%	0,78022	779
2	pekerjaan_kk, sumber_air.	74,483%	85,71%	81,67%	0,79961	779
2	jenis_lantai, pendapatan.	66,207%	95,05%	82,00%	0,80023	780
2	sumber_air, pendapatan.	70,345%	94,44%	83,67%	0,79801	764
2	pekerjaan_kk, pendapatan.	71,724%	95,41%	84,67%	0,82462	780
2	jenis_lantai, nilai_pbb.	75,172%	92,37%	85,00%	0,76365	780
2	nilai_pbb, daya_listrik.	75,172%	92,37%	85,00%	0,77552	795
2	pekerjaan_kk, nilai_pbb.	75,172%	92,37%	85,00%	0,79166	779
2	jenis_lantai, sumber_air.	73,793%	95,54%	85,67%	0,80022	779
2	nilai_pbb, sumber_air.	77,931%	92,62%	86,33%	0,78786	764
2	nilai_pbb, jenis_dinding.	84,828%	92,48%	89,33%	0,83081	764
3	jenis_lantai, pekerjaan_kk,	73,793%	75,35%	75,67%	0,75919	1170

	daya_listrik.					
3	nilai_pbb, daya_listrik, pendapatan.	91,034%	72,93%	79,33%	0,78387	764
3	pekerjaan_kk, jenis_dinding, daya_listrik.	61,379%	96,74%	80,33%	0,79319	905
3	jenis_lantai, jenis_dinding, daya_listrik.	71,034%	86,56%	80,67%	0,76329	1076
3	jenis_dinding, sumber_air, daya_listrik.	62,759%	96,81%	81,00%	0,7842	1061
3	jenis_lantai, nilai_pbb, daya_listrik.	86,897%	77,30%	81,33%	0,80745	1185
3	jenis_lantai, pekerjaan_kk, jenis_dinding.	64,138%	96,88%	81,67%	0,79943	1201
3	jenis_lantai, nilai_pbb, pendapatan.	66,207%	95,05%	82,00%	0,80023	779
3	jenis_dinding, daya_listrik, pendapatan.	79,310%	83,94%	82,67%	0,8367	1169
3	jenis_lantai, daya_listrik, pendapatan.	80,000%	84,67%	83,33%	0,80346	1076
3	pekerjaan_kk, nilai_pbb, sumber_air.	78,621%	86,36%	83,67%	0,82446	1060
3	pekerjaan_kk, sumber_air, daya_listrik.	71,034%	94,50%	84,00%	0,81819	1045
3	jenis_lantai, pekerjaan_kk, pendapatan.	71,724%	95,41%	84,67%	0,82394	1170
3	pekerjaan_kk, daya_listrik, pendapatan.	71,724%	95,41%	84,67%	0,82844	1045
3	pekerjaan_kk, jenis_dinding,	71,724%	95,41%	84,67%	0,82352	858

	pendapatan.					
3	pekerjaan_kk, nilai_pbb, pendapatan.	71,724%	95,41%	84,67%	0,82557	779
3	jenis_lantai, pekerjaan_kk, nilai_pbb.	75,172%	92,37%	85,00%	0,78125	1169
3	pekerjaan_kk, nilai_pbb, daya_listrik.	75,172%	92,37%	85,00%	0,79098	1014
3	pekerjaan_kk, nilai_pbb, jenis_dinding.	73,103%	94,64%	85,00%	0,8614	1185
3	jenis_lantai, jenis_dinding, pendapatan.	73,793%	94,69%	85,33%	0,82754	1169
3	jenis_lantai, nilai_pbb, sumber_air.	73,793%	95,54%	85,67%	0,80022	780
3	jenis_lantai, sumber_air, daya_listrik.	73,793%	95,54%	85,67%	0,80226	967
3	nilai_pbb, sumber_air, daya_listrik.	77,931%	92,62%	86,33%	0,78801	811
3	jenis_lantai, pekerjaan_kk, sumber_air.	73,793%	98,17%	86,67%	0,81273	967
3	sumber_air, daya_listrik, pendapatan.	80,000%	91,34%	86,67%	0,8284	1170
3	jenis_dinding, sumber_air, pendapatan.	80,000%	94,31%	88,00%	0,83924	1216
3	jenis_lantai, nilai_pbb, jenis_dinding.	79,310%	96,64%	88,67%	0,85163	1185
3	nilai_pbb, jenis_dinding, pendapatan.	82,759%	93,02%	88,67%	0,81417	764
3	nilai_pbb, sumber_air,	82,759%	93,02%	88,67%	0,81236	764

