BAB 2
TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Beberapa peneliti terdahulu telah melakukan penelitian yang terkait dengan penelitian ini khusunya tentang data mining dan sistem pakar dengan menggunakan algoritma dan pendekatan yang ada, diantaranya yaitu algoritma C4.5 dan pendekatan *case based reasoning*. Contoh dari penelitian-penelitian yang terkait adalah sebagai berikut:

1. Penelitian yang berjudul "Sistem Penentuan Tingkat Kesejahteraan Anak Menggunakan Algoritma C4.5" oleh Yuli Murdianingsih (2014), permasalahan yang dibahas pada penelitian ini adalah tentang belum adanya sistem yang mampu mendeskripsikan tingkat pencapaian kesejahteraan anak. Metode yang digunakan adalah algoritma C4.5 dengan atribut yang digunakan meliputi fisik, intelektual, emosional, dan sosial spiritual. Masing-masing atribut tersebut kemudian dibuat kedalam tiga kategori yaitu kategori Memadai, kategori Cukup, dan kategori Tidak Memadai, hal ini dimaksudkan agar lebih mudah pada saat pembuatan pohon keputusan. Data-data yang digunakan berupa data primer yakni data yang didapatkan dari hasil wawancara langsung dengan para responden melalui sebuah kuesioner. Keseluruhan data berjumlah 212 dan 176 diantaranya digunakan sebagai data training, sisanya digunakan sebagai data testing. Pengolahan data dilakukan menggunakan RapidMiner, dari hasil pemodelan menggunakan decision tree algoritma C4.5 maka diperoleh hasil akurasi yang sangat tinggi yaitu sebesar 95,65%. [3]
2. Penelitian yang dilakukan oleh Arief Jananto (2013) yang mengangkat judul "Algoritma Naïve Bayes untuk Mencari Perkiraan Waktu Studi Mahasiswa"

didasari tentang bagaimana melakukan prediksi terhadap lama masa studi mahasiswa dengan menggunakan teknik data mining. Algoritma yang dipilih adalah algoritma naïve bayes, data yang terkumpul yakni data mahasiswa yang sudah dinyatakan lulus periode 2004 sampai dengan 2007 berjumlah 266 record dengan mengambil atribut berupa indeks prestasi semester 1, indeks prestasi semester 2, indeks prestasi semester 3, indeks prestasi semester 4, jenis kelamin, kota lahir, tipe sekolah, dan kota sekolah. Fungsi yang digunakan untuk melakukan prediksi dihasilkan melalui query pada MySQL dalam bentuk function (fbayesian) dengan memakai 2 jenis label, Tepat Waktu dan Tidak Tepat Waktu. Dari hasil pengujian didapatkan tingkat kesalahan prediksi sebesar 20% hingga 50% dan rata-rata tingkat kesalahan sebesar 20% hingga 34% [2].

3. Diki Andita Kusuma dan Chairani (2014) dalam penenelitan yang berjudul "Rancang Bangun Sistem Pakar Pendiagnosa Penyakit Paru-Paru Menggunakan Metode Case Based Reasoning" mengangkat permasalahan tentang mahalnya biaya untuk berkonsultasi ke dokter spesialis paru-paru tentang bagaimana gejala-gejala maupun penyebab dari penyakit paru-paru itu sendiri yang menyebabkan semakin banyaknya penderita penyakit tersebut. Rancang bangun sisem ini ditujukan untuk membantu mendiagnosa gejala awal penyakit paru-paru dengan menggunakan metode Case Based Reasoning (CBR). Untuk mendapatkan kasus-kasus yang memiliki kemiripan, penelitian ini menggunakan algoritma Nearest Neighbor untuk menghitung kedekatan antara kasus baru dengan kasus lama berdasarkan bobot dari sejumlah fitur yang ada. Berdasarkan tingkat kemiripan yang diperoleh, sistem akan mengeluarkan diagnose penyakit yang diderita oleh pasien. Kasus yang digunakan berjumlah 8 kasus dan 1 kasus baru untuk dihitung nilai kedekatanya dengan kasus lama. Nilai kedekatan dari 8 kasus lama terhadap kasus baru dari seorang pasien adalah 0.38 terhadap data kasus pertama, 0.45 terhadap data kasus kedua, 0.56 terhadap data kasus ketiga, 0.56 terhadap data kasus ke empat, 0.72 terhadap data kasus ke lima, 0.93 terhadap data kasus ke

enam, 0.52 terhadap data kasus ke tujuh, dan 0.66 terhadap data kasus ke delapan. Kedekatan paling maximum didapat pada kasus ke enam, yaitu sebesar 0.93 atau 93% sehingga dapat disimpulkan bahwa pasien terdiagnosa penyakit radang paru. [4]

Tabel 2.1 Penelitian Terkait

No	Nama Peneliti dan Tahun	Masalah	Metode	Hasil
1.	Yuli Murdianingsih, 2014	Belum adanya sistem yang dapat digunakan untuk mengelola data sosial (data pemenuhan kesejahteraan) yang semakin membesar di kota Bandung	Algoritma klasifikasi C4.5	Akurasi yang dihasilkan pada penelitian ini menggunakan algoritma C4.5 sangat tinggi yaitu sebesar 95,65%
2.	Arief Jananto, 2013	Bagaimana melakukan prediksi menggunakan teknik data mining terhadap lama masa studi mahasiswa	Algoritma Naïve Bayes	Tingkat kesalahan prediksi sebesar 20% hingga 50% dengan rata-rata tingkat salah sebesar 20% hingga 34%
3.	Diki Andita Kusuma, Chairani, 2014	Mahalnya biaya berkonsultasi ke dokter spesialis paru-	*Case Based Reasoning*	Kedekatan paling maximum didapat pada kasus ke

No	Nama Peneliti dan Tahun	Masalah	Metode	Hasil
		paru untu mengetahui tentang gejala-gejala penyakit paru-paru		enam, yaitu 0.93% atau 93%, mengindikasikan bahwa pasien terdiagnosa terkena penyakit radang paru

Dari ketiga penelitian diatas, maka memungkinkan sistem pakar dan algoritma C4.5 tepat digunakan untuk memprediksi ketepatan waktu kelulusan mahasiswa Universitas Dian Nuswantoro karena memiliki tingkat akurasi yang tinggi.

2.2 Mahasiswa

Salah satu komponen penting dalam pendidikan tinggi atau yang umum disebut dengan *universitas* adalah mahasiswa. Membangun karakter mahasiswa dalam berbagai bidang yang positif sangat diperlukan agar kelak para generasi bangsa tersebut dapat memberikan kontribusi karya serta kerja kerasnya untuk bangsa. Peran *universitas* dalam membentuk mahasiswa handal yang berkompeten dalam bidangnya harus diimbangi dengan keaktifan dan *responsive* dari diri mahasiswa itu sendiri pada saat menjalani kuliah. Unit Kegiatan Mahasiswa juga memiliki andil untuk menumbuhkan semangat organisasi dalam diri mereka sekaligus memberikan pondasi *leadership* yang akan berguna dimasa depan terutama di dunia kerja. Peran lain dari seorang mahasiswa terlihat dalam dunia politik. Mahasiswa menjadi sarana yang mengontrol kebijakan pemerintah melalui penyampaian aspirasi masyarakat tentang sikap pemerintah terhadap suatu kebijakan yang dibuat. Mahasiswa dituntut untuk

peka akan masalah-masalah yang ada disekitarnya, terutama masalah sosial yang sedang menjadi isu-isu publik serta berpikir kritis untuk mencari solusi guna memecahkan masalah yang terjadi.

