Aplikasi Datamining
Menggunakan Algoritma ID3 Dengan J2SE

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika



Oleh:

Nama : Dimas Sirin Madefanny
No. Mahasiswa : 02 523 037

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2007

LEMBAR PENGESAHAN PEMBIMBING

Aplikasi Datamining

Menggunakan Algoritma ID3 Dengan J2SE

TUGAS AKHIR

oleh:

Nama : Dimas Sirin Madefanny

No. Mahasiswa : 02 523 037

Yogyakarta, 2007

Telah diterima dan Disetujui dengan baik oleh:

Dosen Pembimbing

Taufiq Hidayat, ST., MCS.

LEMBAR PENGESAHAN DOSEN PENGUJI

Aplikasi Datamining
Menggunakan Algoritma ID3 Dengan J2SE

TUGAS AKHIR

oleh:

Nama : Dimas Sirin Madefanny

No. Mahasiswa : 02 523 037

Telah Dipertahankan di Depan Sidang Penguji sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika
Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 2007

Tim Penguji

Taufiq Hidayat, ST., MCS. _____
Ketua

Lizda Iswari, ST. _____
Anggota I

Yudi Prayudi, Ssi., M.Kom. _____
Anggota II

Mengetahui,
Ketua Jurusan Teknik Informatika
Universitas Islam Indonesia

Yudi Prayudi, Ssi., M.Kom

LEMBAR PERNYATAAN KEASLIAN HASIL TUGAS AKHIR

Saya yang bertanda tangan di bawah ini,

Nama : Dimas Sirin Madefanny

No. Mahasiswa : 02 523 037

Menyatakan bahwa seluruh komponen dan isi dalam Tugas Akhir ini adalah hasil karya saya sendiri.

Apabila di kemudian hari terbukti bahwa ada beberapa bagian dari karya ini adalah bukan hasil karya saya sendiri, maka saya siap menanggung resiko dan konsekuensi apapun.

Demikian pernyataan ini saya buat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 2007

(Dimas Sirin Madefanny)

Dedicated to

Allah the great

Alhamdulillah

Karya ku untuk papa mama ku yang tercinta

Untuk kakak serta adikku yang kusayang

Terima kasih atas doa serta dukungan yang kalian berikan

Terima kasih kepada teman-temanku

Semua

Yang telah mendukung segala usaha ku

VOIPer's

Jalan kita masih panjang

Semoga kita bisa bertemu lagi

Untuk footballers and narutofans

Tingkatkan kemampuan jangan pernah menyerah

SEMANGAT!!!!!

KATA PENGANTAR

بسم الله الرحمن الرحيم

Assalamu'alaikum Warahmatullahi Wabarakatuh

Segala puji dan syukur hanyalah milik Allah Subhanahu Wa Ta'ala yang Maha Pengasih dan Maha Penyayang, yang telah menjadikan manusia beriman dan berilmu setelah sebelumnya berada dalam kondisi yang lemah dan diliputi kebodohan. Atas izin dan kehendak-Nya, tugas akhir berjudul **"Aplikasi Datamining Menggunakan Algoritma ID3 Dengan J2SE "** ini akhirnya dapat terselesaikan. Tugas Akhir ini merupakan salah satu syarat untuk memperoleh gelar sarjana Teknik Informatika di Universitas Islam Indonesia.

Dalam penulisan tugas akhir ini tidak lepas dari saran-saran, bimbingan, dukungan, serta bantuan dari berbagai pihak. Untuk itu, pada kesempatan ini penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Kedua orang tuaku, yang tak pernah putus memberikan dorongan, semangat, dan doanya.
2. Kakak serta adikku yang selalu memberi semangat di saat susah maupun senang.

3. Bapak Fathul Wahid, ST., M.Sc., selaku Dekan Fakultas Teknologi Industri, Universitas Islam Indonesia.
4. Bapak Yudi Prayudi, S.Si., M.Kom., selaku Ketua Jurusan Teknik Informatika, Universitas Islam Indonesia.
5. Bapak Taufiq Hidayat, ST., MCS. selaku Dosen Pembimbing, yang telah memberikan bimbingan dalam penyusunan laporan tugas akhir ini.
6. Seluruh rekan-rekan Informatika UII angkatan 2002, terutama yang telah banyak membantu.
7. Anak-anak KKN Unit 45 angkatan 33.
8. Semua Pihak yang tidak dapat saya sebutkan satu persatu.

Dalam penyelesaian Tugas Akhir ini penulis menyadari bahwa masih banyak terdapat kelemahan dan kekurangan. Oleh karena itu penulis mengharapkan kritik dan saran yang bersifat membangun agar pada masa mendatang menjadi lebih baik.

Akhir kata, Semoga Tugas Akhir ini dapat berguna bagi para penuntut ilmu, para praktisi, dan seluruh masyarakat IT untuk tujuan kemaslahatan dan kepentingan bersama.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Yogyakarta, 2007

Penyusun

SARI

Teknologi komputasi dan media penyimpanan telah memungkinkan manusia untuk mengumpulkan dan menyimpan data dari berbagai sumber dengan jangkauan yang amat luas. Kumpulan dari data yang sangat banyak dengan rentang waktu yang lama menimbulkan *data explosion*, oleh karena itu *datamining* dibutuhkan untuk mendapatkan atau mengekstrak informasi ataupun pola yang penting dari data yang sangat banyak tersebut.

Aplikasi datamining dengan J2SE adalah aplikasi yang menggunakan teknik datamining yaitu algoritma ID3 yang berfungsi untuk memudahkan pengguna dalam mengambil informasi dari jumlah data yang banyak. Dimudahkan dalam arti sistem dapat menampilkan informasi atau pola dari data yang diperoleh ke dalam bentuk yang lebih mudah dipahami atau dimengerti.

Hasil yang didapatkan dari aplikasi pembelajaran mesin ini adalah pohon keputusan. Pohon keputusan/*decision tree* yang dihasilkan dapat digunakan dalam pengambilan keputusan.

Kata Kunci: *Data explosion, datamining, decision tree*

TAKARIR

Datamining	Penggalian informasi dari data
Knowledge discovery	Penemuan pengetahuan
Classification	Klasifikasi
Decision tree	Pohon keputusan
Data explosion	Ledakan data
Database	Basis data
Interface	Tampilan/antarmuka
Algoritma ID3	Algoritma yang digunakan untuk menghasilkan sebuah decision tree
Data warehouse	Gudang data
Root node	Titik awal dari suatu pohon keputusan
Leaf node	Titik cabang pada pohon keputusan
Entropy	Nilai informasi dari suatu data
Information gain	Ukuran yang digunakan untuk membuat decision tree dengan memilih nilai atribut yang terbaik

DAFTAR ISI

HALAMAN JUDUL i
LEMBAR PENGESAHAN PEMBIMBING ii
LEMBAR PENGESAHAN DOSEN PENGUJI iii
LEMBAR PERNYATAAN KEASLIAN HASIL TUGAS AKHIR iv
HALAMAN PERSEMBAHAN v
KATA PENGANTAR vi
SARI viii
TAKARIR ix
DAFTAR ISI x
DAFTAR TABEL xiii
DAFTAR GAMBAR xiv

BAB I PENDAHULUAN 1
1.1 Latar Belakang Masalah 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan Penelitian 3
1.5 Manfaat Penelitian 3
1.6 Metodologi Penelitian 3
1.7 Sistematika Penelitian 5
BAB II LANDASAN TEORI 7
2.1 Datamining 7
2.1.1 Artificial Neural Networks 12
2.1.2 Genetics Algorithm 13
2.1.3 Decision tree 14
2.1.4 Nearest Neighbour Method 20

2.1.5	Rule Induction	21
2.2	Sekilas Tentang Java	21
2.2.1	J2SE(Java™ Platform, Standard Edition)	24
2.2.1.1	J2SE Runtime Environment (JRE)	24
2.2.1.2	J2SE Development Kit (JDK)	25
2.2.2	J2ME(Java™ Platform, Micro Edition)	25
2.2.3	J2EE(Java™ Platform, Enterprice Edition)	26
2.3	UML	27
BAB III	METODOLOGI	33
3.1	Analisis Kebutuhan	33
3.1.1	Metode Analisis	33
3.1.2	Hasil Analisis	34
3.1.2.1	Kebutuhan Masukan	34
3.1.2.2	Kebutuhan Proses	36
3.1.2.3	Kebutuhan Keluaran Sistem	38
3.1.2.4	Kebutuhan Perangkat Lunak	40
3.1.2.5	Kebutuhan Perangkat Keras	41
3.1.2.6	Kebutuhan Antarmuka	41
3.2	Perancangan Sistem	41
3.2.1	Metode Perancangan	41
3.2.2	Hasil Perancangan	42
3.2.3	Perancangan Unified Modelling Language (UML) dan Lainnya	42
3.2.3.1	Use Case Diagram	42
3.2.3.2	Class Diagram	44
3.2.3.3	Sequence Diagram	46
3.2.3.4	Perancangan Antarmuka	51
BAB IV	HASIL DAN PEMBAHASAN	57
4.1	Hasil	57

4.1.1	Implementasi Sistem Datamining Secara Umum	57
4.1.2	Implementasi Struktur Sistem Datamining	57
4.1.3	Implementasi Antarmuka Aplikasi Datamining	58
4.1.4	Implementasi Prosedural	62
4.2	Pembahasan	67
BAB V	SIMPULAN DAN SARAN	72
5.1	Simpulan	72
5.2	Saran	73
DAFTAR PUSTAKA		74
Lampiran		75

DAFTAR TABEL

Tabel 3.1 tabel data masukan "play ball" 33
Tabel 3.2 attribute dan nilai kemungkinannya 34

DAFTAR GAMBAR

Gambar 2.1 gambar sebuah decision tree 15
Gambar 2.2 Platform Java 2 Standard Edition (J2SE) 25
Gambar 2.3 Platform Java 2 Micro Edition (J2ME) 26
Gambar 2.2 Platform Java 2 Enterprise Edition (J2EE) 27
Gambar 3.1 Decision tree aplikasi pembelajaran mesin 39
Gambar 3.2 Use case diagram pada pembelajaran mesin 44
Gambar 3.3 Class Diagram aplikasi pembelajaran mesin 46
Gambar 3.4 Sequence Diagram Input Table 48
Gambar 3.5 Sequence Diagram clear table 48
Gambar 3.6 Sequence Diagram Process data 49
Gambar 3.7 Sequence Diagram Create tree 50
Gambar 3.8 Sequence Diagram clear tree 51
Gambar 3.9 Rancangan antarmuka tampilan utama 52
Gambar 3.10 Rancangan antarmuka pemilihan file 54
Gambar 3.11 Rancangan antarmuka pesan kesalahan 55
Gambar 3.12 Rancangan antarmuka menu About 56
Gambar 4.1 Tampilan Utama 59
Gambar 4.2 Pemilihan file 60
Gambar 4.3 Tampilan pengenalan/Introduction 61
Gambar 4.4 Tampilan Help/Bantuan 61
Gambar 4.5 Tampilan About 62
Gambar 4.6 Tampilan pemilihan file 68
Gambar 4.7 Tampilan penampilan data 69
Gambar 4.8 Tampilan kesalahan pesan 70
Gambar 4.9 Tampilan decision tree 70

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Teknologi komputasi dan media penyimpanan telah memungkinkan manusia untuk mengumpulkan dan menyimpan data dari berbagai sumber dengan jangkauan yang amat luas. Fenomena ini terjadi dalam banyak bidang kehidupan, seperti bisnis, perbankan, pemasaran, produksi sains dan sebagainya. Meskipun teknologi basisdata modern telah menghasilkan media penyimpanan yang ekonomis bagi aliran data yang amat besar ini, teknologi untuk membantu menganalisis, memahami, atau bahkan memvisualisasikannya belumlah tersedia. Kehadiran *data mining* dilatar belakangi dengan problema *data explosion* yang dialami akhir-akhir ini dimana banyak organisasi telah mengumpulkan data sekian tahun lamanya seperti data pembelian, data penjualan, data nasabah, data transaksi dsb. Definisi sederhana dari *data mining* adalah ekstraksi informasi atau pola yang penting atau menarik dari data yang ada di *database* yang besar. Dalam jurnal ilmiah, *data mining* juga dikenal dengan nama *Knowledge Discovery in Databases* (KDD).

Untuk dapat mengambil pengetahuan dari database diperlukan tahap-tahap yang dilalui, antara lain adalah memilih algoritma datamining. Salah satu algoritma yang dapat digunakan untuk mengambil pengetahuan dari database adalah

algoritma ID3 yang dikembangkan oleh J. Ross Quinlan. Algoritma ID3 atau *Iterative Dichotomiser 3 algorithm* adalah algoritma yang digunakan untuk membuat sebuah *decision tree*. Algoritma ini membentuk *decision tree* dari sebuah data tetap yang berguna sebagai pembelajaran untuk mengklasifikasikan data dan memprekdisikan klasifikasi dari data baru. Dengan setiap data harus mempunyai beberapa atribut dengan nilai yang berbeda.

1.2 Rumusan Masalah

Rumusan masalah adalah bagaimana merancang dan mengimplementasikan suatu aplikasi perangkat lunak yang dapat menerjemahkan data ke dalam *decision tree* dan *decision tree* tersebut dapat digunakan untuk pengambilan keputusan.

1.3 Batasan Masalah

Pembatasan masalah disini bukan saja untuk menyederhanakan persoalan yang dihadapi, tetapi juga untuk menyederhakan persoalan tersebut agar tidak menyimpang dari yang diinginkan. Yang menjadi batasan-batasan penelitian tugas akhir ini adalah :

a. Aplikasi hanya berfokus pada satu algoritma yang digunakan sebagai penelitian yaitu Algoritma ID3.
b. Masukan dari aplikasi ini berupa satu tabel dengan satu kolom sebagai atribut target.
c. Atribut yang digunakan adalah yang bersifat diskret.

d. Keluaran dari aplikasi ini adalah sebuah decision tree.

e. Perangkat lunak yang akan digunakan untuk membangun aplikasi ini adalah Java 2 Standard Edition.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk membangun perangkat lunak yang:

a. Mampu untuk menerjemahkan tabel yang dimasukkan ke dalam *decision tree*.

b. *Decision tree* dapat digunakan untuk menentukan/mengambil keputusan.

1.5 Manfaat Penelitian

Dengan adanya penelitian diharapkan dapat memberikan manfaat antara lain:

a. Mempermudah dalam memperoleh informasi dari data yang dimiliki.

b. Mempermudah user dalam melihat data

c. Memberikan kemudahan user dalam menentukan keputusan.

d. Membantu dalam menentukan langkah-langkah atau keputusan yang akan datang.

1.6 Metodologi Penelitian

Metodologi penelitian adalah suatu cara berurutan yang dilakukan dalam penelitian. Metode yang digunakan untuk membantu dalam merancang Aplikasi Datamining Menggunakan Algoritma ID3 dengan J2SE ini adalah :

1. Metode Pengumpulan Data

Metode pengumpulan data merupakan cara pertama yang digunakan dalam melakukan penelitian. Metode pengumpulan data terdiri dari :

a) Metode Observasi

Metode observasi antara lain pengumpulan data dengan mengamati data-data apa saja yang perlu dimasukkan ke dalam aplikasi.

b) Metode Studi Pustaka

Metode studi pustaka terdiri dari pengumpulan data dan informasi dari buku-buku referensi, modul-modul dan artikel yang sesuai dengan penelitian.

2. Metode Perancangan Perangkat Lunak

a) Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak meliputi metode analisis yang digunakan dan analisis sistem yang terdiri dari analisis input, output, fungsi-fungsi yang dibutuhkan, kinerja yang harus dipenuhi, dan sistem antar muka yang diinginkan.

b) Perancangan

Tahap ini merupakan penerjemahan dari data yang telah dianalisis ke dalam bentuk yang mudah dimengerti oleh user, dengan membuat perancangan Aplikasi Proses Pembelajaran yang meliputi perancangan UML dan perancangan *interface* software.

c) Implementasi

Tahap ini dilakukan untuk menterjemahkan masalah yang telah dirancang ke dalam bahasa pemrograman yang telah ditentukan, yaitu dengan menggunakan J2SE.

d) Pengujian

Setelah program selesai dibuat, maka pada tahap ini merupakan uji coba terhadap program tersebut. Pengujian ini dapat dilakukan dengan menggunakan kondisi-kondisi yang berbeda untuk menciptakan suatu aplikasi yang bermanfaat.

1.7 Sistematika Penelitian

Untuk memberikan gambaran menyeluruh mengenai masalah yang akan dibahas, maka digunakan sistematika penulisan tugas akhir. Sistematika penulisan tugas akhir Aplikasi Proses Pembelajaran Menggunakan Algoritma ID3 adalah sebagai berikut :

BAB I PENDAHULUAN

Bab ini berisi latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan yang digunakan untuk merancang suatu aplikasi yang memberikan keputusan terhadap masukan yang diberikan.

BAB II LANDASAN TEORI

bab ini mengemukakan landasan teori yang digunakan dalam melakukan penelitian, yang meliputi teori-teori tentang datamining, algoritma ID3, JAVA, J2SE.

BAB III METODOLOGI

Memuat uraian tentang langkah-langkah dalam penelitian sesuai dengan judulnya. Yaitu analisis perangkat lunak, perancangan kebutuhan perangkat lunak yang akan dikembangkan dan diimplementasi dari perangkat lunak yang sudah dirancang dan dianalisis sebelumnya.