	pendapatan.					
3	pekerjaan_kk, jenis_dinding, sumber_air.	83,448%	92,37%	88,67%	0,86455	1076
3	nilai_pbb, jenis_dinding, daya_listrik.	84,828%	92,48%	89,33%	0,84602	1169
3	jenis_lantai, jenis_dinding, sumber_air.	84,138%	95,31%	90,33%	0,85459	1200
3	nilai_pbb, jenis_dinding, sumber_air.	87,586%	92,70%	90,67%	0,83677	1076
3	pekerjaan_kk, sumber_air, pendapatan.	91,034%	93,62%	92,67%	0,87785	1061
3	jenis_lantai, sumber_air, pendapatan.	91,034%	95,65%	93,67%	0,88465	1091
4	jenis_lantai, pekerjaan_kk, nilai_pbb, daya_listrik.	84,828%	79,87%	82,33%	0,81446	1451
4	jenis_lantai, jenis_dinding, daya_listrik, pendapatan.	79,310%	83,94%	82,67%	0,83553	1185
4	jenis_lantai, pekerjaan_kk, jenis_dinding, daya_listrik.	66,207%	96,97%	82,67%	0,79794	1372
4	nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	79,310%	83,94%	82,67%	0,83758	1232
4	jenis_lantai, nilai_pbb, daya_listrik, pendapatan.	80,000%	84,67%	83,33%	0,8031	1186
4	jenis_lantai, pekerjaan_kk, daya_listrik,	71,724%	95,41%	84,67%	0,82523	1170

	pendapatan.					
4	pekerjaan_kk, nilai_pbb, jenis_dinding, pendapatan.	71,724%	95,41%	84,67%	0,82352	842
4	jenis_dinding, sumber_air, daya_listrik, pendapatan.	87,586%	82,47%	85,00%	0,84915	1232
4	pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik.	73,793%	94,69%	85,33%	0,86582	1481
4	jenis_lantai, nilai_pbb, sumber_air, daya_listrik.	73,793%	95,54%	85,67%	0,81229	967
4	pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik.	75,172%	94,78%	86,00%	0,82417	1076
4	jenis_lantai, pekerjaan_kk, nilai_pbb, pendapatan.	75,862%	95,65%	86,67%	0,83296	1201
4	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air.	73,793%	98,16%	86,67%	0,81976	967
4	jenis_lantai, pekerjaan_kk, sumber_air, daya_listrik.	73,793%	98,16%	86,67%	0,80963	951
4	nilai_pbb, sumber_air, daya_listrik, pendapatan.	80,000%	91,34%	86,67%	0,82826	1373
4	pekerjaan_kk, nilai_pbb, daya_listrik, pendapatan.	75,862%	95,65%	86,67%	0,83846	1061
4	jenis_lantai,	88,276%	85,91%	87,33%	0,83511	1201

	sumber_air, daya_listrik, pendapatan.					
4	jenis_lantai, nilai_pbb, jenis_dinding, daya_listrik.	79,310%	96,64%	88,67%	0,86479	1388
4	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding.	81,379%	94,40%	88,67%	0,865	1622
4	pekerjaan_kk, jenis_dinding, sumber_air, daya_listrik.	83,448%	92,37%	88,67%	0,86384	1092
4	jenis_lantai, pekerjaan_kk, jenis_dinding, pendapatan.	83,448%	93,08%	89,00%	0,85495	1216
4	pekerjaan_kk, jenis_dinding, daya_listrik, pendapatan.	83,448%	93,08%	89,00%	0,85013	1185
4	pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air.	84,828%	92,48%	89,33%	0,86509	1123
4	jenis_lantai, jenis_dinding, sumber_air, daya_listrik.	84,138%	95,31%	90,33%	0,8536	1185
4	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air.	84,138%	95,31%	90,33%	0,86338	1185
4	nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	87,586%	92,70%	90,67%	0,83677	1076
4	jenis_lantai, nilai_pbb, jenis_dinding, pendapatan.	86,897%	94,74%	91,33%	0,85978	1170