2.3 Kelulusan Perguruan Tinggi

Menempuh pendidikan di perguruan tinggi tidaklah mudah. Ada banyak beban pedidikan yang harus diambil dan dijalani sebagai mahasiswa dari mulai masuk hingga kemudian menjadi seorang lulusan yang menyandang gelar. Didalam pedidikan tingkat tinggi terdapat 5 (lima) jenjang pendidikan diantaranya Diploma 1 (D1), Diploma 2 (D2), Diploma 3 (D3), Diploma 4 (D4), dan Sarjana (S1). Dari ke lima jenjang tersebut program sarjana merupakan yang paling banyak dipilih dengan prosentase pemilihan sebanyak 83.0%, seperti yang tampak pada grafik dibawah ini [5]:

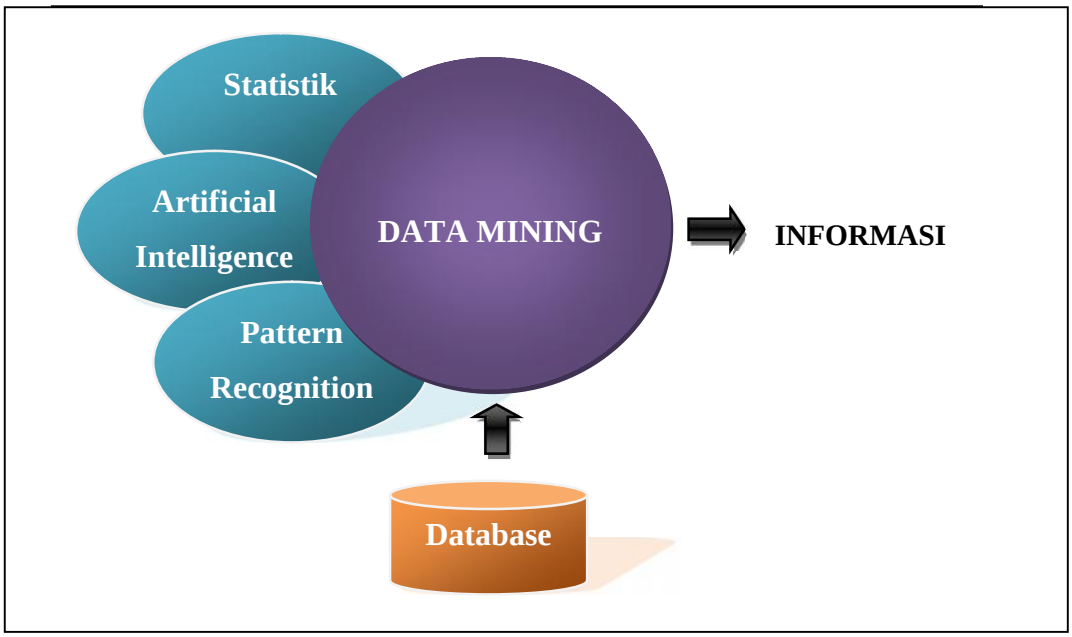
Gambar 2.1 Grafik Perbandingan Jenjang Pendidikan Mahasiswa Aktif

Standar kelulusan merupakan parameter pencapaian mahasiswa dalam menempuh pendidikan tinggi yang meliputi sikap, pengetahuan, dan keterampilan yang dituangkan dalam rumusan pencapaian kelulusan. Dengan standar ini *universitas*

menghasilkan produk yang kreatif dan inovatif serta siap untuk terjun langsung menghadapi dunia kerja. Tingkat materi yang harus dikuasai oleh mahasiswa khusunya program sarjana yaitu minimal telah menguasai konsep teoritis dan skill tertentu secara mendalam.

2.4 Data Mining

Data mining adalah teknik yang memanfaatkan dan menggali hubungan-hubungan yang tersembunyi antar setiap pola dalam data melalui proses ekstrak **[6]**. Data-data yang tertimbun dari waktu ke waktu semakin bertambah jumlahnya, untuk transaksi perhari saja dapat terjadi ratusan transaksi. Data tersebut hanya akan menjadi sampah yang terus memenuhi ruang penyimpanan data bila dibiarkan. Mengingat sebuah perusahaan atau instansi selalu melayani banyak transaksi setiap harinya yang membuat jumlah data yang terkumpul semakin meluap, maka diperlukan suatu ilmu pengetahuan yang mampu menangani data dalam jumlah besar guna menangkap peluang-peluang yang mungkin selama ini belum ditemukan. *Data mining* adalah ilmu yang mampu menangani hal tersebut. Ilmu ini dihasilkan dari 4 (empat) akar bidang ilmu, yaitu **[6]**:



Gambar 2.2 Akar Ilmu Data Mining

1. Statistik

   Bidang ini meliputi ilmu pengetahuan yang mengolah data melalui penghitungan parameter-parameter yang biasanya tersaji dalam bentuk tabel frekuensi. Perhitungan dilakukan menggunakan metode matematis seperti mean, median, rata-rata, dan sebagainya.

2. *Artificial Intelligence* (AI)

   Disiplin ilmu ini mempelajari tentang kecerdasan buatan yang didasarkan pada pola nalar manusia dalam teknik pengolahan informasi.

3. *Pattern Recognition* atau pengenalan pola

   Akar lain dari *data mining* adalah bidang pengenalan pola dimana *data mining* lebih spesifik kepada pengolahan data dari *database* serta menemukan pola asosiasi dan sekunsial.

4. *Database*

   Sistem basis data menyajikan informasi dalam bentuk data-data yang akan diolah menggunakan metode-metode penghitungan tertentu.

2.4.1 Pengelompokan Data Mining

Terdapat beberapa kelompok dalam data mining yang dikelompokkan atas tugas-tugas yang mempu diselesaikan, yaitu :

1. Deskripsi

   Biasanya seorang peneliti maupun analis melakukan suatu percobaan guna mendapatkan cara mendeskripsikan pola serta kecenderungan pada data dimana deskripsi tersebut seringkali memberi kemungkinan penjelasan terhadap suatu pola.

2. Estimasi

   Estimasi mirip dengan klasifikasi, hanya saja yang membedakan adalah variabel target estimasi cenderung mengarah kepada numerik dibandingkan mengarah

kepada kategori. Pada estimasi, record komplit yang menyajikan nilai variabel target untuk nilai prediksi dibutuhkan guna pembangunan model. Kemudian dalam peninjauan setelahnya estimasi nilai variabel target dibentuk dari nilai variabel prediksi.

3. Klasifikasi

   Pada klasifikasi ada sebuah target variable kategori contohnya pengkategorian tinggi badan bisa dikelompokkan menjadi 2 yakni badan tinggi, dan badan pendek.

   a. Klasifikasi Berbasis *Desicion Tree*

   Decision Tree adalah pohon yang berfungsi sebagai prosedur penalaran guna memperoleh jawaban atas permasalahan yang diinputkan [6]. *Desicion tree* banyak diminati dalam berbagai bidang penelitian yang membutuhkan sentuhan olahan data menggunakan algoritma, diantaranya pada bidang kesehatan (diagnosa penyakit), bidang pendidikan (prediksi kelulusan), bidang perbankan (penentuan kredit untuk nasabah), dan sebagainya. Pohon keputusan yang dihasilkan oleh *decision tree* sangat fleksibel karena saran atau prediksinya divisualisasikan dalam bentuk pohon sehingga mudah untuk diamati. Selain itu teknik data mining jenis ini memiliki tingkat akurasi yang termasuk ke dalam kategori tinggi pada banyak kasus permasalahan. Algoritma yang termasuk ke dalam klasifikasi *decision tree* diantaranya yaitu ID3 dan C4.5. Algoritma C4.5 merupakan algoritma perbaikan dari algoritma sebelumnya yaitu ID3 dimana algoritma C4.5 lebih baik karena dapat mengolah data bertipe kategorikal dan numerik.

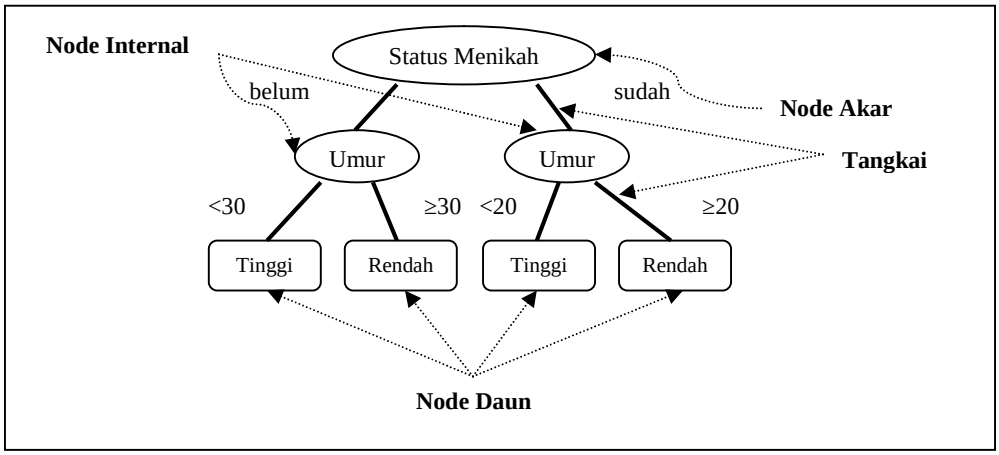
   Pada Tabel 2.2 diperlihatkan contoh kasus klasifikasi data mining tentang seberapa tinggi kemungkinan resiko yang muncul terhadap pengembalian uang pinjaman bank oleh nasabah. Resiko dengan nilai tinggi artinya nasabah susah untuk membayar angsuran pinjaman, sebaliknya bila resiko rendah artinya nasabah diprediksi dapat membayar angsuran pinjaman [6].