BAB IV HASIL DAN PEMBAHASAN

Berisi tentang bagaimana hasil dari aktivitas atau metodologi penelitian yang sudah disamapaikan pada bab sebelumnya dan dilakukannya pembahasan mengenai hasil yang sudah dicapai, diperoleh selama melakukan penelitian tugas akhir seperti mengapa hasil yang diperoleh, kelebihan, kekurangan dari hasil yang diperoleh.

BAB V SIMPULAN DAN SARAN

Berisi tentang uraian simpulan dan saran hasil analisis kerja yang telah dibuat pada bab sebelumnya yaitu hasil dan pembahasan.

BAB II

LANDASAN TEORI

2.1 Datamining

Secara umum datamining sering disebut juga *data discovery/knowledge discovery* (penggalian data/penemuan pengetahuan) adalah sebuah proses analisis data dari sudut pandang yang berbeda dan meringkasnya menjadi sebuah informasi yang berharga. Informasi yang dapat meningkatkan hasil(revenue), pemotongan biaya produksi(cuts costs), atau kedua-duanya. Software datamining adalah sebuah alat untuk menganalisis data. Software tersebut dapat digunakan user untuk menganalisis data dari berbagai macam dimensi atau sudut, mengkelompokkannya , meringkasnya ke aturan yang telah diidentifikasikan.

Walaupun datamining merupakan suatu hal yang baru, tetapi teknologi dari datamining bukanlah hal yang baru lagi. Berbagai macam perusahaan telah menggunakan *powerful*(super) komputer untuk menyelidiki scanner data supermarket yang besar dan menganalisis laporan riset pasar untuk bertahun-tahun. Walau bagaimanapun, inovasi terus menerus dalam kekuatan process komputer, tempat penyimpanan, dan software statistik secara dramatis meningkatkan akurasi dari analisis bersamaan dengan menurunkan biaya produksi.

Sebagai contoh, sebuah rantai grosir di Midwest menggunakan kapasitas dari software Oracle untuk menganalisa pola pembelian lokal disana. Mereka menemukan jika seorang pria akan membeli popok pada hari kamis dan sabtu, mereka juga cenderung untuk membeli bir juga. Untuk analisis yang lebih jauh, menunjukkan bahwa pembeli seperti ini melakukan belanja grosiran pada hari sabtu. Pada hari kamis, mereka hanya membeli beberapa barang saja. oleh karena itu, retailer menyimpulkan bahwa mereka akan menyediakan bir untuk weekend yang mendatang untuk disediakan bagi pembeli tersebut. Informasi baru yang telah ditemukan ini dengan berbagai cara dapat digunakan untuk meningkatkan pendapatan. Sebagai contoh, mereka dapat memindahkan bagian bir lebih dekat dengan bagian popok. Dan mereka dapat menjual bir dan popok dengan harga penuh pada hari kamis.

Data, informasi dan pengetahuan

Data

Data adalah fakta, angka-angka, atau text yang dapat diproses oleh komputer. Sekarang, perusahaan-perusahaan sedang mengakumulasi pertumbuhan data yang sangat banyak dengan berbagai format dan database yang berbeda pula. Ini termasuk:

1. Operasional atau transaksional data seperti penjualan, harga, inventaris, daftar gaji, dan akuntansi.

2. Nonoperational data seperti, seperti penjualan di bidang industri, data ramalan cuaca, dan data ekonomi makro.
3. Meta data, yaitu data tentang data itu sendiri, seperti design logika database atau definisi data dictionary.

Informasi

Pola, kumpulan atau hubungan antara semua data dapat memberikan informasi. Sebagai contoh, analisis nilai retail dari data transaksi penjualan dapat memberikan information terhadap produk mana yang akan dijual dan kapan.

Pengetahuan

Informasi dapat dirubah menjadi pengetahuan tentang pola historik dan trend yang akan datang. Sebagai contoh, ringkasan informasi pada penjualan eceran supermarket dapat dianalisis dengan sedikit upaya promosi untuk memberikan pengetahuan tentang kebiasaan konsumen dalam membeli barang. Denga demikian, produsen maupun pengecer dapat menentukan barang yang mana yang paling cocok untuk dipromosikan.

Data warehouse

Kemajuan yang dramatis dalam pengambilan data, kekuatan memproses, transmisi data, dan kemampuan menyimpan membuat perusahaan-perusahaan mampu untuk mengintegrasikan berbagai macam database ke dalam gudang data/*data warehouse*. *Data warehouse*/gudang data didefinisikan sebagai proses retrieval dan manajemen data yang terpusat. *Data warehousing*, seperti datamining,

merupakan suatu hal yang baru walaupun konsep tersebut sudah ada untuk beberapa tahun sebelumnya. *Data warehousing* merupakan gambaran yang ideal untuk pemeliharaan tempat penyimpanan utama dari semua data perusahaan. Pemusatan data sangat dibutuhkan untuk memaksimalkan analisis dan akses pengguna. Perkembangan dramatikal dari teknologi membuat gambaran ini menjadi nyata untuk perusahaan-perusahaan.

Datamining digunakan terutama oleh perusahaan-perusahaan dengan tingkat fokus kepada konsumen yang tinggi, seperti perusahaan retail, financial, komunikasi dan perusahaan marketing. Datamining membuat perusahaan mampu menentukan hubungan antara faktor-faktor internal seperti harga, positioning produk, atau skill staf-staf, dan faktor-faktor internal seperti indikator ekonomi, kompetisi, dan demografi konsumen. Perusahaan juga mampu menetapkan pengaruh dari penjualan, kepuasan pelanggan, dan keuntungan perusahaan. Akhirnya, mereka mampu untuk menggali ringkasan informasi untuk melihat data transaksional secara detail.

Dengan datamining pengecer dapat menggunakan rekaman hasil penjualan dari pembelian pelanggan untuk mengirim target promosi berdasarkan pada sejarah pembelian individual.

Cara kerja datamining

Seiring teknologi informasi skala besar berkembang memisalkan transaksi dan sistem analitik, datamining menyediakan hubungan antara keduanya. Software datamining menganalisis hubungan dan pola di data transaksi yang disimpan berdasarkan pada query open-ended user. Beberapa tipe dari software analitik tersedia: statistical, machine learning, dan neural networks/jaringan syaraf tiruan. Secara umum, salah satu dari empat tipe hubungan akan diminta:

a. **Classes:** data yang disimpan digunakan untuk menempatkan data pada kelompok-kelompok yang telah ditentukan.

b. **Clusters:** nomor data dikelompokkan berdasarkan hubungan logis atau pilihan konsumen. Misalnya, data dapat digali untuk untuk mengenali segmen pasar atau daya tarik konsumen.

c. **Associations** : data dapat digali untuk mengenali perkumpulan-perkumpulan. Contohnya, bir-popok adalah contoh assosiatif mining.

d. **Sequential patterns** : data digali untuk mengetahui pola kelakuan dan kecenderungan. Contoh, seorang pedagang eceran perlengkapan outdoor bisa meramalkan kemungkinan ransel dibeli berdasarkan pada pembelian konsumen terhadap sleeping bag dan sepatu hiking.

Datamining terdiri dari lima elemen penting:

a. Ambil, ubah, dan muatkan transaksi ke dalam sistem data warehouse.

b. Simpan dan atur data ke dalam sistem database multidimensional / *multidimensional database system*.

c. Sediakan akses data kepada analis perusahaan dan profesional teknologi informasi.

d. Analisis data sesuai aplikasi software.

e. Tampilkan data dalam format yang berguna, seperti grafik atau tabel.

2.1.1 Artificial neural networks

Neural networks atau sering disebut dengan jaringan syaraf merupakan salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut.

Komponen-komponen pada jaringan syaraf terdiri dari :

1. Neuron. Neuron-neuron akan mentranformasikan informasi yang diterima melalui sambungan keluarnya menuju ke neuron-neuron yang lain.
2. Bobot, adalah hubungan yang ada pada jaringan syaraf. Informasi-informasi yang ditransformasikan melalui neuron disimpan pada suatu nilai tertentu pada bobot tertentu.

Informasi (disebut dengan: input) akan dikirim ke neuron dengan bobot kedatangan tertentu. Input ini akan diproses oleh suatu fungsi perambatan yang akan menjumlahkan nilai-nilai semua bobot yang datang. Hasil penjumlahan ini kemudian akan dibandingkan dengan suatu nilai ambang (threshold) tertentu melalui fungsi aktivasi setiap neuron. Apabila, input tersebut melewati suatu nilai

ambang tertentu, maka neuron tersebut akan diaktifkan, tapi kalau tidak, maka neuron tersebut tidak akan diaktifkan. Apabila neuron tersebut diaktifkan, maka neuron tersebut akan mengirimkan output melalui bobot-bobot outputnya ke semua neuron yang berhubungan dengannya. Demikian seterusnya.

2.1.2 Genetics algorithm

Algoritma genetika didasarkan pada analogi pada evolusi biologi. Pada algoritma ini, teknik pencarian dilakukan sekaligus atas sejumlah solusi yang mungkin dikenal dengan istilah **populasi** 8. Individu yang terdapat dalam satu populasi disebut dengan istilah **kromosom**. Kromosom ini merupakan suatu solusi yang masih berbentuk simbol. Populasi awal dibangun secara acak, sedangkan populasi berikutnya merupakan hasil evolusi kromosom-kromosom melalui iterasi yang disebut dengan istilah **generasi**. Pada setiap generasi, kromosom akan melalui proes evaluasi dengan menggunakan alat ukur yang disebut dengan **fungsi fitness**. Nilai fitness dari suatu kromosom akan menunjukkan kualitas kromosom dalam populasi tersbut. Generasi berikutnya dikenal dengan istilah anak (**offspring**) terbentuk dari gabungan 2 kromosom generasi sekarang yang bertindak sebagai induk (**parent**) dengan menggunakan operator penyilangan (**crossover**). Selain operator penyilangan, suatu kromosom dapat juga dimodifikasi dengan menggunakan operator **mutasi**. Populasi generasi yag baru dibentuk dengan cara menyeleksi nilai fitness dari kromosom induk (parent) dan nilai fitness dari kromosom anak (offspring), serta menolak kromosom-kromosom yang lainnya

sehingga ukuran populasi (jumlah kromosom dalam populasi) konstan. Setelah melalui beberapa generasi, maka algoritma ini akan konvergen ke kromosom terbaik.

Secara umum pembelajaran genetik dimulai sebagai berikut :

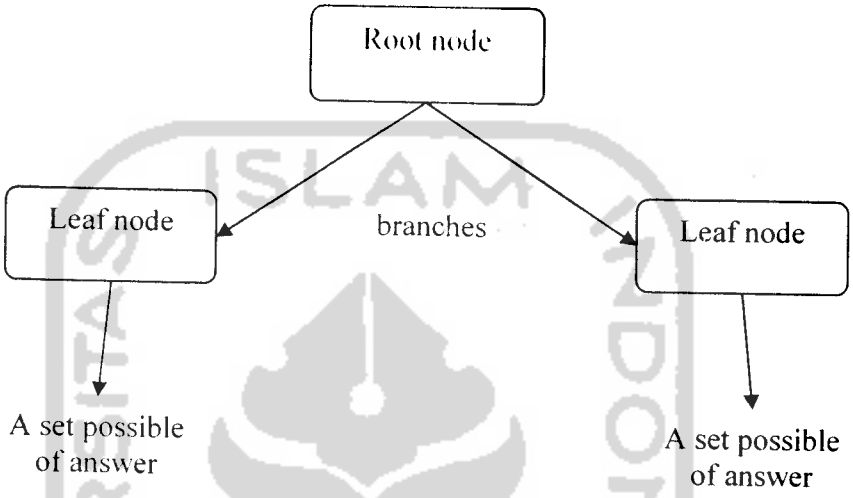
Sebuah populasi awal dibuat yang terdiri dari aturan-aturan yang digenarate secara acak. Tiap rule dapat diwakilkan dengan bit-bit string. Contohnya, sampel-sampel pada *training* set yang diberikan digambarkan dengan dua atribut boolean, A1 dan A2, dan ada 2 class, C1 dan C2. Aturan "IF A1 AND NOT A2 THEN C2" dapat dikodekan dengan bit string "100", dimana dua leftmost bit merepresentasikan atribut A1 dan A2, dan rightmost bit merepresentasikan class. Untuk aturan "IF NOT A1 AND NOT A2

THEN C1 dapat dikodekan dengan "001". Jika atribut mempunyai K nilai, dimana k > 2, maka k bit digunakan untuk mengkodekan nilai-nilai atribut. Class-class dapat dikodekan dengan cara yang sama.

2.1.3 Decision tree

Algoritma ID3 yang pertama kali diperkenalkan oleh Ross Quinlan, digunakan untuk model kualifikasi induksi, atau lebih dikenal sebagai *Decision tree*/pohon keputusan, dari sejumlah data.[DAN97]

Decision tree adalah flow-chart seperti struktur tree, dimana tiap internal node menunjukkan sebuah test pada sebuah atribut, tiap cabang menunjukkan hasil dari test, dan leaf node menunjukkan class-class atau class distribution.



Gambar 2.1 gambar sebuah *decision tree*

Kita bisa melihat kenapa diagram diatas disebut sebagai pohon, dimulai dari akar dan mempunyai cabang yang menuju ke daun(ujung dari grafik paling bawah). Sebagai catatan, daun selalu sebagai sebuah keputusan, dan keputusan khusus mungkin berada di akhir cabang dengan cabang lebih dari satu(Contoh: kita dapat pergi ke biokop dengan dua alasan yang berbeda).

Induksi pohon keputusan adalah pendekatan induksi yang khas untuk mempelajari pengetahuan atas klasifikasi. Syarat pokok untuk melakukan penambangan dengan pohon keputusan ialah:

a. *Attribute-value description*/deskripsi nilai atribut: objek atau kasus harus dapat diungkapkan ke dalam koleksi properti atau atribut yang tetap. Ini berarti harus

mendiskretkan sifat atribut yang kontinyu/terus-menerus, atau telah disediakan di dalam algoritma.

b. *Predefined classes(target attribute values)*/nilai target atribut: nilai dari target harus sudah ditentukan sebelumnya.

c. *Discrette classes:* kelas secara tepat harus didefinisikan. Kelas yang kontinyu harus dipecah ke dalam kategori-kategori yang samar-samar, seperti logam yang "keras, cukup keras, fleksibel, halus, cukup halus".

d. *Sufficient data:* data yang mencukupi, biasanya ratusan atau bahkan ribuan.

Pseudo-code dalam pembentukan pohon keputusan adalah sebagai berikut:

```
Decomposenode(node)
Numdata = node.data.size
NumInputAttribute = numAttributeTarget
Node.entropy = calculateEntropy(node.data)
       For NumInputAttribute

                Numvalues = domain.size
                If alreadyUsedToDecompose then continue
                (jika sudah digunakan sebagai node maka lanjut)

                averageEntropy = 0
                For numValues
                Subset = getSubset(node.data)

                If subset = 0 maka continue
                subEntropy = calculateEntropy(subset)
                averageEntropy += subEntropy*subset.size
                averageEntropy = averageEntropy/numdata

                if selected = false
                selected = true
                bestEntropy = averageEntropy
                selectedAttribute = numAttribute
                else
                if averageEntropy<bestEntropy
                selected = true
                bestEntropy = averageEntropy
```

```
                selectedAttribute = numAttribute
                if selected = false maka return

// membagi dataset menggunakan atribut yang terpilih
    Numvalues = domain[selectedAttribute].size
    Node.decompositionAttribute = selectedAttribute
    Nodechildren = numvalues
       For numvalues
       Nodechildren.data = getSubset(node.data,
       selectedAttribute)
       Nodechildren.decompositionValue = numvalues
```

Nilai-nilai yang ada dalam pseudo-code penyusunan pohon keputusan adalah sebagai berikut:

1. Node adalah titik yang akan dihitung untuk node akar(pada awal) atau cabang dari pohon keputusan.
2. Numdata adalah jumlah data atribut yang akan dikalkulasikan.
3. NumInputAttribute adalah atribut yang akan dihitung.
4. AverageEntropy adalah nilai dari entropy dari atribut yang dihitung.
5. Numvalues adalah jumlah variabel yang ada pada atribut.
6. Subset adalah jumlah data yg sesuai dengan nilai variabel atribut.
7. SelectedAttribute adalah nilai dari atribut yang terpilih sebagai pembagi dalam pohon keputusan.

Ketentuan penilaian di algoritma pohon keputusan adalah seleksi atribut untuk menguji setiap node keputusan di pohon. Tujuannya adalah untuk memilih atribut yang paling berguna untuk mengklasifikasi sample. Ukuran kuantitatif yang baik sebuah atribut adalah properti statistik yang disebut dengan *information gain* yang

menurut sasaran klasifikasinya. Ukuran ini dipakai untuk memilih di antara atribut calon di setiap langkah saat membentuk pohon keputusan. Untuk dapat memperoleh *information gain*, terlebih dahulu menghitung *entropy*. *Entropy* mengukur nilai informasi dari atribut.

Entropy

Untuk dapat memperoleh *information gain* yang tepat, kita perlu menegaskan ukuran yang secara umum dipakai dalam teori informasi, yang disebut entropy, yang menandakan ketidakmurnian koleksi acak dari sample yang diberikan. Diberikan sample set S, entropy dari set S adalah sebagai berikut:

$$Entropy(S) = \sum_{i=1}^{C} p_i \log_2 p_i$$

Dengan:

Pi = bagian kejadian dari dataset yang mengambil nilai dari atribut target

C = jumlah dari dataset

Ukuran kemungkinan ini memberikan indikasi bagaimana ketidakpastian dari data. Dan dalam persamaan menggunakan hitungan Log2 ini melambangkan berapa banyak bit kita akan gunakan untuk menetapkan kelas dalam kejadian acak.