4	jenis_lantai, pekerjaan_kk, jenis_dinding, sumber_air.	84,138%	97,60%	91,33%	0,86341	1201
4	nilai_pbb, jenis_dinding, sumber_air, pendapatan.	90,345%	92,91%	92,00%	0,86791	1232
4	pekerjaan_kk, jenis_dinding, sumber_air, pendapatan.	91,034%	93,62%	92,67%	0,87497	1201
4	pekerjaan_kk, nilai_pbb, sumber_air, pendapatan.	91,034%	93,62%	92,67%	0,87998	1170
4	pekerjaan_kk, sumber_air, daya_listrik, pendapatan.	91,034%	93,62%	92,67%	0,87785	1076
4	jenis_lantai, nilai_pbb, sumber_air, pendapatan.	91,034%	95,65%	93,67%	0,88463	1076
4	jenis_lantai, pekerjaan_kk, sumber_air, pendapatan.	91,034%	95,65%	93,67%	0,89021	1263
4	jenis_lantai, jenis_dinding, sumber_air, pendapatan.	94,483%	95,14%	95,00%	0,90113	1185
5	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik.	73,793%	98,165%	86,667%	0,8197	917
5	jenis_lantai, pekerjaan_kk, nilai_pbb, daya_listrik, pendapatan.	75,862%	95,652%	86,667%	0,8347	1167

5	jenis_lantai, nilai_pbb, sumber_air, daya_listrik, pendapatan.	88,276%	85,906%	87,333%	0,8354	1712
5	jenis_lantai, nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	79,310%	83,942%	82,667%	0,8356	1276
5	nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	87,586%	82,468%	85,000%	0,8500	1821
5	jenis_lantai, pekerjaan_kk, jenis_dinding, daya_listrik, pendapatan.	83,448%	93,077%	89,000%	0,8502	1884
5	jenis_lantai, pekerjaan_kk, jenis_dinding, sumber_air, daya_listrik.	84,138%	97,600%	91,333%	0,8604	1884
5	pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	83,448%	95,276%	90,000%	0,8616	1869
5	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	84,138%	95,313%	90,333%	0,8636	1852
5	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, pendapatan.	83,448%	95,276%	90,000%	0,8641	1167
5	pekerjaan_kk, nilai_pbb, jenis_dinding,	84,828%	92,481%	89,333%	0,8651	1666

	sumber_air, daya_listrik.					
5	jenis_lantai, jenis_dinding, sumber_air, daya_listrik, pendapatan.	87,586%	85,235%	86,667%	0,8670	1868
5	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air.	84,138%	97,600%	91,333%	0,8704	1197
5	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik.	81,379%	94,400%	88,667%	0,8767	1760
5	pekerjaan_kk, jenis_dinding, sumber_air, daya_listrik, pendapatan.	91,034%	93,617%	92,667%	0,8782	1431
5	pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	91,034%	93,617%	92,667%	0,8793	1899
5	pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik, pendapatan.	91,034%	93,617%	92,667%	0,8864	2227
5	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, pendapatan.	91,034%	95,652%	93,667%	0,8889	1369
5	jenis_lantai, pekerjaan_kk, jenis_dinding, sumber_air, pendapatan.	91,034%	95,652%	93,667%	0,8899	1978

5	jenis_lantai, pekerjaan_kk, sumber_air, daya_listrik, pendapatan.	91,034%	95,652%	93,667%	0,8901	1962
5	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	94,483%	95,139%	95,000%	0,9012	1603
6	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	87,586%	85,24%	86,67%	0,86701	1186
6	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	83,448%	95,28%	90,00%	0,85938	1217
6	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	84,138%	97,60%	91,33%	0,87047	1373
6	jenis_lantai, pekerjaan_kk, jenis_dinding, sumber_air, daya_listrik, pendapatan.	91,034%	95,65%	93,67%	0,8899	1279
6	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik, pendapatan.	91,034%	95,65%	93,67%	0,88976	1466
6	pekerjaan_kk, nilai_pbb, jenis_dinding,	98,620%	93,46%	96,00%	0,92182	1528

	sumber_air, daya_listrik, pendapatan.					
6	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	98,621%	95,33%	97,00%	0,93073	1512
7	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	98,621%	95,33%	97,00%	0,93073	1513

4. Data Hasil Untuk Pengujian dengan Kombinasi dan dengan Pruning

Kombinasi atribut		***Precision***	***Recall***	***Accuracy***	***Confidence***	**Waktu (ms)**
1	pekerjaan_kk.	32,414%	95,918%	66,667%	0,66048	405
1	jenis_lantai.	68,966%	67,114%	68,667%	0,71505	420
1	daya_listrik.	83,448%	63,684%	69,000%	0,67133	437
1	jenis_dinding.	43,448%	96,923%	72,000%	0,71322	436
1	sumber_air.	51,034%	97,367%	75,667%	0,72965	421
1	pendapatan.	55,172%	94,118%	76,667%	0,74829	421
1	nilai_pbb.	75,172%	92,373%	84,950%	0,77239	436
2	pekerjaan_kk, daya_listrik.	32,414%	95,918%	66,667%	0,66052	421
2	jenis_lantai, daya_listrik.	68,966%	67,114%	68,667%	0,71505	421
2	jenis_lantai, jenis_dinding.	68,966%	67,114%	68,667%	0,71505	406
2	jenis_lantai, pekerjaan_kk.	68,966%	67,114%	68,667%	0,71505	420
2	jenis_dinding,	43,448%	96,923%	72,000%	0,71322	405