Tabel 2.2 Data Catatan Resiko Kredit

Umur	Status	Risiko
15	Sudah	Tinggi
17	Sudah	Tinggi
18	Belum	Tinggi
24	Sudah	Rendah
25	Belum	Tinggi
28	Sudah	Rendah
29	Belum	Tinggi
40	Belum	Rendah
45	Sudah	Rendah
48	Belum	Rendah

Ciri khas dari kasifikasi *data mining* adalah dibentuknya sejumlah node yang menyusun pohon keputusan, berikut adalah jenis-jenis node klasifikasi:

1) Node akar, pangkal induk dari pohon dan memiliki nol atau lebih tangkai keluaran.
2) Node internal, merupakan node yang bukan nonterminal (tidak memiliki tangkai keluaran). Memiliki minimal satu tangkai keluaran.
3) Tangkai, penghubung antara node akar dengan node internal, dan node internal dengan node daun.
4) Node daun, node yang memilki 1 tangkai masukan dan tidak memiliki satu pun tangkai keluaran. Merepresentasikan label kelas atau keputusan.



Gambar 2.3 Contoh Node Decision Tree

b. Klasifikasi Berbasis *Artificial Neural Network*

Artificial Neural Network (ANN) merupakan teknik yang merekayasa knowledge berbasis sistem saraf manusia yang pemrosesan utamanya terletak pada otak.

c. Klasifikasi Berbasis *Support Vector Machine*

Metode *Support Vector Machine* berakar dari teori pembelajaran statistik. Metode ini hanya menggunakan sejumlah data terpilih saja yang akan digunakan untuk membangun model.

d. Klasifikasi Berbasis *Nearest Neighbor*

Metode ini mengklasifikasikan kemiripan diantara data-data dan algoritmanya hanya akan berjalan jika terdapat data uji yang ingin dicari label kelasnya.

4. Pengklusteran

Pengklusteran yaitu mengelompokkan record, mengamati, serta membangun kelas dari objek yang mempunyai kesamaan. Kluster adalah sekumpulan *record* yang saling mempunyai kesamaan serta mempunyai ketidaksamaan terhadap kluster yang berbeda. Kelompok kluster tidak sama seperti klasifikasi yakni tidak terdapat variable target pada pengklusteran.

5. Asosiasi

Pada data mining, asosiasi memiliki tugas mencari atribut yang ada pada suatu waktu. Algoritma asosiasi umumnya dinamakan dengan analisis keranjang belanja.

2.4.2 Proses *Data Mining*

Secara umum, terdapat tiga proses inti dalam data mining:

1. Eksplorasi atau pemrosesan awal data

   Proses awal data mining terdiri dari berbagai aktivitas yang menyangkut tentang data diataranya adalah pembersihan data, mentransformasikan data, normalisasi data, dan sebagainya.
2. Pembangunan model dan validasi

   Proses ini menganalisis dan memilih model-model yang memiliki tingkat prediksi yang paling baik.
3. Penerapan

   Proses terakhir adalah memasukkan data kedalam model yang dipilih untuk mendapatkan hasil penyelesaian masalah melalui prediksi.

2.5 Algoritma C4.5

Algoritma C4.5 merupakan versi perbaikan dari algoritma ID3, C4.5 mampu memproses fitur bertipe kategorikal dan numerik sedangkan ID3 hanya mampu mengolah fitur bertipe kategorikal saja.

Dalam membuat pohon keputusan menggunakan algoritma C4.5 langkah-langkah yang harus dilakukan adalah sebagai berikut [7]:

1. Menyiapkan data training kemudian membuatnya ke dalam kelas-kelas.
2. Mencari nilai *entropy* menggunakan rumus *entropy*. Setelah *entropy* didapat kemudian mencari nilai *gain* dari setiap atribut yang ada. Atribut yang memiliki nilai *gain* terbesar akan digunakan sebagai akar pohon. Berikut adalah rumus untuk mencari nilai *entropy* dan gain:

$$Entropy(S) = \sum_{i=1}^{n} - pi * log_2 pi \quad (2.1)$$

Keterangan:

S : Himpunan kasus

n : Jumlah partisi S

Pi : Proporsi Si kepada S

$$Gain(S,A) = Entropy(S) - \sum_{i=1}^{n} \frac{|Si|}{|S|} * Entropy(Si) \quad (2.2)$$

Keterangan:

S : Himpunan kasus

A : fitur

n : Jumlah partisi atribut A

|Si| : Proporsi Si terhadap S

|S| : Jumlah kasus dalam S

3. Ulang langkah ke-2 sampai seluruh *record* terpartisi.
4. Langkah-langkah ini akan berakhir apabila seluruh record pada simpul N memilki kelas sama, tidak ada atribut *record* yang terpartisi lagi serta tidak ada atribut cabang yang kosong.

Nilai keakuratan proses klasifikasi diperoleh melalui rumus akurasi dalam persamaan [8] :

$$Akurasi = \frac{\text{Jumlah Pengujian Benar}}{\text{Jumlah Pengujian}} \text{x100\%} \quad (2.3)$$

2.6 Sistem Pakar

Sistem pakar adalah suatu sistem yang didalamnya digunakan pengetahuan manusia yang mana pengetahuan tersebut di inputkan ke dalam komputer untuk dipergunakan

sebegai penyelesaian permasalahan-permasalahan yang umumnya memerlukan kepakaran atau kehebatan manusia [9]. Didalam sistem pakar, pengetahuan para pakar di masukkan ke dalam komputer untuk menyelesaikan berbagai masalah sehingga sistem ini dikenal juga dengan istilah *knowledge-based-expert system*.

2.6.1 Konsep Dasar Sistem Pakar

Terdapat enam konsep di dalam sistem pakar, diantaranya sebagai berikut:

1. Kepakaran

   Kepakaran merupakan pengetahuan yang didapatkan dari hasil berlatih, membaca, serta pengalaman dari mempelajari sesuatu hal sehingga memudahkan para ahli dalam melakukan pengambilan keputusan yang lebih cepat dan baik. Kepakaran mencakup berbagai pengetahuan mengenai:

   a. Fakta atau kebenaran-kebenaran dalam bidang kajian tertentu.
   b. Teori-teori mengenai bidang kajian tertentu.
   c. *Rules* atau prosedur-prosedur berdasarkan bidang kajian umumnya.
   d. Aturan heuristik yang perlu dilakukan terhadap sebuah instansi tertentu.
   e. Strategi global guna menyelesaikan suatu permasalahan.
   f. Pengetahuan mengenai pengetahuan.

2. Pakar

   Pakar adalah orang dengan pengetahuan, pengalaman, juga metode khusus, dan dapat menggunakannya guna menyelesaikan permasalahan ataupun memberikan nasehat [9]. Pakar diharuskan bisa memberi penjelasan serta mempelajari sesuatu yang baru yang berhubungan dengan bahasan masalah. Bila perlu pakar diharuskan dapat menata kembali seluruh pengetahuan yang diperoleh dan mampu menyelesaikan aturan-aturan serta menetapkan relevansi kepakaranya. Oleh sebab itu, pakar diharuskan dapar mengerjakan kegiatan-kegiatan sebagai berikut:

   a. Mengetahui dan memformulasikan masalah.

b. Menyelesaikan masalah dengan cepat dan tepat.
c. Menjelaskan solusi pemecahan masalah.
d. Mengambil pelajaran dari pengalaman.
e. Restrukturisasi pngetahuan.
f. Pemecahan aturan-aturan.
g. Penentuan relevansi.

3. Pemindahan Kepakaran

   Tujuan Sistem Pakar yaitu memindah kepakaran seorang pakar ke dalam komputer, lalu mentransferkanya ke seseorang yang bukan pakar. Pemindahan ini meliputi empat aktivitas sebagai berikut:

   a. Mengakuisisi pengetahuan (dari pakar maupun sumber lainya).
   b. Merepresentasikan pengetahuan (pada komputer).
   c. Menginferensi pengetahuan.
   d. Memindah pengetahuan ke user.