Sebagai contoh, diberikan sample S dengan jumlah datanya adalah 14. Datanya terdiri dari dua nilai yaitu 9 YES dan 5 NO, maka

Entropy(S) = - (9/14) Log2 (9/14) - (5/14) Log2 (5/14) = 0.940

dengan catatan bahwa, entropy = 0 jika seluruh anggota S berasal dari kelas yang sama(data secara utuh telah dikelompokkan). Range entropy datanya adalah 0 (*"perfectly classified"*) sampai 1 (*"totally random"*).

Information Gain

Cara quantitative untuk dapat memahami efek pemisahan dataset dengan memakai sifat khusus (yang merupakan sebagian proses pembentukan pohon) adalah dengan menggunakan Information Gain. Information Gain menghitung penurunan di entropy (bertambah informasi) yang akan menghasilkan pemisahan data dari atributnya. Gain dari atribut A terhadap himpunan data S adalah:

$$Gain\ (S, A) = Entropy\ (S) - \sum_{v \in A} \frac{|S_v|}{|S|} Entropy\ (S_v)$$

Dengan:

v = nilai dari A

|Sv| = subset dari kejadian S

S = jumlah kejadian

A = atribut

Dengan teknik evaluasi cabang ini, dapat secara rekursif melalui subset yang dibuat sampai cabang daun dijangkau seluruhnya dan semua subset murni dengan entropi nol. Ini adalah bagaimana ID3 dan variasi yang lain berjalan.

Dataset S adalah kumpulan dari 14 sample, dimana salah satu atributnya adalah "wind" kecepatan angin. Nilai dari atribut "wind" dapat bernilai kuat

"strong" atau lemah "weak". Klasifikasi dari 14 sample ini adalah 9 YES dan 5 NO. Untuk atribut "wind", terdapat 8 kemunculan bahwa "wind" = "weak" dan 6 kemunculan "wind" = "strong". Untuk "wind" = "weak", 6 sample adalah YES dan 2 NO. Untuk "wind" = "strong", 3 adalah YES dan 3 NO. Oleh karena itu

$$Gain(S,wind) = Entropy(S) - (8/14)*Entropy(S_{weak}) - (6/14)*Entropy(S_{strong})$$

$$= 0.940 - (8/14)*0.811 - (6/14)*1.00$$

$$= 0.048$$

2.1.4 Nearest neighbour method

Nearest Neighbor Classifiers berdasarkan pada proses pembelajaran menggunakan analogi/*learning by analogi*. *Training* sampelnya dideskripsikan dalam bentuk atribut numerik n-dimensi. Tiap sampel mewakili sebuah titik pada ruang n-dimensi. Dengan cara ini, semua *training* sampel disimpan pada pola ruang n-dimensi. Ketika diberikan "unknown" sampel, k-nearest neighbor classifier mencari pola ruang K *training* sampel yang paling dekat "unknown" sampel tersebut. K *training* sampel ini adalah *k nearest neighbor* dari *unknown* sampel. "Closeness: dinyatakan dengan Euclidean Distance, dimana Euclidean Distance antara 2 titik, X = (x1, x2, ….., xn) dan Y = (y1,y2,….,yn) adalah d(X,Y) =

Unknown sampel ditetapkan dengan class yang paling umum diantara k nearest neighborsnya. Ketika k = 1, unknown sampel ditetapkan dengan class dari *training* sampel yang paling dekat dengan pola ruang-nya. Nearest Neighbor menyimpan semua *training* sampel dan tidak membangun classifier sampai sampel

baru (unlabeled) perlu diklasifikasikan, sehingga Nearest Neighbor Classifiers sering disebut dengan instance-based atau lazy learners. Lazy learners dapat menyebabkan biaya penghitungan (computational cost) nya mahal ketika jumlah dari neighbor yang potensial (yakni, *training* sampel yang disimpan) dengan unlabeled sampel yang akan dibandingkan sangat besar. Oleh karena itu, diperlukan teknik pengindeksan yang efisien. Kenyataannya, metode lazy learning lebih cepat pada saat *training* daripada metode eager, tetapi lebih lambat pada saat *classification* karena semua penghitungan ditunda sampai pada saat itu.

2.1.5 Rule induction

Ekstraksi aturan if-then dari data berdasarkan arti statistik.[PAL07]

2.1 Sekilas Tentang JAVA

Perancangan bahasa Java dimulai pada tahun 1991 oleh satu grup programmer dari Sun yang dipimpin oleh Patrick Naughton dan James Gosling. Pada awalnya Java dirancang dengan konsep sebuah bahasa pemrograman yang akan berjalan di perangkat-perangkat mini konsumen seperti pada kotak switch pada Televisi Kabel di AS. Karena perangkat tersebut tidak memiliki memory dan power yang cukup besar maka bahasa pemrograman yang dirancang haruslah kecil, kompak serta tidak memakan memory yang besar. Sebagai tambahan bahwa semua perancang perangkat-perangkat mini menggunakan arsitektur prosessor yang

berbeda maka bahasa pemrograman tersebut tidak boleh terikat pada arsitektur tertentu.

Proyek ini dimulai dengan nama sandi "Green". Sebagai model dari proyek "Green" ini, tim programmer tersebut berencana membangkitkan kembali sebuah proyek gagal yang pernah dikerjakan oleh Niklaus Wirth-pencipta bahasa Pascal, yaitu sebuah virtual machine yang memungkinkan sebuah bahasa pemrograman dijalankan secara independen di platform apapun. Bahasa pemrograman tersebut harus haruslah dicompile sehingga menghasilkan sebuah intermediate code yang kelak akan diinterpretasikan oleh sebuah interpreter yang cocok di virtual machine tersebut. Virtual machine ini adalah cikal bakal dari JVM(Java Virtual Machine).

Sun merilis Java 1.0 pada tahun 1996, Java 1.0 sangatlah lemah dan banyak orang menganggap bahasa ini tidak diperuntukkan untuk pemrograman aplikasi yang serius. Java 1.2 dirilis dua tahun setelah Java 1.0 dirilis, kali ini Java datang dengan toolkits grafik yang sangat memadai serta lebih dekat dengan motto Sun yaitu "Write Once Run Anywhere". Tiga hari setalah dirilis pada Desember 1998, tim marketing dari Sun mengganti namanya menjadi Java 2 Standard Edition Software Development Kit Version 1.2. Disamping J2SE, Sun juga merilis J2ME(Java 2 Micro Edition) untuk aplikasi ponsel dan J2EE(Java 2 Enterprise Edition) untuk aplikasi Server.

Versi 1.3 dan 1.4 dari J2SE dirilis setelah itu dengan penambahan di library dan menutupi bug-bug yang ada. Java pada saat ini menjadi pilihan sebagai aplikasi server oleh banyak programmer.

Versi 5.0 merupakan rilis pertama semenjak Java 1.1 yang men-update bahasa Java secara besar-besaran. Versi ini seharusnya 1.5 akan tetapi oleh tim marketing dari Sun pada konferensi JavaOne pada tahun 2004 diganti menjadi 5.0). Penambahan fiturfitur seperti loop "for each" yang ada di C# dan generic types seperti yang ada di templates C++ membuka lembaran baru bagi Java sebagai bahasa pemrograman pilihan programmer.

Bukan sekedar bahasa tapi juga platform sekaligus arsitektur. Java mempunyai portabilitas yang sangat tinggi. Ia dapat berada pada smartcard, pager, POS (Point of Service), handphone, PDA, palm, TV, Embedded device (PLC, micro controller), laptop, pc, dan bahkan server). Menyadari akan hal ini (one size doesn't fit all) Sun membagi arsitektur Java membagi tiga bagian, yaitu:

1. Standar Java (J2SE), ini adalah edisi pertama yang berisi standar *Java Run Time Environment* (JRE) dan standar *Java Development Kit* (JDK). Platform ini ditujukan untuk pengembangan program aplikasi Java pada komputer desktop dan aplikasi web.
2. Micro Java (J2ME) merupakan subset dari J2SE dan salah satu aplikasinya yang banyak dipakai adalah untuk wireless device / mobile device.

3. Enterprise Java (J2EE) untuk aplikasi berbasis web, aplikasi sistem tersebar dengan beraneka ragam klien dengan kompleksitas yang tinggi. Merupakan superset dari Standar Java

2.1.1 J2SE(Java™ Platform, Standard Edition)

JavaTM Platform, Standard Edition adalah inti dari bahasa pemrograman *java* yang merupakan pondasi dari platform *J2EE* dan *J2ME*. *J2SE* digunakan dalam pengembangan aplikasi desktop dan web. Untuk pengembangan aplikasi diperlukan suatu alat yang disebut *JDK (Java Development Kit)* yang digunakan untuk mengkompilasi dan menjalankan program *java*, di dalamnya terdapat *tool* untuk mengkompilasi program aplikasi berbasis *java*.

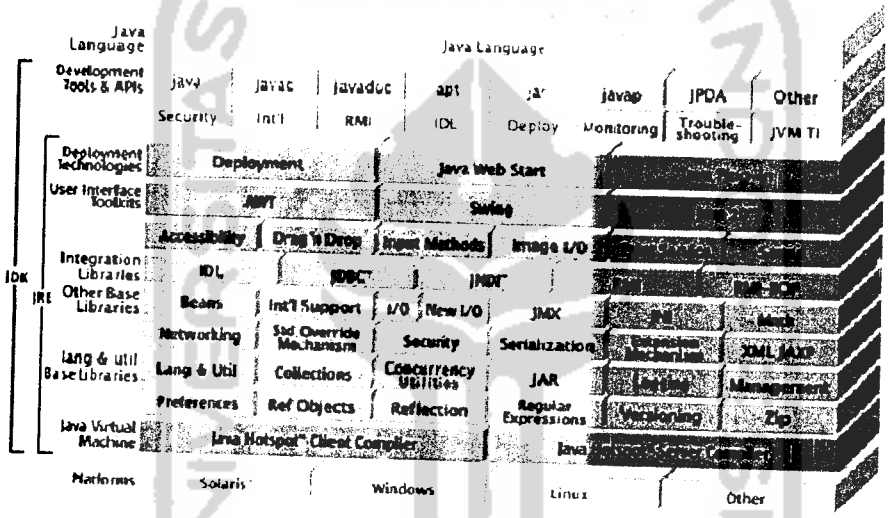
Sun Microsystems pada dasarnya menyediakan 2 buah produk yang bisa didownload secara terpisah dari websitenya yang mana merupakan bagian dari Java 2 Standard Edition.

2.2.1.1 J2SE Runtime Environment(JRE)

JRE merupakan bagian terpenting bagi user ataupun developer karena JRE menyediakan JVM(Java Virtual Machine), perpustakaan(library) dan semua komponen yang diperlukan bagi applet ataupun program Java yang lain agar bias berjalan sebagaimana mestinya.

2.2.1.2 J2SE Development Kit(JDK)

JDK sendiri terdiri dari JRE dan tool-tool pengembangan yang diperlukan oleh para developer seperti contohnya compiler dan debugger berbasis teks(command line).

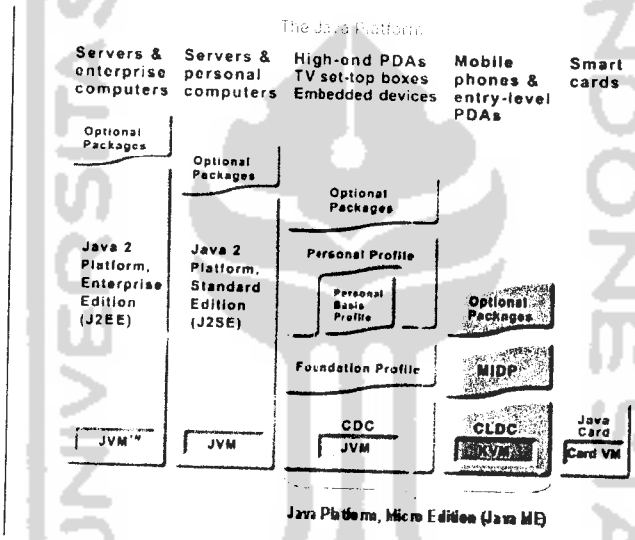


Gambar 2.2 Platform Java 2 Standard Edition (J2SE).

2.2.2 J2ME(Java™ Platform, Micro Edition)

Adalah lingkungan pengembangan yang didesain untuk meletakan perangkat lunak *java* pada barang elektronik beserta perangkat pendukungnya. Biasa digunakan pada telepon selular, *pager*, *Personal Digital Assistants (PDA)* dan sejenisnya. *Java Micro Edition* adalah bagian dari *Java Standard Edition* sehingga juga memiliki keterbatasan terutama jika diaplikasilan pada ponsel. *Java Micro Edition* sangat tergantung pada perangkat (*device*) yang digunakan, bisa dari segi

merek ponsel maupun kemampuan ponsel dan dukungan terhadap teknologi *Java Micro Edition*. Keterbatasan lainya adalah pada ukuran aplikasi, karena memori pada ponsel terbatas. Sebagaian besar ponsel tidak mengijinkan aplikasi *Java Micro Edition* menulis *file* karena alasan keamanan. Arsitektur *Java Micro Edition* dapat dilihat pada gambar 2.3.

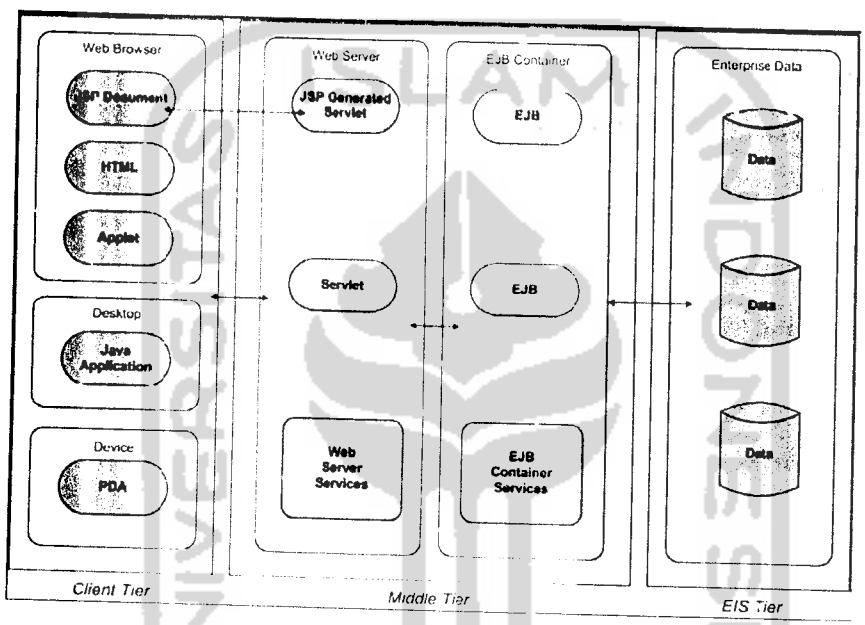


Gambar 2.3 Arsitektur *Java Micro Edition*

2.2.3 J2EE(Java^{TM} Platform, Enterprice Edition)

Java^{TM} Platform, Enterprice Edition adalah kumpulan teknologi yang cukup kuat dan berada di atas lingkungan *Java^{TM} Platform, Standard Edition Java^{TM} Platform, Enterprice Edition* berbasis pada *java2* yang berusaha untuk menyediakan sebuah lingkungan aplikasi yang bersifat reliable dan stabil serta dapat dijalankan pada beberapa lingkungan sistem operasi. *Java Enterprice Edition* menggunakan modul aplikasi multitier terdistribusi untuk aplikasi perusahaan.

Application logic dibagi menjadi komponen-komponen sesuai dengan fungsinya dan komponen-komponen aplikasi lainya Arsitektur *Java Enterprise Edition* dapat dilihat pada gambar 2.4.



Gambar 2.4 Arsitektur Aplikasi *Java 2 Enterprise Edition*

2.2 UML

UML (*Unified Modeling Language*) adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan software berbasis OO (*Object-Oriented*). UML sendiri juga memberikan standar penulisan sebuah sistem blue print, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem software (http://www.omg.org).

Pendekatan analisa & rancangan dengan menggunakan model OO mulai diperkenalkan sekitar pertengahan 1970 hingga akhir 1980 dikarenakan pada saat itu aplikasi software sudah meningkat dan mulai komplek. Jumlah yang menggunakaan metoda OO mulai diuji cobakandan diaplikasikan antara 1989 hingga 1994, seperti halnya oleh Grady Booch dari *Rational Software Co.,* dikenal dengan OOSE (*Object-Oriented Software Engineering*), serta James Rumbaugh dari *General Electric*, dikenal dengan OMT (*Object Modelling Technique*).

OMT

Object Modelly Technique yang dibuat *Rumbaugh* bertujuan untuk rancangan analisis terstruktur dengan pemodelan *entity-relationship*.

Tahapan:

1. Analisis
2. Desain Sistem
3. Desain Obyek
4. Implementasi

OOSE

Object Oriented Software Engineering dibuat *Jacobson* adalah perancangan yang lebih menekankan pada *Use case*.