	daya_listrik.					
2	pekerjaan_kk, jenis_dinding.	43,448%	96,923%	72,000%	0,71322	406
2	jenis_dinding, sumber_air.	51,034%	97,367%	75,667%	0,72965	421
2	jenis_lantai, sumber_air.	51,034%	97,367%	75,667%	0,72965	436
2	nilai_pbb, sumber_air.	51,034%	97,367%	75,667%	0,73	545
2	pekerjaan_kk, sumber_air.	51,034%	97,367%	75,667%	0,72965	405
2	sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	421
2	daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	421
2	jenis_dinding, pendapatan.	55,172%	94,118%	76,667%	0,74775	483
2	jenis_lantai, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
2	nilai_pbb, pendapatan.	55,172%	94,118%	76,667%	0,74829	436
2	pekerjaan_kk, pendapatan.	55,172%	94,118%	76,667%	0,74829	436
2	sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74829	436
2	jenis_lantai, nilai_pbb.	75,172%	92,373%	84,950%	0,77239	437
2	nilai_pbb, daya_listrik.	75,172%	92,373%	84,950%	0,77239	421
2	nilai_pbb, jenis_dinding.	75,172%	92,373%	84,950%	0,77239	421
2	pekerjaan_kk, nilai_pbb.	75,172%	92,373%	84,950%	0,77239	406
3	jenis_lantai, jenis_dinding, daya_listrik.	68,966%	67,114%	68,667%	0,71505	421
3	jenis_lantai, pekerjaan_kk, daya_listrik.	68,966%	67,114%	68,667%	0,71898	467
3	jenis_lantai, pekerjaan_kk, jenis_dinding.	68,966%	67,114%	68,667%	0,71505	405

3	pekerjaan_kk, jenis_dinding, daya_listrik.	43,448%	96,923%	72,000%	0,71322	405
3	jenis_dinding, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	406
3	jenis_lantai, jenis_dinding, sumber_air.	51,034%	97,367%	75,667%	0,72965	405
3	jenis_lantai, nilai_pbb, sumber_air.	51,034%	97,367%	75,667%	0,72965	406
3	jenis_lantai, pekerjaan_kk, sumber_air.	51,034%	97,367%	75,667%	0,72965	453
3	jenis_lantai, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	421
3	nilai_pbb, jenis_dinding, sumber_air.	51,034%	97,367%	75,667%	0,72965	406
3	nilai_pbb, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,73	546
3	pekerjaan_kk, jenis_dinding, sumber_air.	51,034%	97,367%	75,667%	0,72965	405
3	pekerjaan_kk, nilai_pbb, sumber_air.	51,034%	97,367%	75,667%	0,73	514
3	pekerjaan_kk, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	405
3	jenis_dinding, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
3	jenis_dinding, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74292	577
3	jenis_lantai, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	390

3	jenis_lantai, jenis_dinding, pendapatan.	55,172%	94,118%	76,667%	0,74829	406
3	jenis_lantai, nilai_pbb, pendapatan.	55,172%	94,118%	76,667%	0,74829	390
3	jenis_lantai, pekerjaan_kk, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
3	jenis_lantai, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74435	514
3	nilai_pbb, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74861	545
3	nilai_pbb, jenis_dinding, pendapatan.	55,172%	94,118%	76,667%	0,74775	483
3	nilai_pbb, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74861	514
3	pekerjaan_kk, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
3	pekerjaan_kk, jenis_dinding, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
3	pekerjaan_kk, nilai_pbb, pendapatan.	55,172%	94,118%	76,667%	0,74861	499
3	pekerjaan_kk, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74426	499
3	sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
3	jenis_lantai, nilai_pbb, jenis_dinding.	75,172%	92,373%	84,950%	0,77239	420
3	jenis_lantai, pekerjaan_kk, nilai_pbb.	75,172%	92,373%	84,950%	0,77239	406