4. Inferensi (*Inferencing*)

   Inferensi adalah suatu prosedur (program) yang memiliki keahlian dalam mengerjakan suatu penalaran [9]. Inferensi dimunculkan pada sebuah komponen yang bernama mesin inferensi yang meliputi prosedur-prosedur tentang penyelesaian permasalahan. Segala pengetahuan yang diperoleh oleh pakar diletakkan pada basis pengetahuan oleh sistem pakar. Mesin inferensi bertugas mengambil keputusan atas dasar basis pengetahuan yang diperolehnya.

5. Aturan-aturan (*Rules*)

   Umumnya *software* sistem pakar merupakan sistem yang menggunakan basis *rule* (*rule-based-system*), yakni menyimpan pengetahuan dalam suatu bentuk berupa *rule* yang memiliki peran sebagai pemecah permasalahan.

6. Kemampuan Memberi Penjelasan

   Sistem Pakar memiliki kemampuan memberikan penjelasan mengenai usulan maupun rekomendasi yang diberikan yang mana hal tersebut dilakukan pada

subsistem bernama subsistem penjelasan. Sistem ini mempunyai bangian dimana bagian tersebut memberikan kemungkinan kepada sistem guna melakukan pemeriksaan penalaran yang dibangunya sendiri serta membri penjelasan tentang berbagai operasinya.

2.6.2 Struktur Sistem Pakar

Terdapat komponen-komponen utama didalam sistem pakar, yakni antarmuka pengguna (*user interface*), basis data sistem pakar (*expert system database*), fasilitas akuisisi pengetahuan (*knowledge acquisition facility*), dan mekanisme inferensi (*inference mechanism*). Terdapat satu komponen yang hanya ada di beberapa sistem pakar, yakni fasilitas penjesalan (*explanation facility*) (Martin dan Oxman, 1988) [10].

1. Antarmuka Pengguna (User Interface)

   Komponen ini merupakan perangkat lunak menjadi media komunikasi antara pengguna dengan sistem.
2. Basis Data Sistem Pakar (Expert System Database)

   Basis data ini berisi pengetahuan-pengetahuan dari para pakar terhadap subyek tertentu yang diperlukan untuk memahami, merumuskan, serta menyelesaikan masalah dimana pengetauan tersebut diperoleh dari pakar, jurnal, majalah, maupun sumber pengetahuan yang lain. Basis data sistem pakar terdiri dari 2 elemen dasar :

   a. Fakta, merupakan situasi masalah dan teori yang terkait.
   b. Heuristik khusus atau rules, yang langsung menggunakan pengetahuan untuk menyelesaikan masalah khusus.
3. Fasilitas Akuisisi Pengetahuan (Knowledge Acquisition Facility)

   Fasilitas ini merupakan perangkat lunak yang memberikan fasilitas dialog antara pakar dengan sistem yang gunanya adalah untuk memasukkan fakta-fakta dan kaidah-kaidah sesuai dengan perkembangan ilmu. Fasilitas akuisisi pengetahuan

meliputi proses pengumpulan, pemindahan, dan perubahan dari kemampuan pemecahan masalah seorang pakar maupun sumber pengetahuan terdokumentasi ke dalam komputer yang bertujuan untuk memperbaiki dan atau mengembangkan basis pengetahuan (knowledge-base).

4. Mekanisme Inferensi (Inference Mechanism)

   Mekanisme inferensi adalah perangkat lunak yang melakukan penalaran dengan menggunakan pengetahuan yang ada untuk menghasilkan suatu kesimpulan atau hasil akhir. Terdapat pemodelan proses berpikir manusia dalam komponen ini. Terdapat dua cara untuk mengerjakan inferensi, yakni [9]:

   a. Forward chaining

      Cara ini merupakan kelompok multiple inferensi yang melakukan pencarian dari suatu masalah kepada solusinya. Inferensi dimulai dengan informasi yang tersedia dan barulah memperoleh konklusi.

   b. Backward chaining

      Cara ini dimulai dari ekspektasi apa yang diinginkan terjadi (hipotesis), kemudian mencari bukti yang mendukung (atau kontradiktif) dari ekspektasi tersebut.

5. Fasilitas Penjelasan (Explanation Facility)

   Fasilitas ini memberi penjelasan kepada pengguna mengapa komputer meminta suatu informasi dari pengguna serta dasar apa yang digunakan komputer sehingga mampu menyimpulkan suatu kondisi.

2.6.3 Case Based Reasoning

Cased Based Reasoning (CBR) adalah sebuah teknik dalam penyelesaian permasalahan berdasarkan pengalaman sebelumnya, guna mencari kemiripan kasus-kasus melalui pengadaptasian masalah yang lalu-lalu dengan masalah yang baru agar menghasilkan sebuah kesimpulan (Armengol, 2001) [11].

Terdapat empat tipe CBR yang digambarkan oleh Aamodt dan Plaza (1994) sebagai sebuah proses melingkar yakni :

Gambar 2.4 Skema Proses CBR

1. *Retrieve* : merupakan proses perolehan berbagai kasus yang memilki kesamaan dan selajutnya dibandingkan dengan berbagai kasus dimasa lalu.
   Terdapat proses-proses didalam *retrieve*, yakni :
   a. Indentifikasi masalah
   b. Memulai pencocokan
   c. Penyeleksian
2. *Reuse* : yakni menggunakan ulang masalah-masalah yang pernah ada di masa lalu guna mendapatkan solusi untuk menyelesaikan masalah yang terjadi sekarang atau masalah baru.
3. *Revise* : merupakan proses mengubah dan mengadopsi solusi yang ditawarkan jika dibutuhkan. Terdapat 2 tugas utama pada proses *revise* yakni :
   a. Evaluasi solusi, yakni melihat hasil dari membandingkan solusi dengan keadaan yang sebenarnya.
   b. Memperbaiki kesalahan, meliputi pengenalan kesalahan dari solusi yang dibuat dan membuat penjelasan tentang kesalahan tersebut.
4. *Retain* : menggabungkan solusi kasus baru ke *knowledge* yang telah ada.

Pada saat terjadi permasalahan baru, pertama-tama sistem akan melakukan. proses *retrieve*. Berikutnya, sistem melakukan proses *reuse*, menggunakan informasi permasalahan sebelumnya yang memiliki kemiripan untuk menyelesaikan masalah yang baru. Kemudian proses *revise*, mengevaluasi serta memperbaiki kembali informasi untuk mengatasi berbagai kesalahan yang terdapat pada masalah yang baru. Proses terakhir yaitu retain, mengindeks dan mengintegrasi, sekaligus mengekstrak solusi yang baru yang nantinya akan disimpan pada *knowledge-base* untuk menyelesaikan masalah yang akan datang.

2.7 RapidMiner

RapidMiner berkembang sejak tahun 2001, sebelumnya disebut dengan nama YALE (*Yet Another Learning Environment*). *Software* ini dikembangkan oleh Ralf Klinkenberg, Ingo Mierswa, serta Simon Fischer pada *Unit Artificial Intelligence* dari *Technical University of Dortmund*.

RapidMiner adalah *platform* analisis modern yang meliputi *data mining*, *mechine learning*, analisis prediktif, *text mining* dan analisis bisnis [12]. *Software* ini digunakan untuk mengukur kinerja algoritma dan untuk menemukan algoritma terbaik yang akan berguna untuk klasifikasi, prediksi dan teknik lainya di *data mining*. RapidMiner merupakan *software* yang *user friendly* dan memiliki GUI (*Graphic User Interface*) yang efektif yang digunakan untuk bekerja dengan mudah. RapidMiner memberikan *mechine learning* dan data prosedur termasuk *loading data* dan transformasi (*Extract, Transform, Load* (ELT)), data *preprocessing* dan pemodelan statistik, visualisasi dan analisis prediktif, penyebaran dan evaluasi.

Terdapat sifat-sifat yang dimiliki oleh RapidMiner, yakni :

1. Penulisan menggunakan bahasa Java. Hal ini memungkinkan RapidMiner bisa berjalan pada sistem operasi yang berbeda-beda.
2. Proses menemukan pengetahuan dituangkan dalam model operator *trees*.