Tahapan:

1. Membuat model Requirement dan Analisis.
2. Desain dan Implementasi

3. Model Pengujian (*Test Model*)

Kelemahan saat itu disadari oleh Booch maupun Rumbaugh adalah tidak adanya standar penggunaan model yang berbasis OO, ketika mereka bertemu ditemani rekan lainnya Ivar Jacobson dari Objectory mulai mendiskusikan untuk mengadopsi masing-masing pendekatan metoda OO untuk membuat suatu model bahasa yang uniform / seragam yang disebut UML (*Unified Modeling Language*) dan dapat digunakan oleh seluruh dunia.

Dalam suatu proses pengembangan software, analisa dan rancangan telah merupakan terminologi yang sangat tua. Pada saat masalah ditelusuri dan spesifikasi dinegoisasikan, dapat dikatakan kita berada pada tahap rancangan. Merancang adalah menemukan suatu cara untuk menyelesaikan masalah, salah satu tool / model untuk merancang pengembangan software yang berbasis object oriented adalah UML.[DHA03]

Konsep Objek

Obyek dalam *'software analysis & design'* adalah sesuatu berupa konsep (*concept*), benda (*thing*), dan sesuatu yang membedakannya dengan iingkungannya. Secara sederhana obyek adalah mobil, manusia, *alarm* dan lainlainnya. Tapi obyek dapat pula merupakan sesuatu yang abstrak yang hidup didalam sistem seperti tabel, *database, event, system messages.*

Obyek dikenali dari keadaannya dan juga operasinya. Sebagai contoh sebuah mobil dikenali dari warnanya, bentuknya, sedangkan manusia dari suaranya.

Ciri-ciri ini yang akan membedakan obyek tersebut dari obyek lainnya. Alasan mengapa saat ini pendekatan dalam pengembangan software dengan *object-oriented*, pertama adalah *scalability* dimana obyek lebih mudah dipakai untuk menggambarkan sistem yang besar dan komplek. Kedua *dynamic modeling*, adalah dapat dipakai untuk permodelan sistem dinamis dan *real time*.

Teknik Dasar OOA/D *(Object-Oriented Analysis/Design)*

Dalam dunia pemodelan, metodologi implementasi obyek walaupun terikat kaidah-kaidah standar, namun teknik pemilihan obyek tidak terlepas pada subyektifitas software analyst & designer. Beberapa obyek akan diabaikan dan beberapa obyek menjadi perhatian untuk diimplementasikan di dalam sistem. Hal ini sah-sah saja karena kenyataan bahwa suatu permasalahan sudah tentu memiliki lebih dari satu solusi. Ada 3 (tiga) teknik/konsep dasar dalam OOA/D, yaitu pemodulan (*encapsulation*), penurunan (*inheritance)* dan *polymorphism.*

a. Pemodulan (*Encapsulation*)

Pada dunia nyata, seorang ibu rumah tangga menanak nasi dengan *menggunakan rice cooker*, ibu tersebut menggunakannya hanya dengan menekan tombol. Tanpa harus tahu bagaimana proses itu sebenarnya terjadi. Disini terdapat penyembunyian informasi milik *rice cooker*, sehingga tidak perlu diketahui seorang ibu. Dengan demikian menanak nasi oleh si ibu menjadi sesuatu yang menjadi dasar bagi konsep *information hiding.*

b. Penurunan (*Inheritance*)

Obyek-obyek memiliki banyak persamaan, namun ada sedikit perbedan. Contoh dengan beberapa buah mobil yang mempunyai kegunaan yang berbeda-beda. Ada mobil bak terbuka seperti truk, bak tertutup seperti sedan dan minibus. Walaupun demikian obyek-obyek ini memiliki kesamaan yaitu teridentifikasi sebagai obyek mobil, obyek ini dapat dikatakan sebagai obyek induk (*parent*). Sedangkan minibus dikatakan sebagai obyek anak (*child*), hal ini juga berarti semua operasi yang berlaku pada mobil berlaku juga pada minibus.

c. Polymorphism

Pada obyek mobil, walaupun minibus dan truk merupakan jenis obyek mobil yang sama, namun memiliki juga perbedaan. Misalnya suara truk lebih keras dari pada minibus, hal ini juga berlaku pada obyek anak (*child*) melakukan metoda yang sama dengan algoritma berbeda dari obyek induknya. Hal ini yang disebut *polymorphism*, teknik atau konsep dasar lainnya adalah ruang lingkup/pembatasan. Artinya setiap obyek mempunyai ruang lingkup kelas, atribut, dan metoda yang dibatasi.

Aspek modeling yang berbeda

Ada tiga aspek utama dalam pemodelan sistem yang mampu didukung oleh UML:

a. Functional Model, untuk menunjukkan fungsionalitas dari suatu sistem dari sudut pandang user atau pengguna. Ini dicapai dengan menggunakan Use Case Diagram.

b. Object Model, untuk menunjukkan struktur dan substruktur dari suatu sistem dengan menggunakan object, atribut, operasi dan juga asosiasi. Ini dicapai dengan menggunakan Class Diagram.

c. Dynamic Model, menunjukkan internal behavior dan suatu sistem. Ini dicapai dengan menggunakan Sequence Diagram, Activity Diagram dan juga Statechart Diagram.

BAB III

METODOLOGI

3.1 Analisis Kebutuhan

3.1.1 Metode Analisis

Metode analisis dalam penelitian ini, digunakan metode dengan pendekatan berarah-obyek (*object oriented*). Dengan pendekatan tersebut, program bersifat *modular*. Dalam konsep pemrograman berorientasi obyek, modul-modul tersebut dapat digunakan kembali (*reusable*), sehingga dalam pembuatan aplikasi yang besar, dimudahkan bagi pengembang untuk memelihara aplikasi yang dikembangkan. Metode analisis ini juga dilakukan dengan pengumpulan data melalui observasi, studi literatur berupa buku dan sumber lainnya yang mendukung dan berkaitan dengan obyek yang diteliti yaitu *Decision Tree*.

Terdapat standar dalam perancangan berarah objek yaitu dengan standar *UML*. *UML* digunakan memodelkan perancangan piranti lunak (sistem) sebelum rancangan tersebut diimplementasikan ke dalam bahasa pemrograman. *UML* memodelkan perancangan dengan beberapa macam diagram, diantaranya *use case diagram, class diagram,* dan *sequence diagram.*

Dengan digunakannya rancangan model sebelum dilakukan implementasi perangkat lunak, diharapkan pengembangan sistem dapat berjalan sesuai dengan

rancangan pernah dibuat sebelumnya dan mencapai tujuan yang diharapkan dari proses perancangan.

3.1.2 Hasil Analisis

Dari analisis yang telah dilakukan sebelumnya, maka akan didapatkan hasil analisis. Hasil analisis adalah langkah untuk menentukan kebutuhan perangkat lunak. Kebutuhan yang dimaksud adalah kebutuhan masukan *(input)*, proses keluaran *(output)*. diperangkat lunak, antarmuka dari sistem yang telah dirancang dan yang akan diimplementasikan ke dalam bahasa pemrograman.

3.1.2.1 Kebutuhan Masukan

Kebutuhan masukan dari sistem adalah data berupa tabel. Data tabel yang dimaksud adalah data yang mempunyai minimal dua kolom atribut. Satu kolom sebagai kolom atribut masukan dan satu kolom sebagai kolom atribut target. Dari setiap kolom terdapat nilai-nilai yang akan dipergunakan untuk kalkulasi, dan nilai dari setiap atribut harus bersifat diskret. Dalam melakukan kalkulasi, keakuratan data sangatlah penting, oleh karena itu data masukan yang diberikan harus telah teruji dan layak untuk diproses.

Ketentuan lain yang harus dipenuhi agar masukan dapat diproses dengan lancar adalah peletakan kolom target harus berada pada posisi terakhir dari kolom pada tabel masukan. Sistem akan membaca masukan dengan atribut target berada pada kolom terakhir dari tabel, maka dari itu selain kolom terakhir sistem akan

mengenalinya sebagai atribut masukan dari sistem. Nilai dari setiap atribut adalah *Case Sensitive* yang berarti bahwa besar kecilnya huruf akan mempunyai persepsi nilai yang berbeda. Ketentuan yang tidak kalah pentingnya adalah jumlah data yang disediakan dianjurkan tidak terlalu sedikit, apabila terlalu sedikit perolehan pengetahuan juga akan semakin sedikit dan akan menurunkan tingkat keakuratan hasil pohon keputusan.

Sebagai contoh, diberikan data untuk menentukan apakah kita dapat bermain atau tidak dengan kendala-kendala seperti disebutkan pada tabel 3.1.

Outlook	Temperature	Humidity	Wind	Play ball
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Tabel 3.1 tabel data masukan "play ball"

Kolom pertama sampai dengan kolom kelima adalah kolom atribut masukan, dan kolom terakhir adalah kolom atribut target. Jumlah dari sample diatas adalah 14

dengan nilai atributnya adalah diskret. Dari data diatas diperoleh informasi seperti pada tabel 3.2

Atribut	Kemungkinan Nilai
Outlook	Sunny, Overcast, Rain
Temperature	Hot, Mild, Cool
Humidity	High, Normal
Wind	Strong, Weak
Play ball	Yes, No

Tabel 3.2 attribute dan nilai kemungkinannya

Data masukan dari sistem yang telah dipilih akan ditampilkan oleh sistem pada bagian sebelah kiri dari interface sistem. Sistem akan menampilkan isi dari data yang diberikan oleh user dalam bentuk tabel, dengan nama kolom dan jumlah data yang sama seperti user berikan.

3.1.2.2 Kebutuhan Proses

Setelah sistem membaca data masukan yang telah diberikan, maka proses kalkulasi akan segera dimulai. Sistem akan mulai dengan menghitung nilai entropy atribut dari kolom target. Nilai entropy akan digunakan untuk proses kalkulasi dalam perolehan nilai *information gain.*

Salah satu contoh proses kalkulasi dari entropy adalah proses kalkulasi nilai entropy target yaitu Play ball dengan jumlah sample adalah 14 dengan 9 YES dan 5 NO. Nilai entropy-nya adalah

Entropy(S) = - (9/14) Log2 (9/14) - (5/14) Log2 (5/14) = 0.940

Apabila nilai entropy telah didapat, maka langkah selanjutnya yang akan dilakukan oleh sistem adalah menghitung nilai *information gain*. Penghitungan nilai *information gain* untuk atribut Wind adalah sebagai berikut:

Wind = weak kemunculannya berjumlah 8, Wind = strong berjumlah 6. Untuk Wind = weak 6 sample YES dan 2 sample NO. Untuk Wind = strong 3 sample YES dan 3 sample NO.

$$\text{Entropy}(S_{weak}) = -(6/8)*\log2(6/8) - (2/8)*\log2(2/8) = 0.811$$

$$\text{Entropy}(S_{strong}) = -(3/6)*\log2(3/6) - (3/5)*\log2(3/6) = 1.00$$

$$\text{Gain}(S,\text{Wind}) = \text{Entropy}(S) - (8/14)*\text{Entropy}(S_{weak}) - (6/14)\text{Entropy}(S_{strong})$$

$$= 0.940 - (8/14)*0.811 - (6/14)*1.00$$

$$= 0.048$$

Gain(S, Outlook) = 0.246

Gain(S, Temperature) = 0.029

Gain(S, Humidity) = 0.151

Karena *information gain* outlook adalah yang paling besar, maka outlook digunakan sebagai atribut keputusan dalam root node.

Karena outlook mempunyai tiga nilai kemungkinan, maka root node mempunyai tiga cabang (sunny, overcast, rain). Langkah selanjutnya adalah

menghitung *information gain* untuk nilai atribut sisanya, Humidity, Temperature, Wind.

S_{sunny} = {D1, D2, D8, D9, D11} = 5 sample dengan outlook = sunny

Gain(S_{sunny}, Humidity) = 0.970

Gain(S_{sunny}, Temperature) = 0.570

Gain(S_{sunny}, Wind) = 0.019

Humidity mempunyai nilai *information gain* yang paling tinggi, oleh karena itu digunakan sebagai node keputusan. Proses ini berlangsung sampai semua data terklasifikasi semuanya atau kita kehabisan atribut.

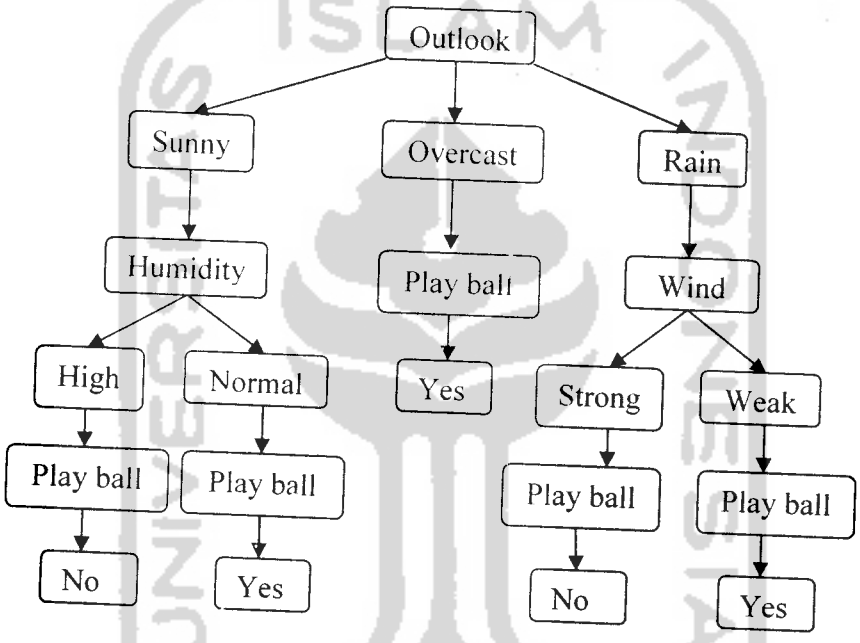
3.1.2.3 Kebutuhan Keluaran Sistem

Untuk kebutuhan keluaran dari sistem ini adalah sebuah *decision tree*/pohon keputusan. Data masukan yang semula jumlahnya sangat banyak dan kompleks, dengan hasil keluaran dari sistem akan mempermudah user dalam memahami isi dari data masukan yang telah diberikan. Bentuk dan ukuran dari pohon keputusan otomatis lebih ringkas dan lebih mudah untuk dilihat.

Dari data masukan sistem yang semula banyak dan kompleks, dengan keluaran sistem berupa sebuah pohon keputusan akan mempermudah user dalam

memahami isi dari data tersebut. Dari hasil pohon keputusan tersebut, diharapkan dapat membantu user dalam pengambilan keputusan.

Contoh keluaran dari sistem dari hasil kalkulasi data diatas terlihat pada gambar 3.1



Gambar 3.1 *Decision tree* **aplikasi datamining**

Membaca pohon keputusan

Terdapat hubungan antara representasi pohon keputusan dengan representasi logika yang dapat diambil agar pohon keputusan yang dipelajari agar lebih mudah dimengerti(dibaca). Setiap pohon keputusan sebenarnya adalah sebuah disjungsi dari suatu implikasi (if ... then statements). Dari pohon diatas, kita dapat membacanya dari *root*(akar) *node* sampai dengan *leaf*(daun) *node:*

If Outlook = Sunny and Humidity = High then Play ball = No

If Outlook = Sunny and Humidity = Normal then Play ball = Yes

If Outlook = Overcast then Play ball = Yes

If Outlook = Rain and Wind = Strong then Play ball = No

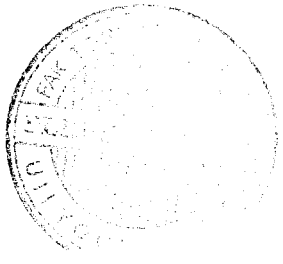
If Outlook = Rain and Wind = Weak then Play ball = Yes

Decision tree tidak harus menjadi representasi dalam proses pengambilan keputusan, tetapi bisa dijadikan sebagai pengelompokan masalah. Apabila kita mengambil makna tersebut diatas, kita dapat memahami bahwa: jika kita ingin proses pengambilan keputusan untuk bermain bola, kita akan bermain bola jenis apa, apakah sepak bola, atau bola basket dan sebagainya.

3.1.2.4 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang diperlukan untuk melakukan penelitian Datamining Menggunakan Algoritma ID3 dengan J2SE adalah sebagai berikut:

a. Sistem operasi dalam pengembangan perangkat lunak adalah *Microsoft Windows XP Professional SP 1 Edition*, meskipun tidak menutup kemungkinan menggunakan sistem operasi lainnya.
b. *Java™ Platform, Standard Edition* dengan *JDK* menggunakan *Sun Java JDK 1.6*, sebagai platform pengembangan perangkat lunak.
c. *IDE Netbean 5.5* digunakan dalam pengembangan sistem.



3.1.2.5 Kebutuhan Perangkat Keras

Untuk pengembangan aplikasi, penulis menggunakan *PC* menggunakan komputer dengan spesifikasi sebagai berikut:

a. *Processor AMD Athlon XP 2000.*
b. *Monitor*
c. *RAM DDR 512 MB.*
d. *Hardisk 120 GB.*

3.1.2.6 Kebutuhan Antarmuka

Kebutuhan antarmuka yang dibutuhkan yaitu antarmuka menu utama yang dilengkapi dengan antarmuka pendukung seperti antarmuka pemilihan file, dan antarmuka bila terjadi kesalahan dalam sistem.

3.2 Perancangan Sistem

3.2.1 Metode Perancangan

Perancangan sistem datamining ini, digunakan metodologi berarah obyek, orientasi tersebut memandang suatu sistem menjadi obyek-obyek, artinya sistem dipecah menjadi obyek-obyek yang lebih kecil. Untuk memodelkannya digunakan pemodelan UML (*Unified Modelling Language*) yang menjadi standar dalam pemodelan berarah obyek dan mempunyai karakteristik yaitu bahasa yang *process-independent* (bahasa yang saling bebas terhadap proses didalamnya).