3	nilai_pbb, jenis_dinding, daya_listrik.	75,172%	92,373%	84,950%	0,77239	390
3	pekerjaan_kk, nilai_pbb, daya_listrik.	75,172%	92,373%	84,950%	0,77239	420
3	pekerjaan_kk, nilai_pbb, jenis_dinding.	75,172%	92,373%	84,950%	0,77239	405
3	jenis_lantai, nilai_pbb, daya_listrik.	75,172%	92,373%	85,000%	0,77239	405
4	jenis_lantai, pekerjaan_kk, jenis_dinding, daya_listrik.	68,966%	67,114%	68,667%	0,71505	436
4	jenis_lantai, jenis_dinding, sumber_air, daya_listrik.	51,034%	97,368%	75,667%	0,72965	388
4	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air.	51,034%	97,367%	75,667%	0,72965	405
4	jenis_lantai, nilai_pbb, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	390
4	jenis_lantai, pekerjaan_kk, jenis_dinding, sumber_air.	51,034%	97,367%	75,667%	0,72965	437
4	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air.	51,034%	97,367%	75,667%	0,72965	420
4	jenis_lantai, pekerjaan_kk, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	437
4	nilai_pbb, jenis_dinding, sumber_air,	51,034%	97,367%	75,667%	0,72965	421

	daya_listrik.					
4	pekerjaan_kk, jenis_dinding, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	436
4	pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air.	51,034%	97,367%	75,667%	0,72965	406
4	pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,73	498
4	jenis_dinding, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	390
4	jenis_lantai, jenis_dinding, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	390
4	jenis_lantai, jenis_dinding, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74435	499
4	jenis_lantai, nilai_pbb, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
4	jenis_lantai, nilai_pbb, jenis_dinding, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
4	jenis_lantai, nilai_pbb, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74435	530
4	jenis_lantai, pekerjaan_kk, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
4	jenis_lantai,	55,172%	94,118%	76,667%	0,74829	530

	pekerjaan_kk, jenis_dinding, pendapatan.					
4	jenis_lantai, pekerjaan_kk, nilai_pbb, pendapatan.	55,172%	94,118%	76,667%	0,74829	624
4	jenis_lantai, pekerjaan_kk, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74435	499
4	jenis_lantai, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74435	498
4	nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	406
4	nilai_pbb, jenis_dinding, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74292	545
4	nilai_pbb, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74861	499
4	pekerjaan_kk, jenis_dinding, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
4	pekerjaan_kk, jenis_dinding, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74426	530
4	pekerjaan_kk, nilai_pbb, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74861	499
4	pekerjaan_kk, nilai_pbb, jenis_dinding, pendapatan.	55,172%	94,118%	76,667%	0,74829	436

4	pekerjaan_kk, nilai_pbb, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74458	623
4	pekerjaan_kk, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74426	515
4	jenis_lantai, nilai_pbb, jenis_dinding, daya_listrik.	75,172%	92,373%	84,950%	0,77239	405
4	jenis_lantai, pekerjaan_kk, nilai_pbb, daya_listrik.	75,172%	92,373%	84,950%	0,77239	405
4	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding.	75,172%	92,373%	84,950%	0,77239	436
4	pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik.	75,172%	92,373%	84,950%	0,77239	421
5	jenis_lantai, pekerjaan_kk, jenis_dinding, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	389
5	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	405
5	jenis_lantai, pekerjaan_kk, jenis_dinding, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	405
5	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding,	51,034%	97,367%	75,667%	0,72965	405

	sumber_air.					
5	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	405
5	pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	405
5	jenis_lantai, jenis_dinding, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74435	499
5	jenis_lantai, nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	406
5	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74435	514
5	jenis_lantai, nilai_pbb, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74435	515
5	jenis_lantai, pekerjaan_kk, jenis_dinding, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74435	514
5	jenis_lantai, pekerjaan_kk, nilai_pbb, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	406
5	jenis_lantai,	55,172%	94,118%	76,667%	0,74829	421

	pekerjaan_kk, nilai_pbb, jenis_dinding, pendapatan.					
5	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74435	530
5	jenis_lantai, pekerjaan_kk, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74435	514
5	nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
5	pekerjaan_kk, jenis_dinding, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74422	530
5	pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	406
5	pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74426	530
5	pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74455	640
5	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding,	75,172%	92,373%	84,950%	0,77239	405

	daya_listrik.					
6	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik.	51,034%	97,367%	75,667%	0,72965	405
6	jenis_lantai, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74435	514
6	jenis_lantai, pekerjaan_kk, jenis_dinding, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74435	498
6	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74829	405
6	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, pendapatan.	55,172%	94,118%	76,667%	0,74435	545
6	jenis_lantai, pekerjaan_kk, nilai_pbb, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74435	530
6	pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik,	55,172%	94,118%	76,667%	0,74422	530

	pendapatan.					
7	jenis_lantai, pekerjaan_kk, nilai_pbb, jenis_dinding, sumber_air, daya_listrik, pendapatan.	55,172%	94,118%	76,667%	0,74435	514