3. Merepresentasikan XML internal guna memungkinkan format standar pertukaran data.
4. Penggunaan bahasa *scripting* yang memungkinkan untuk eksperimen dalam skala besar dan pengotomatisasian eksperimen.
5. Konsep multi-layer yang menjadikan tampilan data menjadi efisien serta memastikan penanganan data.
6. Mempunyai GUI (*Graphic User Interface*), *command line mode*, serta Java API yang bisa dipanggil melalui program lain.

2.7.1 Fitur RapidMiner

Adapun fitur-fitur yang terdapat pada RapidMiner adalah sebagai berikut :

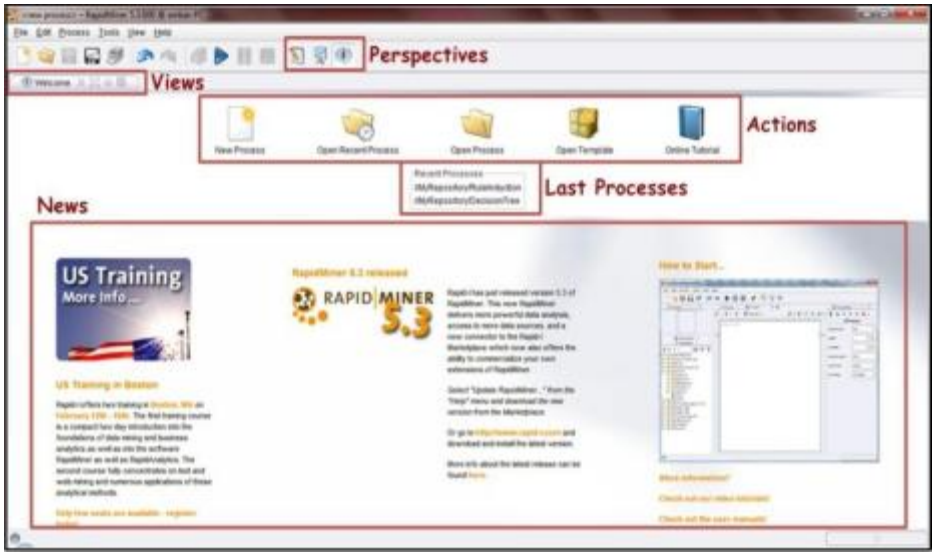
1. Terdapat banyak algoritma data mining seperti *decision tree* dan *self-organization map*.
2. Bentuk grafis yang handal seperti tumpang tindih diagram historgram, *tree chart* dan 3d *Scatter plots*.
3. Memiliki banyak variasi *plugin*, seperti *text plugin* yang dapat digunakan untuk melakukan analisis teks.
4. Tersedia prosedur data mining dan *machine learning* termasuk ELT, data *preprocessing*, visualisasi, *modelling* dan evaluasi.
5. Proses data mining disusun berdasarkan operator-operator yang *nestable*, dideskripsikan dengan XML, dan dibangun dengan GUI.
6. Mengintegrasikan proyek *data mining* Weka dan statistik R.

2.7.2 *Interface* RapidMiner

RapidMiner memiliki tampilan antarmuka yang *user friendly* yang memberikan kemudahan kepada pengguna dalam memakainya dan dikenal dengan sebutan *Perspective*. Ada tiga *perspective* yang yang terdapat pada RapidMiner yakni:

1. Welcome Perspective

Welcome perspective merupakan tampilan antarmuka *perspective* yang pertama yang terdapat pada *interface*.



Gambar 2.5 Tampilan *Welcome Perspective*

Berikut adalah penjelasan tentang bagian-bagian pada *Welcome Perspective*:

a. ***Perspective*** : bagian ini terdiri dari ikon-ikon yang berguna untuk menampilkan *perspective* dari RapidMiner dan pengguna bisa mengkonfigurasi *toolbar* ini sesuai dengan keinginannya.

b. ***Views*** : bagian ini merujuk kepada pandangan (*view*) yang sedang ditampilkan.

c. ***Actions*** : merujuk kepada daftar aksi yang bisa digunakan setelah membuka RapidMiner, aksi-aksi tersebut diantaranya :

   1) ***New*** : digunakan untuk membuat proses analisis baru. Yang harus dilakukan dalam membuat proses analisis baru yaitu menentukan nama serta lokasi proses, dan *Data Repository*.

   2) ***Open Recent Process*** : aksi ini digunakan untuk membuka proses yang baru saja ditutup. Cara lain untuk melakukan aksi ini yaitu dengan *double* klik pada salah satu daftar yang terdapat di *Recent Process* lalu secara

otomatis tampilan *Welcome Perspective* beralih menjadi tampilan *Design Perspective*.

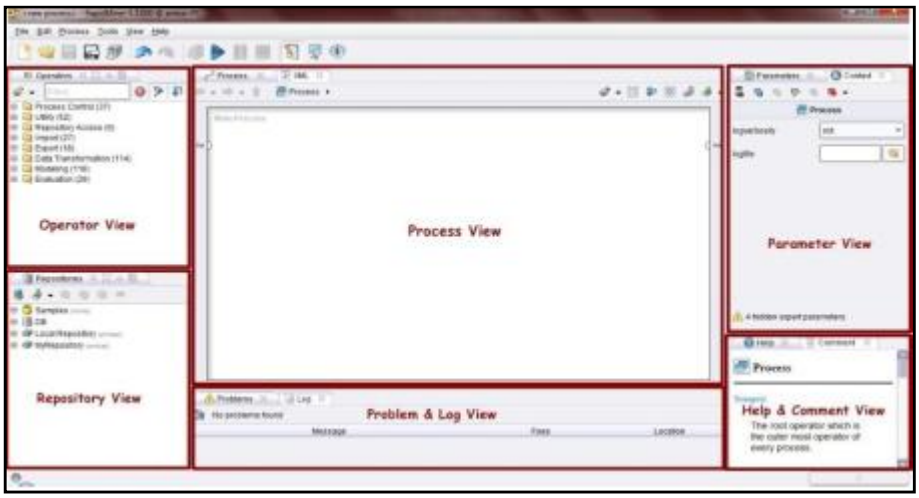
3) ***Open Process*** : berguna untuk membuka *Repository Browses* yang berisikan daftar proses.
4) ***Open Template*** : aksi ini menunjukkan pilihan lain yang sudah ditentukan oleh proses analisis.
5) ***Online Tutorial*** : berguna untuk memulai tutorial secara *online*.

d. ***Last Processes*** : bagian ini menunjukkan daftar proses yang baru saja dilakukan.

e. ***News*** : bagian ini akan menampilkan berita-berita terbaru yang berkaitan dengan RapidMiner. Berita tersebut hanya akan muncul jika komputer terhubung dengan internet.

2. *Design Perspective*

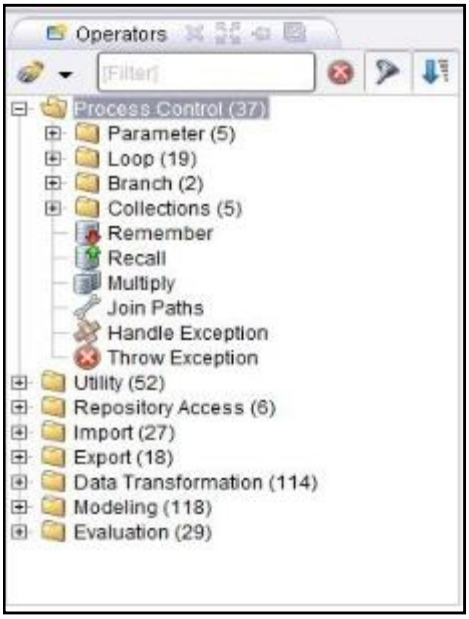
Design perspective merupakan area kerja pada RapidMiner yang berguna untuk membuat serta melakukan pengolahan terhadap proses analisis.



Gambar 2.6 Tampilan *Design Perspective*

a. ***Operator View*** : menyajikan seluruh operator atau langkah kerja dalam bentuk hierarki agar bisa bekerja pada proses analisis. Operator ini memiliki beberapa kelompok operator, yaitu :

1) ***Process Control*** : berguna untuk mengatur aliran proses melalui operator perulangan dan percabangan.
2) ***Utility*** : Operator Bantuan seperti operator *macros*, *loggin*, subproses, dan sebagainya.
3) ***Repository Access*** : berguna untuk membaca atau menulis akses pada repository melalui operator-operator yang terdapat didalamnya.
4) ***Import*** : terdiri dari berbagai operator yang berguna untuk membaca data dan objek dari format tertentu seperti file, *database*, dan lain-lain.
5) ***Export*** : berisi operator yang berguna untuk menulis data dan objek menjadi format tertentu.
6) ***Data Transformation*** : kelompok ini berisi seluruh operator yang dapat digunakan dalam transformasi data dan meta data.
7) ***Modelling*** : terdiri dari proses data mining guna menerapkan model yang dihasilkan menjadi set data yang baru.
8) ***Evaluation*** : digunakan untuk menghitung kualitas pemodelan dan untuk data baru.