3.2.2 Hasil Perancangan

Hasil perancangan dari sistem datamining ini, berupa aplikasi yang berfungsi untuk menciptakan sebuah pohon keputusan atau *decision tree* dari input yang berupa sebuah tabel yang dibaca sistem dari sebuah file dengan format filenya adalah text.

3.2.3 Perancangan *Unified Modelling Language (UML)* dan Lainnya

Dalam penelitian ini, digunakan beberapa diagram saja yang dapat mewakili perancangan sistem. Dalam perancangan sistem datamining ini digunakan (dibatasi) oleh tiga buah diagram beserta kegunaannya, sebagai berikut:

a. *Use Case Diagram*, mempunyai kegunaan yaitu bagaimana pengguna berinteraksi dengan sistem.
b. *Class Diagram*, mempunyai kegunaan yaitu hubungan-hubungan antara class, fitur dan
c. *Sequence Diagram*, mempunyai kegunaan yaitu interaksi antar obyek.

3.2.3.1 *Use Case Diagram*

Berikut penjelasan dari diagram *use case* yang menggambarkan hubungan aktor dengan *use case* pada sistem datamining ini dan untuk lebih jelasnya dapat dilihat pada gambar 3.2:

a. Sistem dalam hal ini adalah sistem datamining yaitu lingkungan dimana semua komponen *usecase* itu berada.

b. Aktor yang terlibat dalam sistem adalah *end user* yang berfungsi untuk melakukan pemasukan data tabel yang akan digunakan sistem untuk pembuatan pohon keputusan atau *decision tree*.

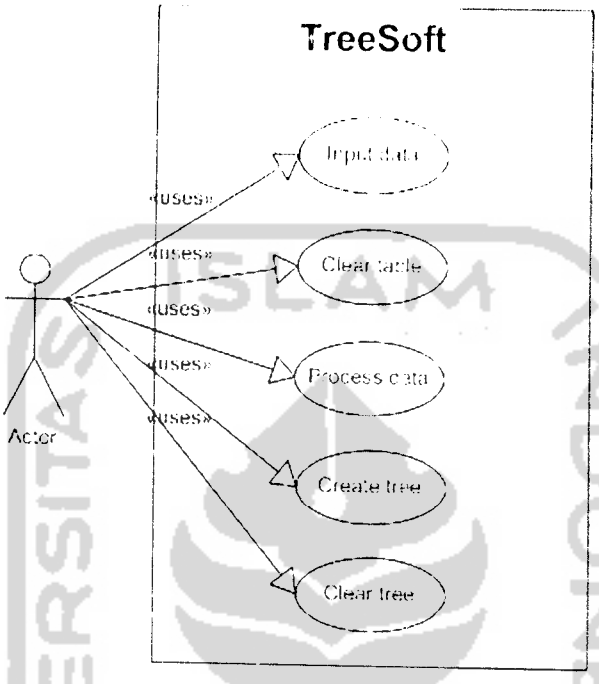
c. *Use case input table* akan melakukan pembacaan data dari sebuah file text yang kemudian akan ditampilkan ke dalam tabel

d. *Use case clear table* akan melakukan penghilangan data yang ada pada tabel

e. *Use case process data* akan melakukan komputasi terhadap data yang dibaca dari tabel kemudian akan menyimpan nilainya di dalam system

f. *Use case create tree* akan mengambil data dari sistem dari hasil kalkulasi dan akan menampilkan data tersebut ke dalam bentuk tree yang merupakan sebuah pohon keputusan

g. *Use case clear tree* akan melakukan penghilangan pohon keputusan



Gambar 3.2 *Use case diagram* **pada datamining**

3.2.3.2 *Class Diagram*

Aplikasi yang menggunakan perancangan berorientasi objek dapat diilustrasikan dalam struktur kelas-kelas dan hubungan antar kelas yang ada. Dalam UML digunakan notasi class diagram untuk menggambarkan hal tersebut.

Pada aplikasi J2SE, kelas yang akan dibangun dalam aplikasi datamining akan dikelompokkan dalam suatu paket (*package*), yaitu package mysoft. Kelas-kelas tersebut antara lain:

1. Class Datamining

   Merupakan kelas utama untuk menampilkan data input dan hasil dari pembelajaran.

2. Class Filter

   Merupakan kelas yang digunakan untuk melakukan filter terhadap file yang akan dijadikan sebagai input.

3. Class Utils

   Merupakan kelas yang digunakan untuk menentukan jenis penyaringan jenis file yang telah ditentukan.

4. Class dataTable

   Merupakan kelas yang digunakan untuk menampilkan nilai dari input data ke dalam tabel.

5. Class ID3

   Merupakan kelas yang digunakan untuk melakukan kalkulasi dari sistem pembelajaran.

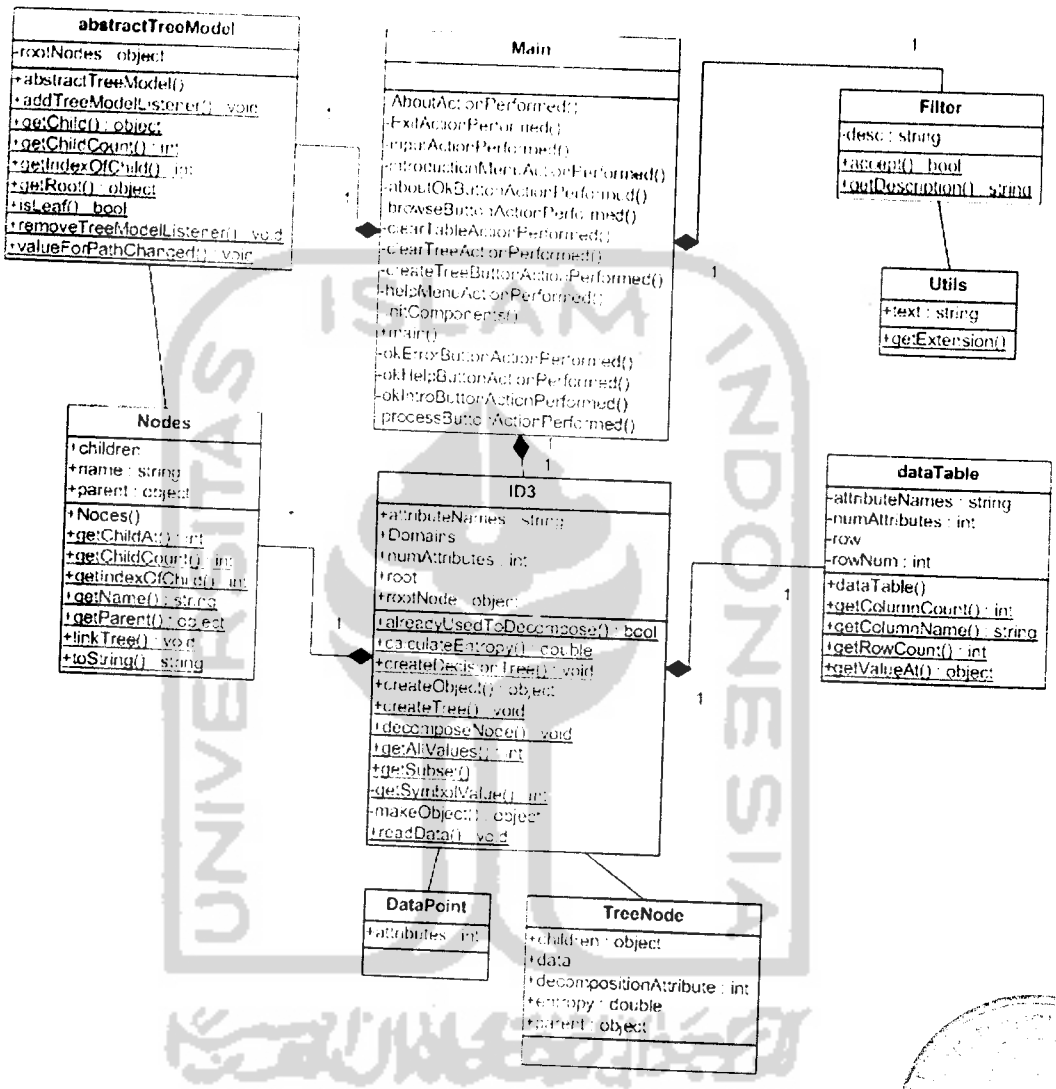
6. Class Nodes

   Merupakan kelas yang digunakan untuk membuat node-node dalam decision tree.

7. Class abstractTreeModel

   Merupakan kelas yang digunakan untuk menampilkan struktur pohon dari hasil kalkulasi dari input sistem, yang merupakan sebuah decision tree.

Lebih jelasnya *class diagram* dapat dilihat pada gambar 3.3



Gambar 3.3 *Class Diagram* **aplikasi datamining**

3.2.3.3 *Sequence Diagram*

Sequence Diagram menggambarkan perilaku sistem secara dinamis dan memperlihatkan interaksi dari obyek-obyek. Interaksi antar obyek dapat disusun berdasarkan urutan waktu yang menunjukkan skenario dan urut-urutan pertukaran data.

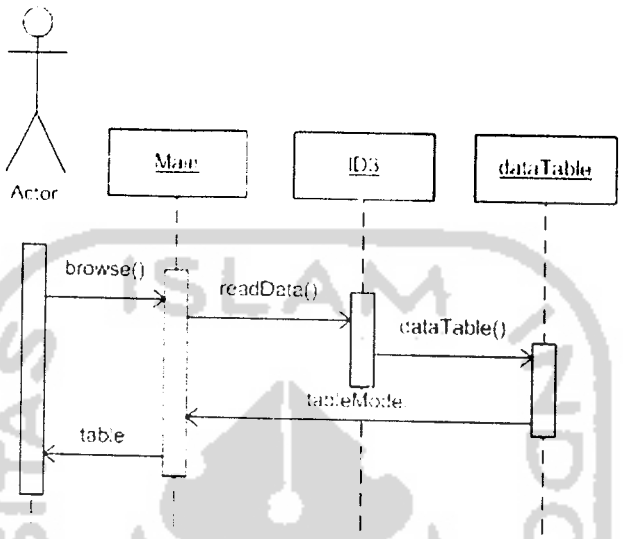
Dari tahapan analisis kebutuhan yang dilakukan sebelumnya maka dapat dibentuk beberapa *sequence diagram* untuk menunjukkan urutan-urutan proses dari masing-masing *use case*.

Pada aplikasi datamining, terdapat beberapa *sequence diagram*, yaitu:

a. *Sequence diagram input table* untuk memilih masukan sistem.
b. *Sequence diagram clear table* untuk menghapus isi dari tabel.
c. *Sequence diagram process data* untuk memproses data.
d. *Sequence diagram clear tree* untuk menghapus *decision tree*.
e. *Sequence diagram create tree* untuk membuat pohon keputusan.

3.2.3.3.1 Sequence Diagram untuk skenario input table

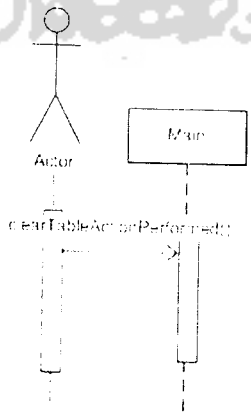
Diagram ini menjelaskan tentang bagaimana cara memberikan data masukan ke dalam sistem. Seorang aktor akan memulai skenarionya dengan cara menekan tombol browse pada sistem atau melalui sub menu input data. Dengan demikian, aktor akan memilih data masukan yang akan digunakan untuk diproses. Setelah data yang tepat telah dipilih, sistem akan mulai membaca data yang diberikan. Data akan dibaca dimulai dari baris pertama yang merupakan nama dari kolom atribut, diikuti dengan nilai atribut pada baris berikutnya. Setelah data selesai dibaca oleh sistem, maka sistem akan menampilkan data ke dalam tabel yang berada pada panel bagian kiri pada sistem. Gambar 3.4 menunjukkan *sequence diagram* untuk skenario *input table* yang dilakukan oleh *actor user*.



Gambar 3.4 *Sequence Diagram Input Table*

3.2.3.3.2 *Sequence Diagram* untuk skenario menghapus isi tabel

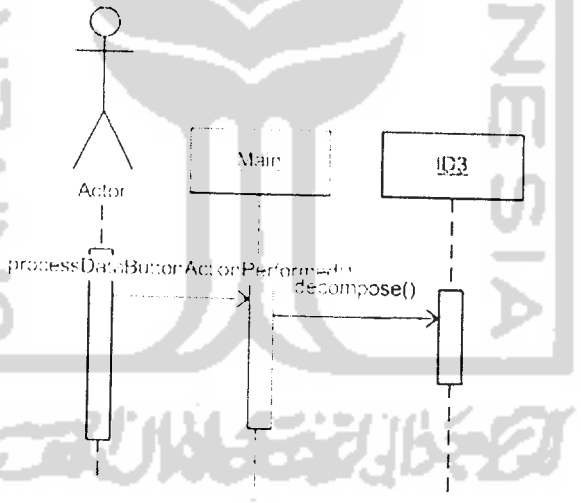
Diagram ini akan menjelaskan tentang aktor dalam menghapus isi dari tabel. Cara aktor dalam menghapus tabel sangat sederhana, yaitu hanya dengan menekan tombol clear table yang telah tersedia pada sistem. Gambar 3.5 menunjukkan *sequence diagram* untuk skenario *clear table* yang dilakukan oleh *actor user*.



Gambar 3.5 *Sequence Diagram clear table*

3.2.3.3.3 Sequence Diagram untuk skenario process data

Untuk memproses data, diagram dibawah ini memperjelas bagaimana sistem akan menjalankan prosesnya. Aktor akan menekan tombol process data, dengan syarat bahwa masukan telah tersedia sesuai dengan syaratnya yang dibutuhkan, maka aktor dapat melakukan pemrosesan data. Setelah diproses, maka data akan tersimpan di dalam sistem. Nilai yang tersimpan adalah nilai dari pengetahuan dari data yang telah diberikan sebagai masukan dari sistem. Gambar 3.6 menunjukkan *sequence diagram* untuk skenario *process data* yang dilakukan oleh *actor* user.

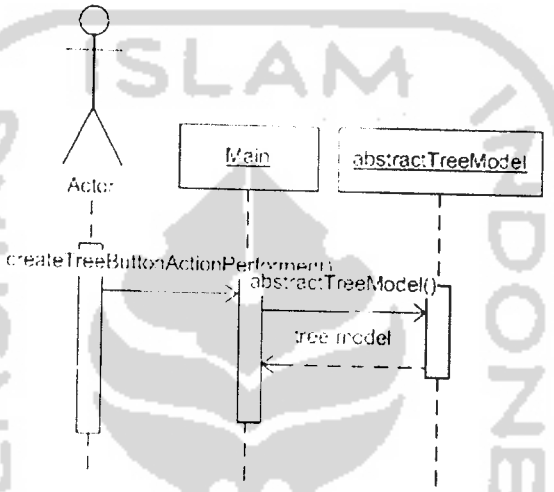


Gambar 3.6 *Sequence Diagram Process data*

3.2.3.3.4 Sequence Diagram untuk skenario Create Tree

Di bawah ini adalah diagram yang menjelaskan bagaimana aktor akan membuat pohon keputusan. Setelah data telah diproses dan tersimpan di dalam sistem, dengan menekan tombol create tree maka sistem akan mengambil nilai yang

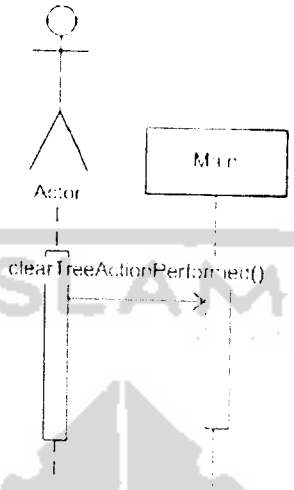
sudah tersimpan untuk membuat sebuah pohon keputusan. Gambar 3.7 menunjukkan *sequence diagram* untuk skenario *create tree* yang dilakukan oleh *actor* user.



Gambar 3.7 *Sequence Diagram Create tree*

3.2.3.3.5 Sequence Diagram untuk skenario Clear tree

Skenario terakhir di dalam sistem ini adalah menghapus pohon keputusan. Pohon keputusan akan segera terhapus dengan cara aktor menekan tombol clear tree. Gambar 3.8 menunjukkan *sequence diagram* untuk skenario *clear tree* yang dilakukan oleh *actor user*.



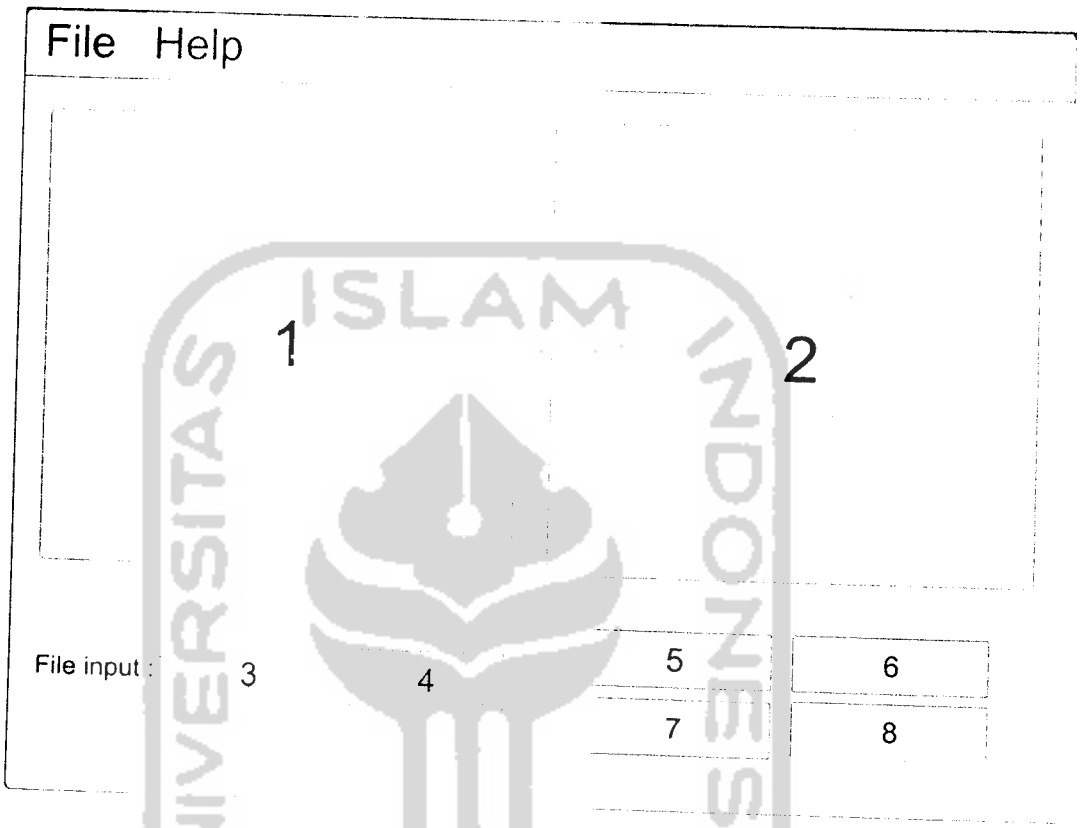
Gambar 3.8 *Sequence Diagram clear tree*

3.2.3.4 Perancangan Antarmuka

Perancangan antarmuka menggambarkan desain tampilan dari sistem, ilustrasi dari rancangan antarmuka terhadap sistem yang akan diaplikasikan. Desain antarmuka yang utama ditujukan kepada *user*, dimana antarmuka didesain sedemikian rupa untuk memudahkan penggunaan sistem aplikasi ini.

3.2.3.4.1 Rancangan antarmuka tampilan utama

Antarmuka ini merupakan tampilan utama ketika aplikasi dijalankan. Di dalamnya terdapat menu *toolbars* yang berada di bagian bawah pada tampilan. Rancangan tampilan utama dapat dilihat pada gambar 3.9



Gambar 3.9 Rancangan antarmuka tampilan utama

Dari gambar diatas terdapat panel dan tombol-tombol yang digunakan di dalam sistem guna menjalankan proses-proses yang dibutuhkan. Tampilan-tampilan tersebut antara lain adalah:

1. Panel tabel data masukan

   Di dalam panel tersebut akan ditampilkan isi dari data masukan yang telah dipilih oleh user untuk diproses.

2. Panel pohon keputusan

   Pada panel ini, akan ditampilkan hasil akhir dari sistem yang merupakan sebuah pohon keputusan dari data masukan di sebelah kiri panel.

3. Kolom alamat data masukan

   Di dalam kolom ini akan ditampilkan alamat/lokasi dari data masukan berada.

4. Tombol browse

   Digunakan untuk memilih data yang akan dijadikan masukan untuk sistem ini.

5. Tombol clear table

   Dengan menekan tombol ini user dapat menghilangkan isi dari panel tabel data masukan.

6. Tombol clear tree

   User akan menghilangkan pohon keputusan dari sistem dengan cara menekan tombol ini.

7. Tombol process data

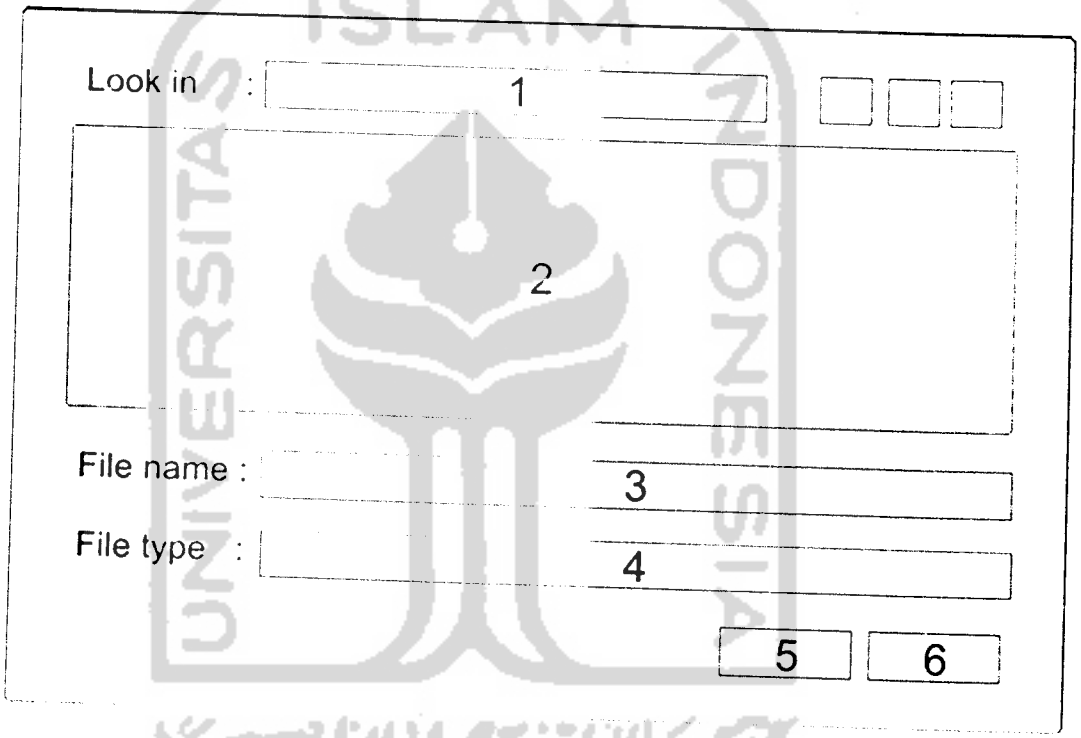
   Setelah data masukan untuk sistem telah dipilih, user akan menekan tombol ini untuk melakukan kalkulasi terhadap data masukan, kemudian sistem akan menyimpan hasil dari kalkulasi dan menunggu untuk dipanggil dengan menekan tombol create tree

8. Tombol create tree

   Setelah proses dilakukan maka sistem akan memanggil nilai tersebut untuk ditampilkan ke dalam panel pohon keputusan dengan cara menekan tombol create tree.

3.2.3.4.2 Rancangan antarmuka menu memilih file

Antarmuka memilih file digunakan untuk memilih file yang akan dijadikan sebagai masukan dari aplikasi. Rancangan antarmuka menu memilih file dapat dilihat pada gambar 3.10



Gambar 3.10 Rancangan antarmuka pemilihan file

Keterangan dari hasil rancangan diatas adalah sebagai berikut:

1. Kolom alamat folder data

   Di dalam kolom ini akan ditampilkan nama dari folder tempat data masukan berada.

2. Kolom isi folder

Kolom tersebut akan menampilkan isi dari folder yang tertera pada kolom alamat folder data.

3. Kolom file name

   Nama dari data masukan yang dipilih akan terlihat pada kolom ini.

4. Kolom file type

   Jenis data masukan yang boleh digunakan sebagai masukan akan tertera pada kolom ini, format lain dari data masukan dapat diperlebar dengan menekan kolom ini.

5. Tombol Ok

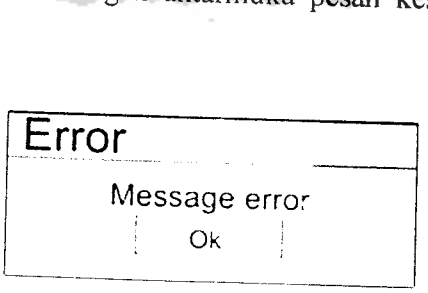
   Tombol untuk memilih data yang diinginkan

6. Tombol Cancel

   Untuk membatalkan pilihan data

3.2.3.4.3 Rancangan antarmuka pesan kesalahan

Antarmuka untuk tampilan penanganan kesalahan akan menampilkan kesalahan yang ditemukan. Rancangan antarmuka pesan kesalahan dapat dilihat pada gambar 3.11

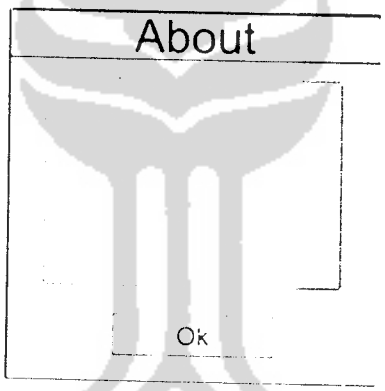


Gambar 3.11 Rancangan antarmuka pesan kesalahan

Tampilan pesan kesalahan akan terlihat seperti gambar 3.12. Dengan memperhatikan jenis kesalahan yang tertera pada interface tersebut user dapat memahami jenis kesalahan yang telah dilakukan. Untuk menghilangkan pesan kesalahan user dapat menekan tombol yang berada di tengah interface.

3.2.3.4.4 Rancangan antarmuka Menu About

Antarmuka menampilkan informasi tentang aplikasi. Rancangan antarmuka menu *about* dapat dilihat pada gambar 3.12



Gambar 3.12 Rancangan antarmuka menu *About*

Interface dari menu about akan memberikan informasi tentang software yang digunakan. Keterangan dari software terdapat pada kolom di tengah interface.

BAB IV
HASIL dan PEMBAHASAN

4.1 Hasil

4.1.1 Implementasi Sistem Datamining Secara Umum

Implementasi sistem adalah tahap dimana sistem mampu diterapkan ke dalam keadaan yang sesungguhnya, setelah perancangan yang sudah dilakukan sebelumnya. Dari implementasi, akan diketahui keadaan sistem yang telah dibuat, apakah dapat berjalan dengan baik atau tidak, menghasilkan keluaran sistem yang sesuai dengan perancangan yang telah dibuat.

4.1.2 Implementasi Struktur Sistem Datamining

Implementasi dari perancangan sistem datamining adalah diimplementasikannya rancangan *class diagram* ke dalam bahasa pemrograman java. Di dalam sistem terdapat tujuh buah *class* yang digunakan. Semuanya terdapat dalam satu paket yaitu paket *treesoft*. Implementasi dari rancangan sistem adalah sebagai berikut:

a. *Class Datamining* adalah sebagai *class* utama dari sistem datamining.
b. *Class Filter* adalah *class* sebagai kelas penyaring data masukan sistem datamining.

c. *Class Utils* adalah *class* yang digunakan untuk menentukan jenis data yang dapat digunakan dalam datamining.

d. *Class dataTable* adalah *class* yang berfungsi untuk menampilkan isi dari data masukan ke dalam tabel yang berada pada lingkungan sistem.

e. *Class ID3* adalah *class* yang dibuat untuk menghitung data masukan sehingga menghasilkan sebuah keluaran yaitu sebuah *decision tree*.

f. *Class Nodes* adalah *class* untuk membuat node-node tree dari pohon keputusan.

g. *Class abstractTreeModel* adalah *class* yang digunakan untuk menampilkan hasil dari keluaran ke dalam bentuk pohon yang ada di dalam lingkungan sistem.

4.1.3 Implementasi Antarmuka Aplikasi Datamining

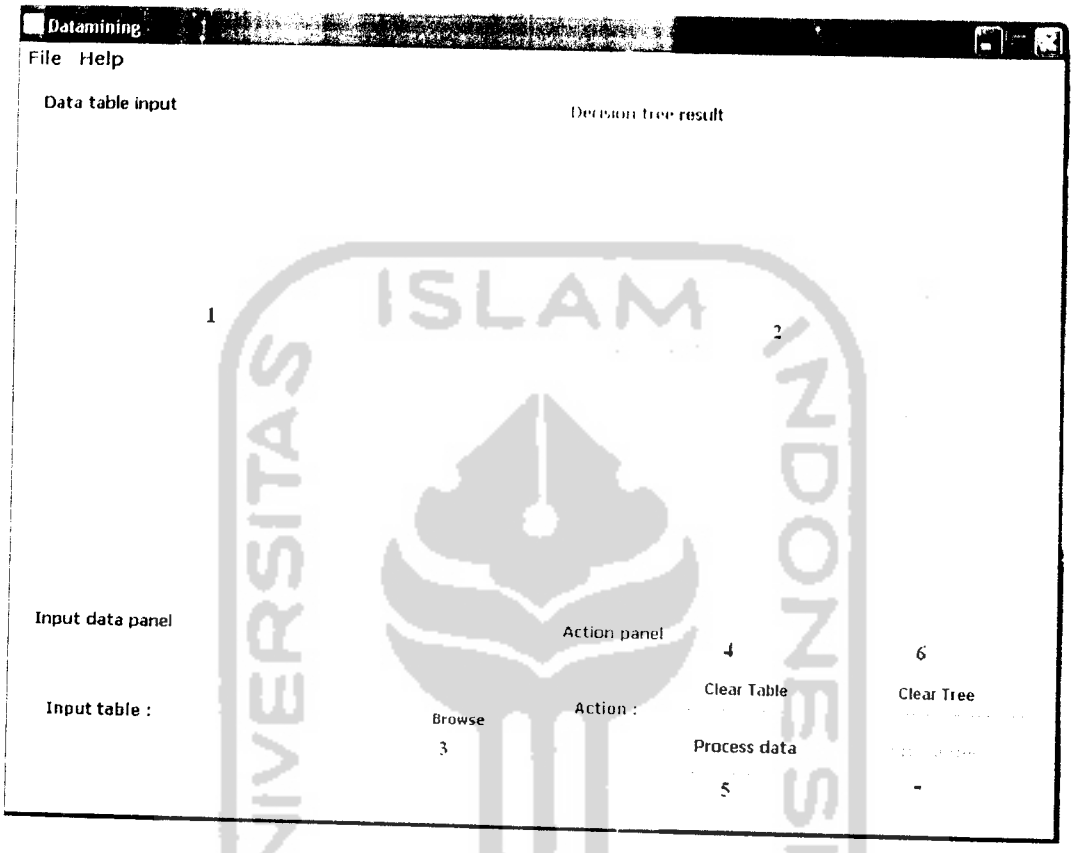
Secara umum implementasi antar muka sistem proses datamining dipresentasikan pada *class Datamining*. Semua fungsi yang dibutuhkan oleh sistem terdapat pada *class* utama ini, karena semua aktivitas sistem akan berada dalam lingkungan antar muka ini.

Tampilan halaman utama

Halaman utama adalah halaman tempat user dan sistem berinteraksi. Pada halaman ini terdapat dua buah menu yang dapat digunakan oleh user yaitu:

a. Menu *File* yang digunakan untuk melakukan proses input data masukan dari sistem, serta sub menu untuk keluar dari sistem.

b. Menu *Help* berfungsi untuk menampilkan menu bantuan yang berisi tentang petunjuk pemakaian program.



Gambar 4.1 Tampilan Utama

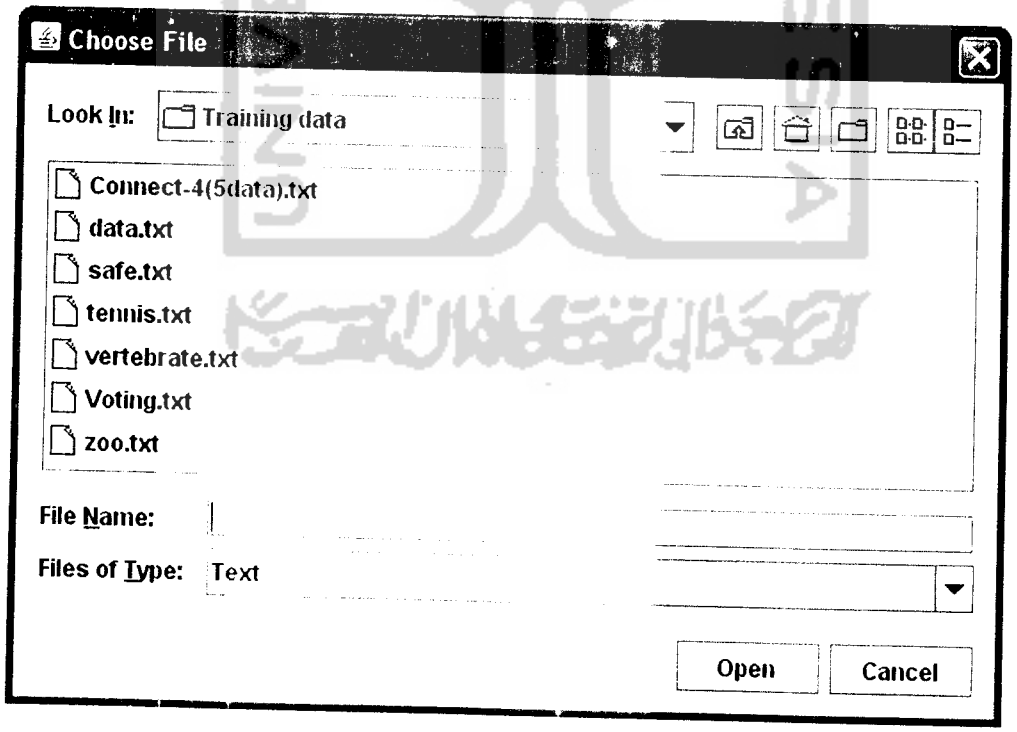
Fungsi-fungsi lain yang berada pada lingkungan sistem adalah:

1. Kolom input tempat untuk data tabel akan ditampilkan
2. Kolom pohon keputusan adalah tempat untuk hasil dari proses akan ditampilkan
3. Tombol *browse* digunakan untuk memilih data masukan.
4. Tombol *clear table* untuk menghapus isi dari tabel yang berada pada lingkungan sistem.