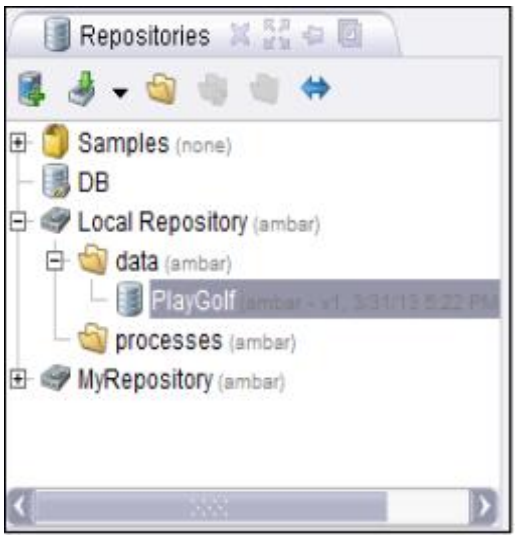


Gambar 2.7 Kelompok Operator dalam Bentuk Hierarki

b. ***Repository View*** : merupakan komponen utama pada *Design Perspective* selain Operator *View*. *Repository view* bisa digunakan untuk mengelola sekaligus menata proses analisis menjadi proyek dan sebagai sumber data maupun meta data.
c. ***Process View*** : menunjukkan langkah-langkah tertentu pada proses analisis serta sebagai penghubung langkah-langkah tersebut yang mana penghubung tersebut dapat dilepas kembali.
d. ***Parameter View*** : beberapa operator pada RapidMiner membutuhkan satu atau lebih parameter agar dapat diindikasikan sebagai fungsionalitas yang benar.
e. ***Help & Comment View*** : *Help view* menunjukkan penjelasan mengenai operator yang dipilih melalui Operator *View* maupun *Process View*, sedangkan *Comment View* digunakan untuk menuliskan komentar pada langkah-langkah proses tertentu.
f. ***Problem & Log View*** : setiap peringatan dan pesan kesalahan akan ditampilkan pada *Problem View*.

2.7.3 *Decesion Tree* pada RapidMiner

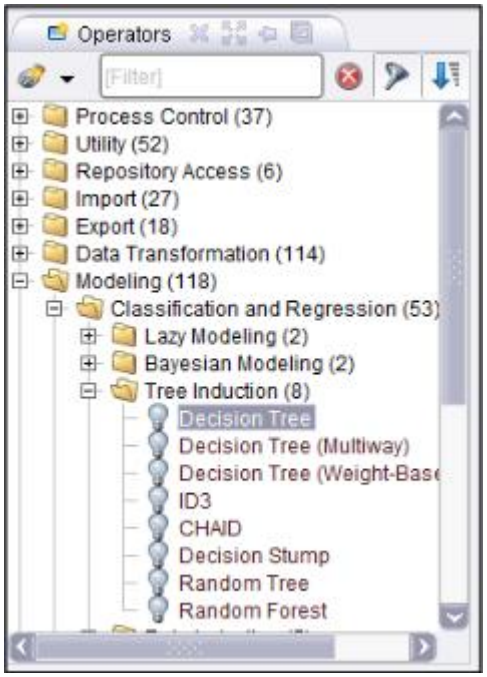
RapidMiner menyediakan *tools* yang dapat digunakan untuk membuat *decision tree*. Langkah-langkah dalam membuat *decition tree* pada RapidMiner yaitu pertama-tama import data ke dalam *repository* RapidMiner kemudian lakukan *drag* dan *drop* data tersebut pada *view process* untuk mengubah data yang berisi atribut pohon keputusan menjadi opreator *retrieve*.



Gambar 2.8 Lokasi Tabel Pada *Repository*

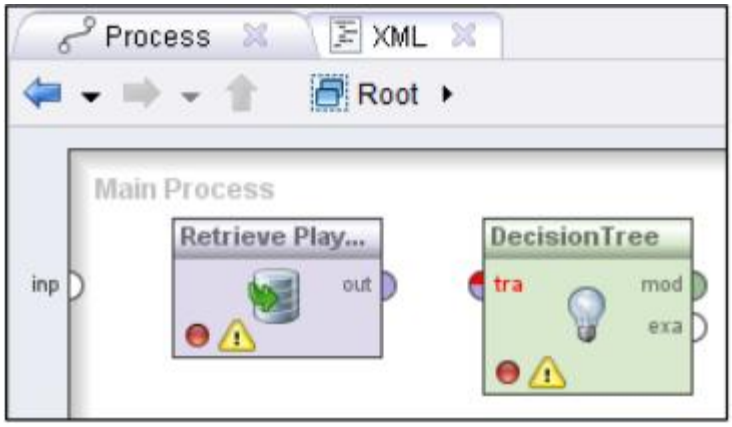
Gambar 2.9 *Repository* PlayGolf pada *Main Process*

Selanjutnya pilih operator *Decision Tree* pada *View Operators* dengan cara pilih *Modelling* lalu pilih *Classification and Regression*, lalu pilih *Tree Induction* dan pilih *Decision Tree*.



Gambar 2.10 Daftar Operator pada *View Operators*

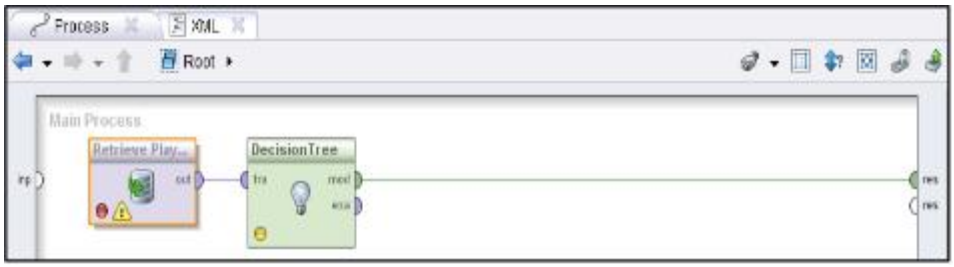
Langkah berikutnya, *drag* dan *drop* operator *Decision Tree* ke dalam *View Process* dan posisikan disamping operator *Retrieve*.



Gambar 2.11 Posisi Operator *Decision Tree*

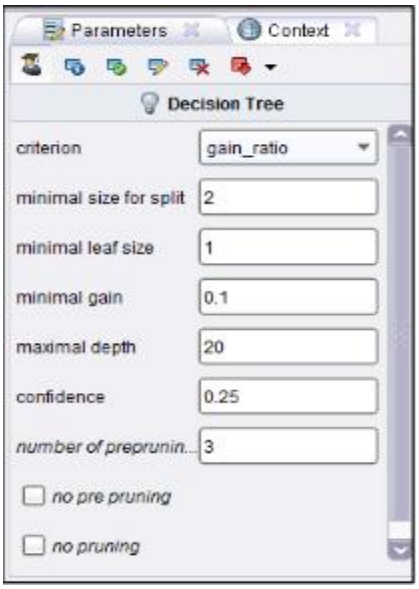
Langkah berikutnya, hubungkan operator *Retrieve* dengan operator *Decision Tree*, caranya tarik garis dari tabel *PlayGolf* ke operator *Decision Tree* kemudian tarik garis lagi dari operator *Decision Tree* ke result disisi kanan. Operator *Decision Tree*

berfungsi untuk melakukan prediksi terhadap atribut-atribut yang dimasukkan ke dalam operator *Retrieve*.