5. Tombol *process data* berfungsi untuk memulai proses kalkulasi dari data masukan yang telah dipilih.
6. Tombol *create tree* akan menampilkan hasil dari kalkulasi data masukan tersebut yang berupa sebuah pohon keputusan.
7. Tombol *clear tree* akan menghapus isi dari pohon keputusan di lingkungan sistem.

Tampilan halaman pemilihan data masukan

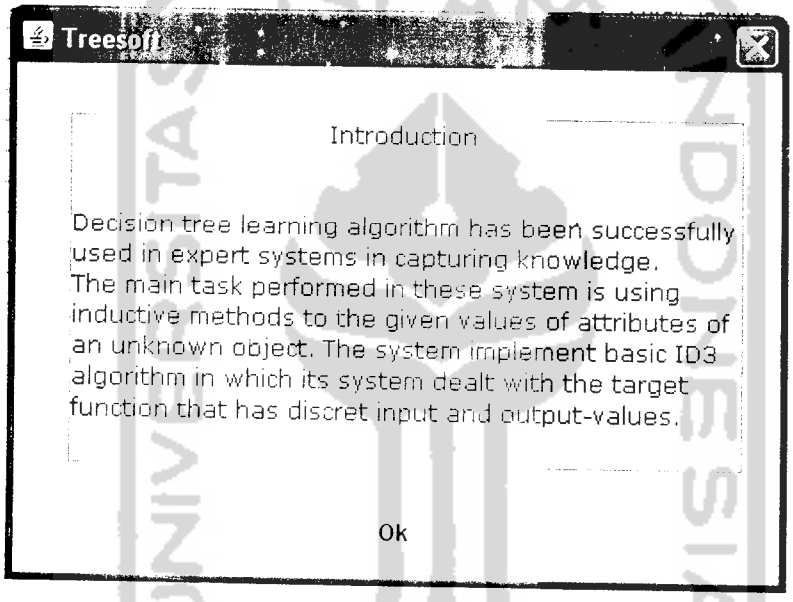
Di dalam tampilan ini user akan dihadapkan pada tampilan yang mengarah pada pemilihan data masukan dari sistem. Lebih jelasnya dapat dilihat pada gambar 4.2



Gambar 4.2 Pemilihan file

Tampilan halaman pengenalan/Introduction

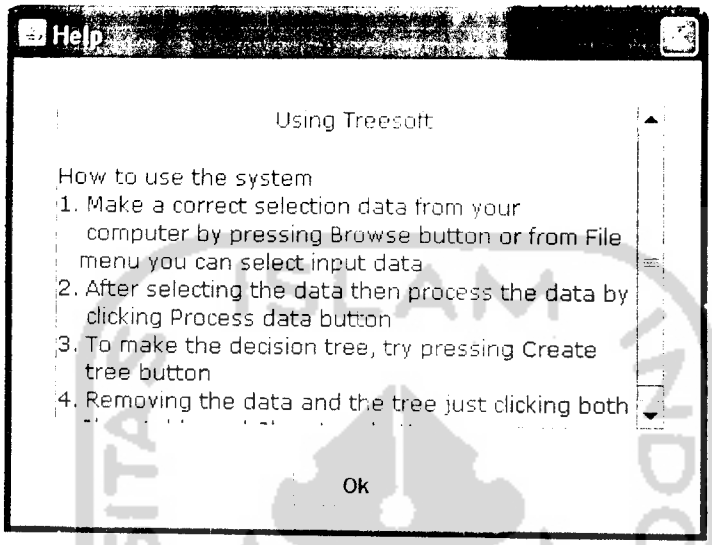
Halaman ini berisi tentang ringkasan dari sistem yang dibangun. Menjelaskan sedikit tentang penggunaan metode yang digunakan oleh sistem. Jelasnya dapat dilihat pada gambar 4.3



Gambar 4.3 Tampilan pengenalan/Introduction

Tampilan halaman bantuan/Help

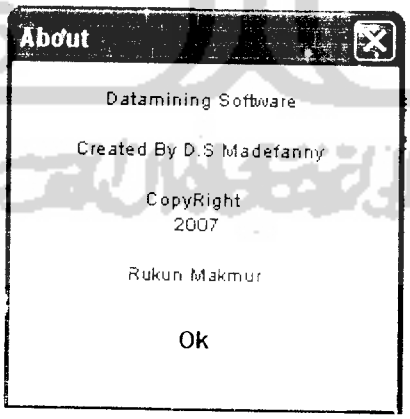
Adalah halaman yang memberikan penjelasan secara singkat tentang cara pemakaian dari sistem yang akan digunakan. Jelasnya dapat dilihat pada gambar 4.4



Gambar 4.4 Tampilan Help/Bantuan

Tampilan halaman tentang program/About

Berisi tentang penjelasan pembuat aplikasi sistem datamining menggunakan algoritma ID3. jelasnya dapat dilihat pada gambar 4.5



Gambar 4.5 Tampilan About

Implementasi Prosedural

Implementasi procedural dari aplikasi ini dikelompokkan ke dalam kelas-kelas seperti yang telah dijelaskan pada bagian perancangan yaitu:

1. *Class Main*

Metode-metode yang digunakan untuk pembuatan pohon keputusan pada kelas ini adalah sebagai berikut:

a. Metode untuk melakukan proses input data

   Nama metode : browseButtonActionPerformed atau inputActionPerformed

   Jenis : private

b. Metode untuk memproses kalkulasi data

   Nama metode : processButtonActionPerformed

   Jenis : private

c. Metode untuk membuat pohon keputusan

   Nama metode : createTreeButtonActionPerformed

   Jenis : private

2. *Class ID3*

Metode-metodenya antara lain adalah sebagai berikut:

a. Metode untuk mengecek atribut apakah telah digunakan untuk proses kalkulasi

   Nama metode : alreadyUsedToDecompose

   Jenis : public

b. Metode untuk menghitung entropy

Nama metode : calculateEntropy

Jenis : public

c. Metode untuk menghitung data masukan dan membuat pohon keputusan

Nama metode : createDecisionTree

Jenis : public

d. Metode untuk membuat object

Nama metode : createObject

Jenis : private

e. Metode untuk membentuk pohon keputusan

Nama metode : createTree

Jenis : private

f. Metode untuk menghitung kembali node

Nama metode : decomposeNode

Jenis : public

g. Metode untuk mendapatkan semua nilai atribut

Nama metode : getAllValues

Jenis : public

h. Metode untuk mendapatkan nilai dari atribut

Nama metode : getSubset

Jenis : public

i. Metode untuk mendapatkan nilai dari atribut

Nama metode : getSymbolValue

Jenis : public

j. Metode untuk menciptakan object

Nama metode : makeObject

Jenis : private

3. *Class dataTable*

Metode-metode yang terdapat dalam kelas ini adalah:

a. Metode untuk menghitung kolom

Nama metode : getColumnCount

Jenis : public

b. Metode untuk mendapatkan nama kolom

Nama metode : getColumnName

Jenis : public

c. Metode untuk menghitung baris

Nama metode : getRowCount

Jenis : public

d. Metode untuk mendapatkan nilai kolom X dari baris ke Y

Nama metode : getValueAt

Jenis : public

4. *Class Nodes*

Metode yang ada di dalamnya adalah:

a. Metode untuk mendapatkan cabang

   Nama metode : getChildAt

   Jenis : public

b. Metode untuk menghitung cabang

   Nama metode : getChildCount

   Jenis : public

c. Metode untuk mendapatkan nilai index cabang

   Nama metode : getIndexOfChild

   Jenis : public

d. Metode untuk mendapatkan nama titik

   Nama metode : getName

   Jenis : public

e. Metode untuk mendapatkan nama cabang sebelumnya

   Nama metode : getParent

   Jenis : public

f. Metode untuk merubah nilai object menjadi string

   Nama metode : toString

   Jenis : public

5. *Class abstractTreeModel*

   Metode-metode yang ada pada kelas ini adalah:

a. Metode untuk mendapatkan cabang

Nama metode : getChild

Jenis : public

b. Metode untuk menghitung anak cabang

Nama metode : getChildCount

Jenis : public

c. Metode untuk mendapatkan nilai index dari cabang

Nama metode : getIndexOfChild

Jenis : public

d. Metode untuk mendapatkan titik akar

Nama metode : getRoot

Jenis : public

e. Metode untuk memvalidasi nilai untuk cabang

Nama metode : isLeaf

Jenis : public

6. *Class Filter*

Metode yang dimilikinya adalah:

a. Metode untuk memvalidasi dari file

Nama metode : accept

Jenis : public

b. Metode untuk mendapatkan nama isi saringan

Nama metode : getDescription

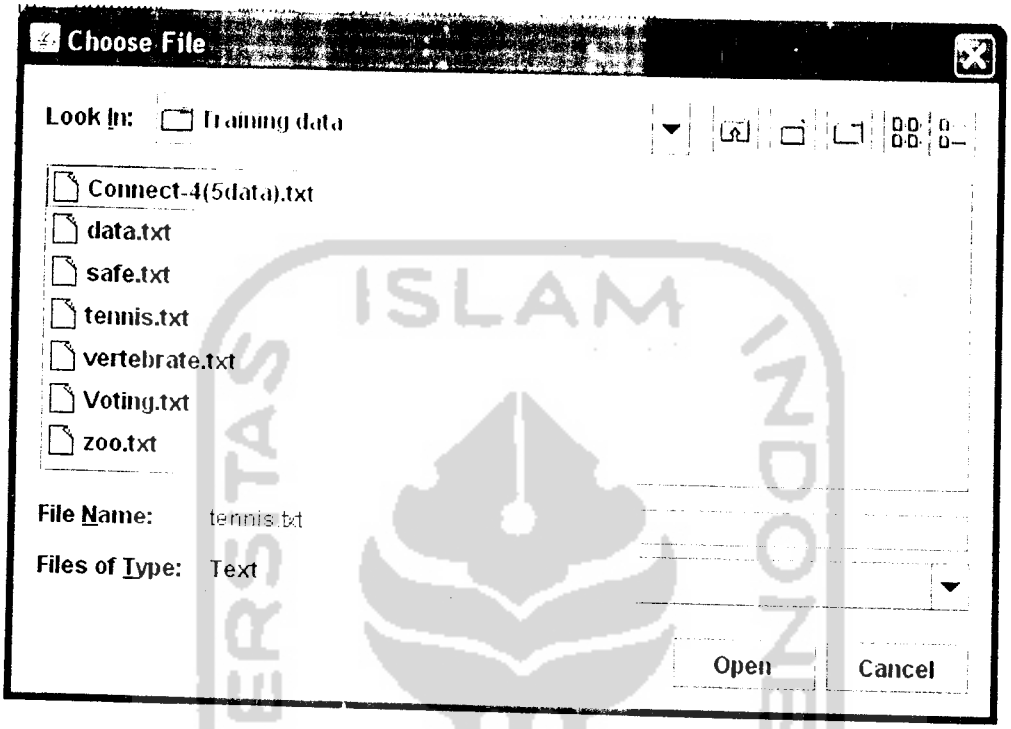
Jenis : public

4.2 Pembahasan

Pengujian disini dilakukan dengan cara melakukan input data (data text) ke dalam sistem, yang kemudian akan ditampilkan ke dalam bentuk tabel. Karena masukan yang akan diproses oleh sistem berupa tabel.

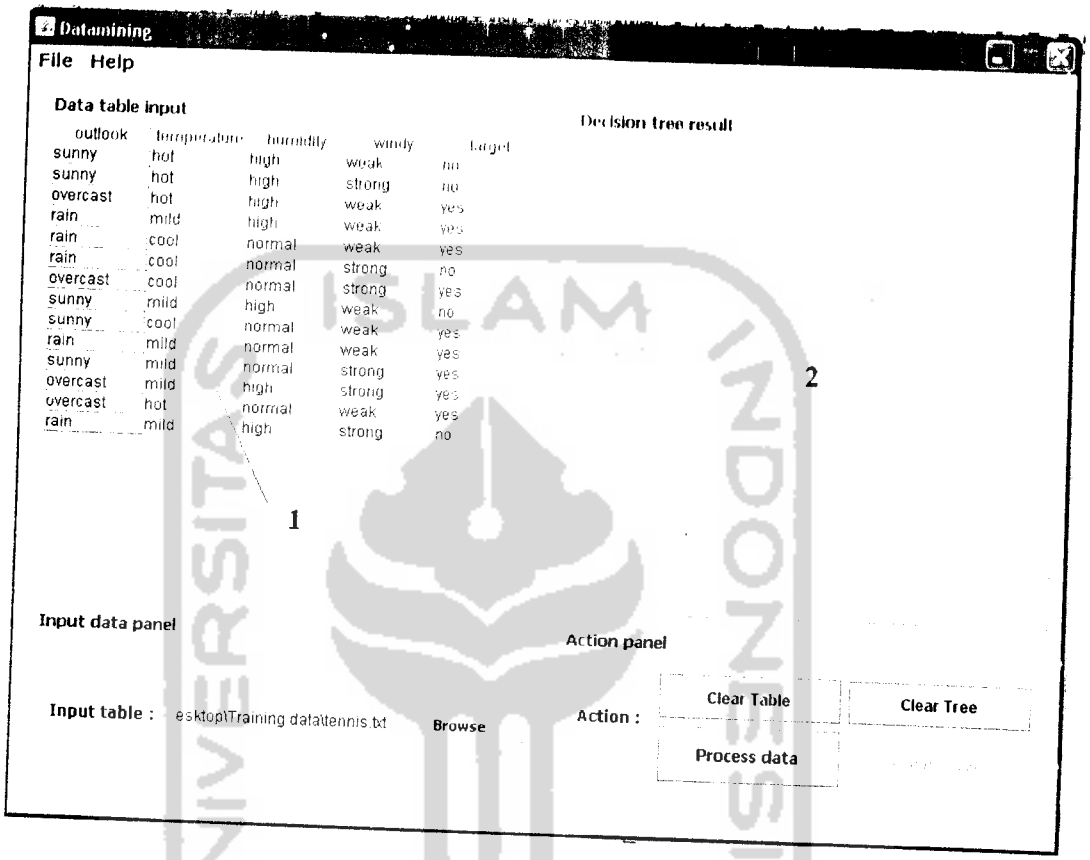
Pengujian dilakukan dengan memasukkan input data dengan format data text pada sistem. Data input yang dimasukkan harus mempunyai aturan yang sudah ditetapkan pada rancangan yang telah dibuat, supaya sistem mampu untuk memproses data tersebut.

Sebagai contoh dilakukan pengujian dengan memasukkan data text dengan nama filenya "tennis.txt" . Untuk lebih jelasnya dapat dilihat pada contoh gambar 4.6



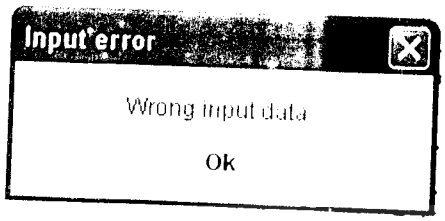
Gambar 4.6 Tampilan pemilihan file

Setelah data input dipilih, kemudian data akan dimunculkan ke dalam sebuah tabel yang berada pada sisi kiri antar muka. Untuk lebih jelasnya dapat dilihat pada gambar 4.7.



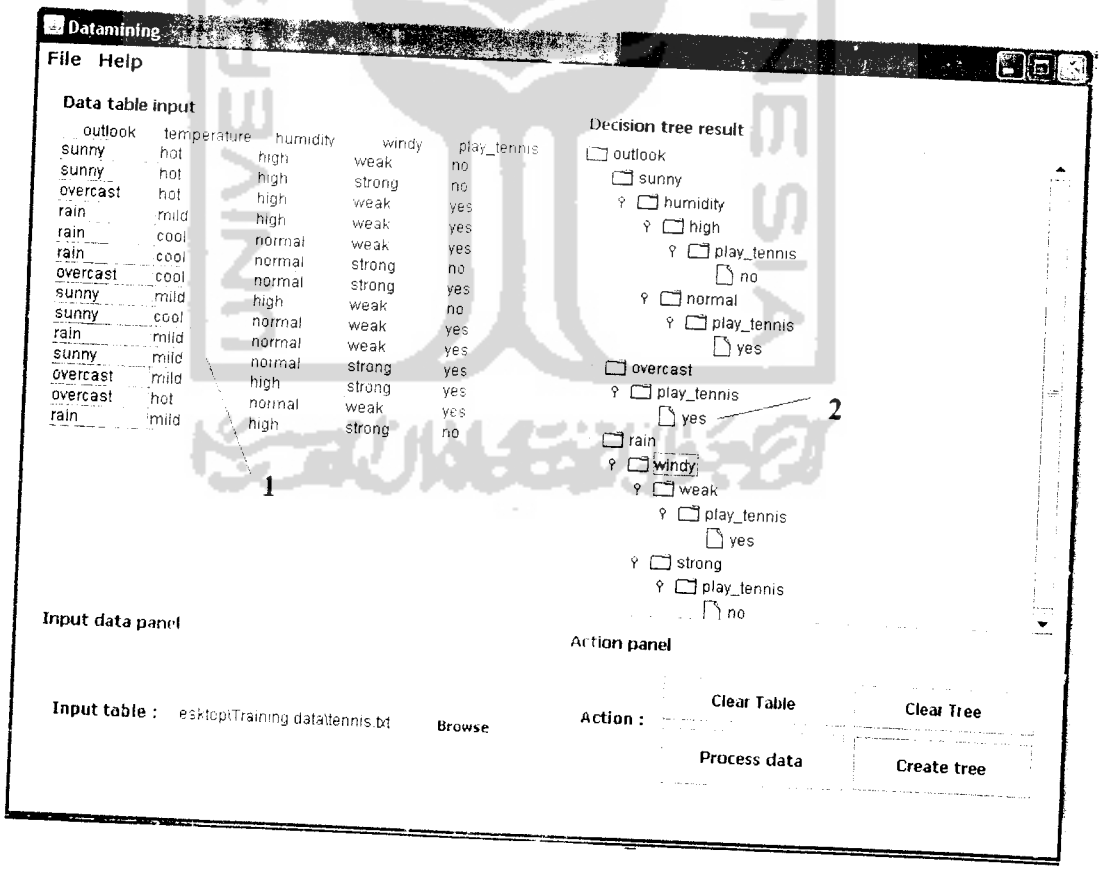
Gambar 4.7 Tampilan penampilan data

Setelah data masukan telah dipilih kemudian akan ditampilkan pada kolom no 1. Masukan yang akan diproses oleh sistem adalah masukan dengan minimal satu kolom sebagai kolom atribut masukan dan satu kolom sebagai kolom atribut target. Kolom atribut target harus berada pada posisi akhir (sebelah kanan) pada tabel. Apabila tidak berada pada posisinya, kolom atribut target dapat disesuaikan. Apabila masukan belum ditentukan maka akan muncul kotak kesalahan yang akan memberikan pesan. Untuk lebih jelasnya dapat dilihat pada gambar 4.8.



Gambar 4.8 Tampilan kesalahan pesan

Setelah user menekan tombol proses, maka user akan mendapatkan hasil pembelajaran berupa pohon keputusan (*decision tree*) yang akan ditampilkan di kolom no 2 yang berada pada sebelah kanan dari antar muka sistem. Untuk lebih jelasnya dapat dilihat pada gambar 4.9.



Gambar 4.9 Tampilan *decision tree*

Hasil dari proses kalkulasi data masukan akan terlihat seperti kolom no 2 sebelah kanan dari form utama. Dari data masukan akan dihasilkan sebuah pohon keputusan yang akan membantu pengguna dalam melihat data yang berada pada kolom no 1 di sebelah kiri pada form utama. Dengan pohon keputusan yang ditampilkan diatas, maka pengguna dapat dengan mudah mengerti isi dari data masukan dan dapat memperoleh informasi dari data masukan yang telah dipilih seperti diatas.

Pohon keputusan yang akan dibentuk dapat berasal dari berbagai jenis data yang sudah mengikuti semua aturan atau syarat sebagai data masukan, sehingga aplikasi dapat menghitung sesuai dengan prosedur yang ada.

BAB V

SIMPULAN dan SARAN

5.1 Simpulan

Kesimpulan yang dapat diambil dari pengembangan sistem datamining menggunakan algoritma ID3 yaitu:

a. Induksi algoritma ID3 adalah sebuah konsep pembelajaran yang didefinisikan oleh atribut dengan nilai-nilai diskret.
b. Algoritma ID3 mengambil kesimpulan sebuah pohon keputusan dengan membuatnya dari atas ke bawah, *breadth-first* (algoritma penelusuran pohon yang dimulai dari titik akar yang kemudian menelusuri ke titik daunnya), dan menggunakan algoritma *greedy* dalam memperoleh *information gain* yang terbaik.
c. Kesimpulan sederhana dari induksi algoritma ID3 adalah pohon keputusan yang lebih pendek dengan nilai yang tinggi dari *information gain* atribut lebih dekat ke akar keputusan.
d. Overfitting (model statistik mempunyai parameter yang terlalu banyak) adalah sebuah masalah untuk induksi pohon keputusan, dapat disebabkan karena adanya noise pada training data, atau oleh training data yang kecil.
e. Algoritma ID3 akan memakan waktu yang lama untuk data yang besar.

f. User akan lebih mudah dalam melihat informasi dari data yang dimasukkan berdasarkan dari pohon keputusan yang dihasilkan.

g. Algoritma ID3 yang digunakan terbatas pada data yang bersifat diskret, untuk data yang bersifat kontinyu maka data harus diklasifikasikan terlebih dahulu.

h. Atribut masukan yang banyak akan membuat pohon keputusan yang dibuat akan menimbulkan banyak cabang dari pohon keputusan yang dibuat, variabel dari atribut masukan juga akan sangat berpengaruh karena setiap nilai dari variable atribut masukan akan dikalkulasikan juga dalam pembuatan pohon.

5.2 Saran

Beberapa saran untuk pengembangan sistem datamining berikutnya adalah sebagai berikut:

a. Tipe masukan untuk proses pembelajaran diperbanyak dari penelitian yang sudah ada, karena semakin banyak tipe masukan yang diperoleh, hasil yang akan diperoleh juga akan bervariasi.

b. Hasil dari pohon keputusan dibuat agar lebih *user friendly* agar lebih mudah dipahami oleh pengguna.

c. Fitur-fitur untuk sistem semakin diperbanyak untuk mendukung hasil yang lebih akurat.

d. Tata letak komponen antarmuka lebih dipermudah, serta disederhanakan untuk kemudahan *user*.

DAFTAR PUSTAKA

[COL07] Colton, Simon. 2004. Decision Tree Learning available at http://www.doc.ic.ac.uk/~sgc/teaching/v231/lecture11.html

[DAN97] Dankel, Douglas. 1997. The ID3 Algorithm available at http://www.cise.ufl.edu/~ddd cap6635/Fall-97/Short-papers/2.htm

[DHA03] Dharwiyanti, Sri & Wahono,Romi Satria, 2003. Pengantar Unified Modeling Language (UML) available at http://www.ilmukomputer.com

[JAS07] Jason Frand, 2007. Datamining available at http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.htm

[KAD04] Kadir, Abdul. 2004. Dasar Pemrograman Java. Yogyakarta : Penerbit ANDI

[SRI07] Sri Hartati,G. Herry Suharto,B. Soesilo Wiyono,M. 2007. Pemrograman GUI Swing Java dengan NetBeans 5. Yogyakarta : Penerbit ANDI

[PAL07] Palace, Bill. 2007. An Introduction to Datamining available at http://www.thearling.com/text/dmwhite/dmwhite.htm

Lampiran

1. Data: 1984 United States Congressional Voting Records Database.

   Jumlah data: 435

   Jumlah attribute: 16

   Informasi atribut:

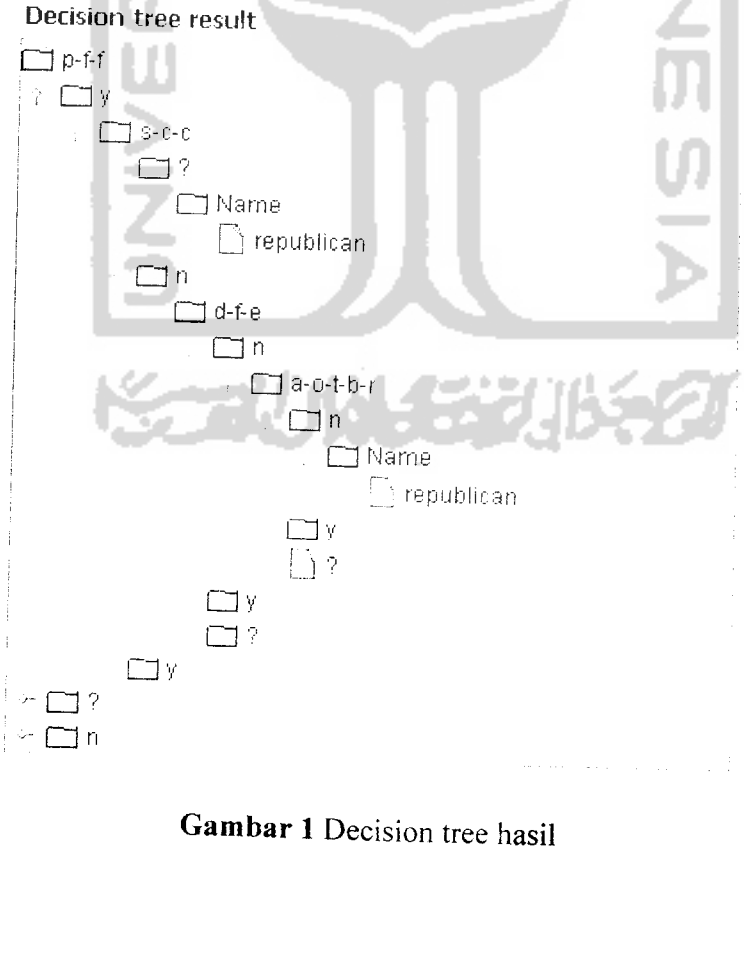
1. Class Name(target): 2 (democrat, republican) -> Name
2. handicapped-infants: 2 (y,n) -> hd-in
3. water-project-cost-sharing: 2 (y,n) -> w-pr-c-sh
4. adoption-of-the-budget-resolution: 2 (y,n) -> a-o-t-b-r
5. physician-fee-freeze: 2 (y,n) -> p-f-f
6. el-salvador-aid: 2 (y,n) -> e-a
7. religious-groups-in-schools: 2 (y,n) -> r-g-s
8. anti-satellite-test-ban: 2 (y,n) -> a-s-t-b
9. aid-to-nicaraguan-contras: 2 (y,n) -> a-n-c
10. mx-missile: 2 (y,n) -> m-m
11. immigration: 2 (y,n) -> imi
12. synfuels-corporation-cutback: 2 (y,n) -> s-c-c
13. education-spending: 2 (y,n) -> e-s
14. superfund-right-to-sue: 2 (y,n) -> s-r-s
15. crime: 2 (y,n) -> cri
16. duty-free-exports: 2 (y,n) -> d-f-e

17. export-administration-act-south-africa: 2 (y,n) -> e-a-sa

Potongan isi dari data

republican n y n y y y n n n y ? y y y n y
republican n y n y y y n n n n n y y y n ?
democrat ? y y ? y y n n n n y n y y n n
democrat n y y n ? y n n n n y n y n n y
democrat y y y n y y n n n n y ? y y y y
democrat n y y n y y n n n n n n y y y y
democrat n y n y y y n n n n n n ? y y y
republican n y n y y y n n n n n n n y y ? y
republican n y n y y y n n n n n n y y y n y
democrat y y y n n n y y y n n n n n ? ?

Hasil dari pengujian dari data diatas terbentuk sebuah decision tree seperti gambar 1



Gambar 1 Decision tree hasil

Dari data diatas dapat ditentukan partai mana yang akan dipilih berdasarkan atribut yang ada. Sebagai root nodenya adalah atribut physician-fee-freeze yang menjadi atribut penentu keputusan, selanjutnya diikuti oleh variabel yang dimiliki oleh atribut tersebut. Cabang dari decision tree selanjutnya adalah synfuels-corporation-cutback dengan variabel ? maka dapat ditentukan bahwa partai yang dipilih adalah republican.

2. Data: Connect-4 opening database

   Informasi tentang Connect-4:

   Database ini berisi mengenai gerakan yang boleh dilakukan di dalam game connect-4 yang dimainkan oleh dua orang. Data tersebut berisi mengenai langkah-langkah yang dilakukan oleh para pemain.

   This database contains all legal 8-ply positions in the game of connect-4 in which neither player has won yet, and in which the next move is not forced.

   Jumlah data: 67557 (data asli) jumlah dikurangi untuk memperpendek waktu kalkulasi menjadi ±300 data.

   Jumlah atribut: 42.

   Informasi atribut: (x=player x has taken, o=player o has taken, b=blank)

   The board is numbered like

```
6 . . . . . .
5 . . . . . .
4 . . . . . .
3 . . . . . .
2 . . . . . .
1 . . . . . .
  a b c d e f g
```

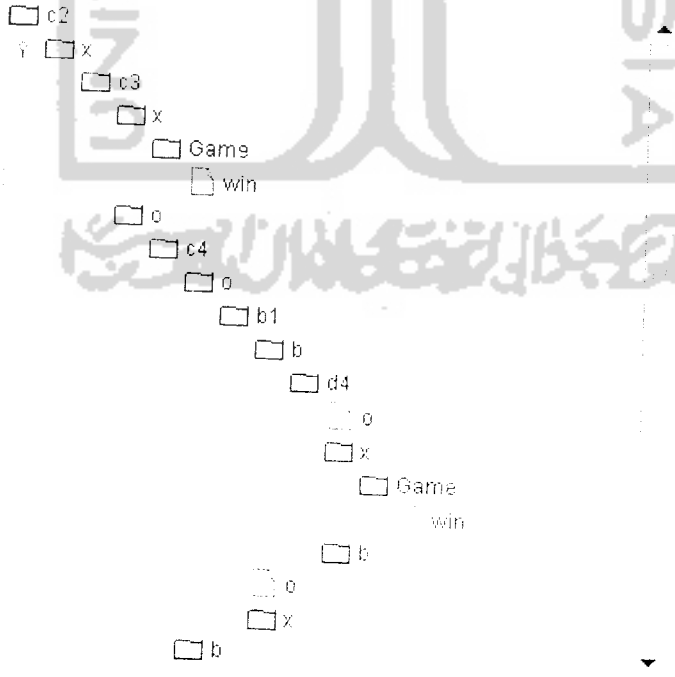
1. a1: {x,o,b}
2. a2: {x,o,b}
3. a3: {x,o,b}
4. a4: {x,o,b}
5. a5: {x,o,b}
6. a6: {x,o,b}
7. b1: {x,o,b}
8. b2: {x,o,b}
9. b3: {x,o,b}
10. b4: {x,o,b}
11. b5: {x,o,b}
12. b6: {x,o,b}
13. c1: {x,o,b}
14. c2: {x,o,b}
15. c3: {x,o,b}
16. c4: {x,o,b}
17. c5: {x,o,b}
18. c6: {x,o,b}
19. d1: {x,o,b}
20. d2: {x,o,b}
21. d3: {x,o,b}
22. d4: {x,o,b}
23. d5: {x,o,b}
24. d6: {x,o,b}
25. e1: {x,o,b}
26. e2: {x,o,b}
27. e3: {x,o,b}
28. e4: {x,o,b}
29. e5: {x,o,b}
30. e6: {x,o,b}
31. f1: {x,o,b}
32. f2: {x,o,b}
33. f3: {x,o,b}
34. f4: {x,o,b}
35. f5: {x,o,b}
36. f6: {x,o,b}
37. g1: {x,o,b}
38. g2: {x,o,b}
39. g3: {x,o,b}
40. g4: {x,o,b}
41. g5: {x,o,b}
42. g6: {x,o,b}

```
43. Class: {win,loss,draw}
```

Potongan isi dari data Connect-4

b b b b b b b b b b b o x x o b b x o x o b b b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x x b b b x o x o b b o b b b b b b b b b b b b b b b b win
b b b b b b o b b b b b o x x b b b x o x o b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x x b b b x o x o b b b b b b b b b b b b b b b b b b win
o b b b b b b b b b b o x x b b b x o x o b b b b b b b b o b b b b b b b b b win
b b b b b b b b b b b o x x b b b x o x o b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x x b b b x o x o b b b b b b b b b b b b o b b b b b win
b b b b b b b b b b b o x o o b b x o x x b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x o x b b x o x o b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x o x o b x o x b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x o x b b x o x b b b o b b b b b b b b b b b b b b win
b b b b b o b b b b b o x o x b b x o x b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x o x b b x o x b b b b b b b b b b b b b b b b b b win
o b b b b b b b b b b o x o x b b x o x b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x o x b b x o x b b b b b b b b b b b b b b b b b b win
b b b b b b b b b b b o x o x b b x o x b b b b b b b b b b b b o b b b b b win
b b b b b b b b b b b o x o o b b x o x b b b x b b b b b b b b b b b b b b loss

Hasil dari pengujian dari data diatas terbentuk sebuah decision tree seperti gambar 2



Gambar 2 Decision tree hasil

Dari data diatas, didapatkan suatu informasi mengenai langkah-langkah yang telah dilakukan dan didapatkan pola sesuai dengan data yang dimasukkan. Semakin banyak data yang dimasukkan akan semakin banyak informasi yang akan diperoleh. Decision tree yang diperoleh akan berbeda-beda tergantung dari informasi yang akan diambil berdasarkan data masukan.

Penjelasan dari decision tree diatas adalah sebagai berikut:

Jika titik C2 di isi oleh pemain X dan titik C3 diisi oleh X maka game dimenangkan oleh X

Jika titik C3 di isi oleh pemain O dan titik C4 diisi oleh O dan B1 kosong dan D4 diisi oleh X maka game dimenangkan oleh X

Pola tersebut akan terbentuk setelah langkah-langkah sudah dilakukan, dan posisi dari setiap pemain berada pada posisi tersebut.