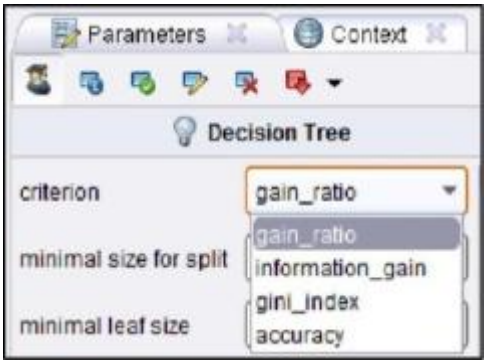


Gambar 2.12 Menghubungakan Tabel *PlayGolf* dengan Operator *Decision Tree*

Operator *Decision Tree* memilki input *training* set (tra) yang mana *port* ini adalah *output* dari operator *retrieve* dan menghasilkan *ExampleSet* yang bisa diproses menjadi *decision tree*. Terdapat pula *output* model (mod) yang akan mengkonversi atribut menjadi model keputusan dalam bentuk *decision tree*, dan *output* exa dimana hasil *output* yang dihasilkan tidak merubah inputan yang masuk dari *port* ini. Berikutnya yaitu mengatur parameter sesuai dengan kebutuhan.



Gambar 2.13 Parameter *Decision Tree*



Gambar 2.14 Tipe *Criterion*

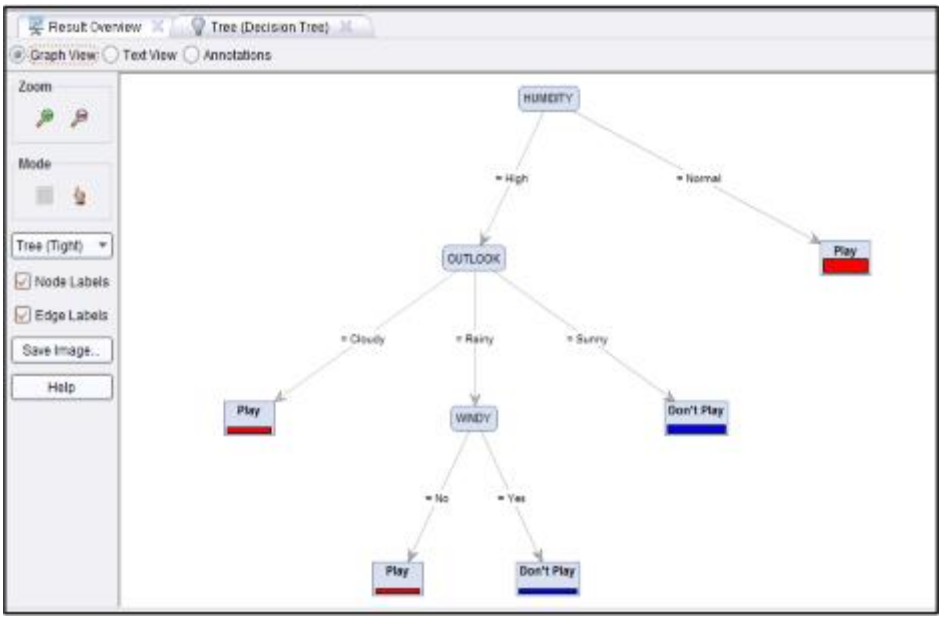
1. ***Criterion*** : digunakan untuk memilih kriteria untuk menentukan atribut yang akan dijadikan sebagai akar *decision tree*. Kriteria-kriteria pada criterion yaitu :
   a. ***Gain_ratio*** : menghasilkan *information gain* untuk setiap atribut yang memberikan nilai atribut yang seragam.
   b. ***Information_gain*** : dengan metode ini semua semua *entropy* akan dihitung.
   c. ***Gini_index*** : memisahkan atribut yang dipilih memberikan penurunan indeks gini rata-rata yang dihasilkan *subset*.
   d. ***Accuracy*** : memilih beberapa atribut untuk memecah pohon yang memaksimalkan akurasi dari keseluruhan pohon.
2. ***Minimal size of split*** : digunakan untuk menentukan ukuran dari simpul-simpul pada *decision tree*.
3. ***Minimal leaf size*** : pohon yang dihasilkan sedemikian rupa memiliki himpunan bagian simpul daun setidaknya sebanyak jumlah minimal *leaf size*.
4. ***Minimal gain*** : merupakan nilai *gain* minimal yang ditentukan untuk menghasilkan simpul pohon keputusan.
5. ***Maximal depth*** : digunakan untuk membatasi ukuran putusan pohon.
6. ***Confidence*** : berguna untuk menentukan tingkat kepercayaan yang digunakan untuk pesimis kesalahan perhitungan pemangkasan.

7. ***Number of prepurning alternative*** : parameter ini menyesuaikan jumlah node alternatif mencoba untuk membelah ketika split dicegah dengan prepurning pada simpul tertentu.

Setelah parameter selesai diatur, kemudian klik ikon *Run* pada *toolbar* untuk menampilkan hasil.

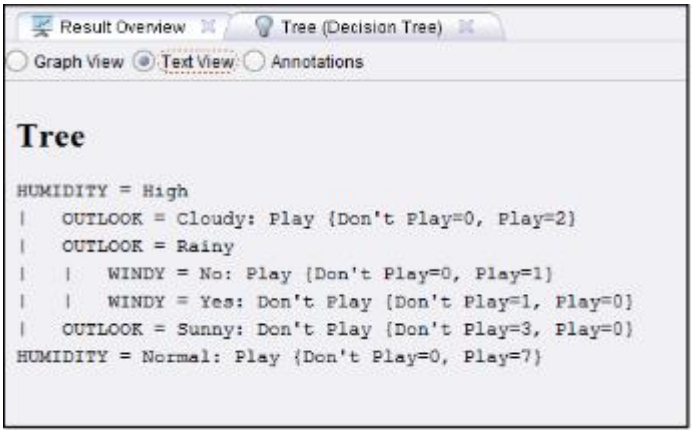
Gambar 2.15 Ikon Run

Untuk menampilkan hasil berupa pohon keputusan pilih *Graph View*.



Gambar 2.16 Hasil Berupa *Graph* Pohon Keputusan

Sedangkan untuk menampilkan hasil berupa teks maka pilih *Text View*.

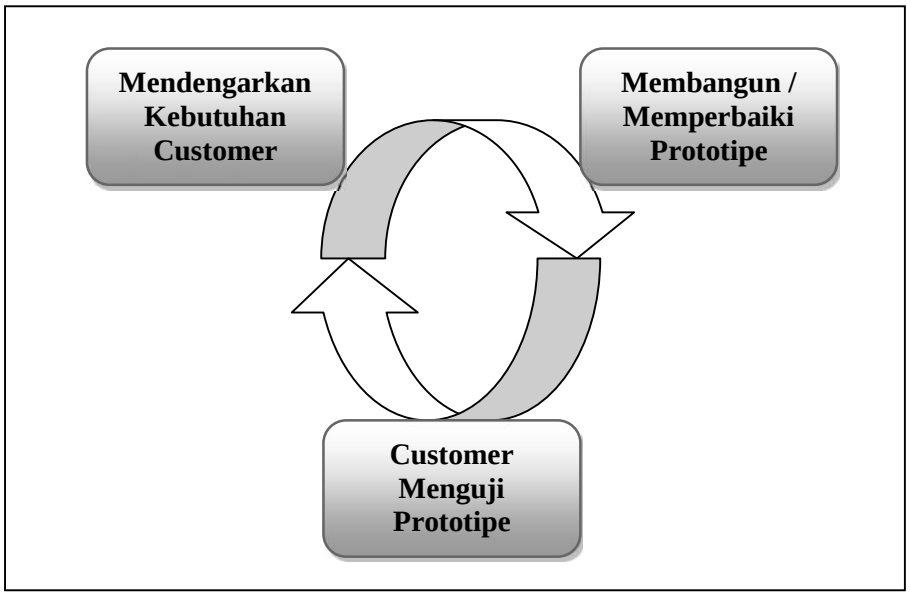


Gambar 2.17 Hasil Berupa Penjelasan Teks

2.8 Pengembangan Sistem

2.8.1 Model *Prototipe*

Prototipe adalah sebuah metode yang rapih serta runtut guna membangun sistem. Model *prototipe* dapat menjadi sarana penghubung dalam mempertemukan kesepakatan antara pengembang dan *customer* tentang hal teknis serta spesifikasi yang dibutuhkan yang berkaitan dengan sistem. Model ini diawali dengan pengumpulan kebutuhan sistem dari *customer*. Kemudian mulai membuat *prototipe* sistem sehingga *customer* mendapatkan deskripsi lebih jelas tentang apa yang dibutuhkan sebenarnya. *Prototipe* tersebut dievaluasi oleh customer secara terus menerus hingga sesuai dan cocok dengan pemikiran serta keinginan customer.

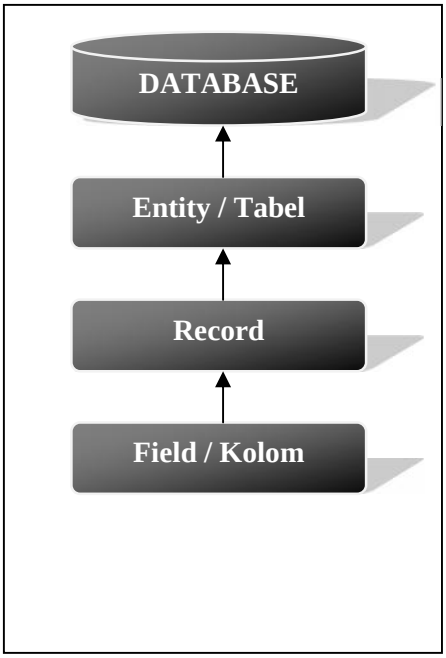


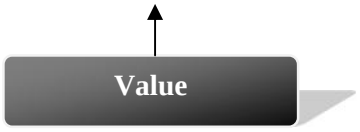
Gambar 2.18 Ilustrasi Model *Prototipe*

Model *prototipe* sesuai untuk mendeskripsikan kebutuhan *user* dengan detil sebab *customer* mendapatkan gambaran secara langsung dan nyata tentang sistem yang akan dibangun. Dengan demikian, proses yang terjadi didalam model ini mampu meminimalisir ketidaksinkronan pemikiran antara customer dan pengembang, dan nantinya hasil akhirnya pun akan lebih tepat dengan kemauan *customer*.

2.8.2 Database

Sistem *database* merupakan sistem yang terkomputerisasi yang bertujuan untuk memelihara data atau informasi serta menyiapkanya kapanpun informasi itu diperlukan, terutama bila diperlukan dalam waktu yang cepat. Dengan sistem *database* pengaksesan sebuah data menjadi lebih cepat dan praktis, selain itu penggunaanya juga dapat menghemat ruang penyimpanan karena *database* mampu menampung data dalam skala yang besar. *Database* memungkinan penggunaan *multiple user* atau lebih dari satu orang yang dapat mengaksesnya. Namun, sistem ini juga menyediakan fitur yang dapat digunakan untuk mengatur hak akses *user* terhadap suatu data sehingga tidak semua data bisa diakses oleh sembarang *user*. Terdapat beberapa tingkatan data yang tersedia di dalam sistem *database*, yaitu [13] :





Gambar 2.19 Tingkatan Data Pada *Database*

1. *Value* / nilai

   Value merupakan data dengan tingkat terendah pada *database* dan disimpan pada sebuah kolom atau elemen.

2. *Field* / Kolom / Atribut

   Tingkat kedua pada *database* yaitu *filed* atau kolom. Tingkatan ini berperan menyusun *record-record*.

3. *Record* / Baris

   Baris atau *record* merupakan sekumpulan *field* yang berkaitan satu sama lain sehingga terbentuk sebuah tabel dan pada setiap *record* tersimpan sebuah informasi mengenai objek yang disimpan.

4. *Entity* / Tabel

   Entity atau tabel merupakan sekumpulan *record* yang mendeskripsikan mengenai subjek data.

5. *Database*

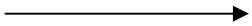
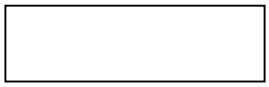
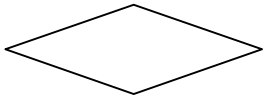
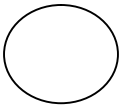
   Database merupakan sekumpulan tabel yang mendeskripsikan sebuah subjek data.

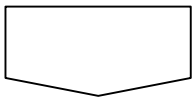
2.8.3 Flowchart

Flowchart merupakan alat bantu yang dapat digunakan untuk mengetahui dan memahami alur kerja dari sebuah sistem yang dituangkan dalam bentuk diagram. Diagram tersebut berisi urutan-urutan tentang bagaimana suatu proses ataupun sistem berjalan. Dengan demikian alur kerja sistem dapat dipahami melalui gambaran logika yang direpresentasikan menggunakan simbol-simbol.

Berikut adalah simbol-simbol yang dapat digunakan di dalam membuat sebuah *flowchart* :

Tabel 2.3 Simbol-simbol *Flowchart*

Simbol	Nama	Fungsi
	Terminator	Permulaan / akhir program
	Garis Alir (*Flow Line*)	Arah aliran program
	*Preparation*	Proses inisialisasi / pemberian nilai awal
Simbol	**Nama**	**Fungsi**
	Proses	Proses perhitungan / pengolahan data
	*Input* / *Output* Data	Proses *input* / *output* data, parameter, dan informasi
	*Decision*	Perbandingan pernyataan dan penyeleksian data yang memberikan pilihan untuk langkah selanjutnya
	*On Page Connector*	Penghubung bagian-bagian *flowchart* yang berada pada satu halaman

	*Off Page Connector*	Penghubung bagian-bagian *flowchart* yang berada pada halaman yang berbeda

2.8.4 Java

Java merupakan bahasa pemrograman yang tidak bergantung kepada mesin. Artinya, java merupakan pemrograman lintas *platform* dan didukung dengan konsep *object oriented programming* [14]. Java memiliki beberapa karakteristik, diantaranya :

1. Sederhana

   Penggunaan sintaks pada Java hampir sama dengan C++, akan tetapi sudah mengalami perbaikan khususnya menghapus pemakaian *pointer*.
2. Berbasis objek

   Java merupakan bahasa yang berorientasi objek sehingga program bisa dibangun dengan modular serta bisa digunakan ulang. Dunia nyata dimodelkan menjadi objek dan berinteraksi dengan sesama objek tersebut.
3. Mudah untuk didistribusikan

   Java dibangun guna memudahkan dalam pendistribusian aplikasi.
4. *Interpreter*

   Program java berjalan dengan penggunan *interpreter* yakni java *virtual machine*. Dengan demikian *source code* yang sudah terkompilasi bisa berjalan di *platform* yang lain.
5. *Robust*

   Robust memiliki arti bisa diandalkan. Java memiliki kereabilitasan yang tinggi. *Compiler* yang terdapat di dalam java memilki keahlian yang teliti dalam pendeteksian *error*.
6. Aman

Sebagai bahasa program yang digunakan untuk pembuatan aplikasi internet sekaligus dapat didistribusikan, java tidak memberikan akses secara bebas terhadap sistem sehingga sistem aman.

7. *Architecture neutral*

   Java menghasilkan program yang bisa berjalan diberbagai *platform*, ini disebabkan java bekerja dengan java *virtual mechine*.

8. *Portabel*

   Program java bisa berjalan pada platform lain dan tidak perlu dilakukan kompilasi ulang.

9. *Performance*

   Performa java seringkali disebut lamban, namun hal ini bisa diatasi dengan semakin majunya teknologi di jaman sekarang ini yang menyediakan berbagai prosesor dengan tingkat pemrosesan yang cepat.

10. *Multithread & Dinamis*

    Java menghasilkan program yang bisa mengerjakan banyak tugas sekaligus serta lebih dinamis karena sengaja didesain untuk lingkungan pengembangan.

2.8.5 Struktur Program Java

1. *Package*

   Package merupakan sebuah cara untuk mengelompokkan dan mengorganisasikan kelas-kelas ke dalam sebuah *library*. *Package* berjalan cara pembuatan direktori berserta folder baru sesuai dengan nama *package*. Setelah itu simpan file *class* di dalam folder tersebut.

2. *Import*

   Import berguna untuk memberitahu program untuk merujuk ke kelas-kelas yang ada di dalam *package*.

3. Kelas

Kelas adalah bagian yang terpenting pada Java. Kelas merupakan sebuah *body* yang didalamnya terdapat variabel, method, dan juga kelas inner. Kelas akan terdeklarasi secara otomatis ketika pembuatan file baru.

4. Method

   Method merupakan bagian didalam program yang tugasnya menjelasan tingkah laku dari objek. Method dibagi menjadi tiga kategori yaitu :

   a. Konstruktor : method yang dieksekusi pertama kali setelah method main.
   b. Fungsi / prosedur : fungsi berguna mengembalikan suatu nilai. Sedangkan prosedur tidak mengembalikan suatu nilai.
   c. *Main* : method utama yang pertama kali dipanggil untuk menjalankan program.