BAB II
LANDASAN TEORI

2.1 Data Mining

Secara sederhana, *data mining* merupakan ekstraksi informasi yang tersirat dalam sekumpulan data. Data mining merupakan sebuah proses untuk menggali kumpulan data dan menemukan informasi di dalamnya. (Turban, E., dkk. 2005). Data mining merupakan proses pengekstrakan informasi dari jumlah kumpulan data yang besar dengan menggunakan algoritma dan teknik gambar dari statistik, mesin pembelajaran dan sistem manajemen *database*. Penggalian data ini dilakukan pada sekumpulan data yang besar untuk menemukan pola atau hubungan yang ada dalam kumpulan data tersebut (Kusrini dan E.T. Lutfi. 2009). Hasil penemuan yang diperoleh setelah proses penggalian data ini, kemudian dapat digunakan untuk analisis yang lebih lanjut.

Data mining yang disebut juga dengan *Knowledge-Discovery in Database* (KDD) adalah sebuah proses secara otomatis atas pencarian data di dalam sebuah memori yang amat besar dari data untuk mengetahui pola dengan menggunakan alat seperti klasifikasi, hubungan (*association*) atau pengelompokan (*clustering*). Proses KDD ini terdiri dari langkah-langkah sebagai berikut (Han, J. dan M. Kamber. 2006):

1. *Data Cleaning*, proses menghapus data yang tidak konsisten dan kotor.
2. *Data Integration*, penggabungan beberapa sumber data.
3. *Data Selection*, pengambilan data yang akan dipakai dari sumber data.
4. *Data Transformation*, proses dimana data ditransformasikan menjadi bentuk yang sesuai untuk diproses dalam data mining.
5. *Data Mining*, suatu proses yang penting dengan melibatkan metode untuk menghasilkan suatu pola data.
6. *Pattern Evaluation*, proses untuk menguji kebenaran dari pola data yang mewakili *knowledge* yang ada didalam data itu sendiri.

7. *Knowledge Presentation*, proses visualisasi dan teknik menyajikan *knowledge* digunakan untuk menampilkan *knowledge* hasil *mining* kepada *user*.

2.2 Metode Data Mining

Pada umumnya metode *data mining* dapat dikelompokkan kedalam dua kategori yaitu *deskriptif* dan *prediktif*. Metode *deskriptif* bertujuan untuk mencari pola yang dapat dimengeti oleh manusia yang menjelaskan karakteristik dari data. Metode *prediktif* menggunakan ciri-ciri tertentu dari data. Pada umumnya metode *data mining* dapat dikelompokkan kedalam dua untuk melakukan prediksi.

Metode-metode yang ada dalam *data mining* adalah sebagai berikut (Tang, ZhaoHui and J. MacLennan. 2005):

1. *Classification*

   Klasifikasi (*Classification*) merupakan proses untuk menemukan sekumpulan model yang menjelaskan dan membedakan kelas-kelas data, sehingga model tersebut dapat digunakan untuk memprediksi nilai suatu kelas yang belum diketahui pada sebuah objek. Untuk mendapatkan model, kita harus melakukan analisis terhadap data latih (*training set*). Sedangkan data uji (*test set*) digunakan untuk mengetahui tingkat akurasi dari model yang dihasilkan. Klasifikasi dapat digunakan untuk memprediksi nama atau nilai kelas dari suatu obyek data. Metode inilah yang digunakan dalam tugas akhir ini.

2. *Clustering*

   Pengelompokan (*Clustering*) merupakan proses untuk melakukan segmentasi. Digunakan untuk melakukan pengelompokan secara alami terhadap atribut suatu set data, termasuk kedalam *supervised task*. Contoh *clustering* seperti mengelompokkan dokumen berdasarkan topiknya.

3. *Assosiation*

   Tujuan dari metode ini untuk menghasilkan sejumlah *rule* yang menjelaskan sejumlah data yang berhubung kuat satu dengan yang

lainnya. Sebagai contoh *assosiation analysis* dapat digunakan untuk menentukan produk yang datang secara bersamaan oleh banyak pelanggan, atau bisa juga disebut dengan *basket analysis*.

4. *Regression*

   Regression mirip dengan klasifikasi. Perbedaan utamanya adalah terletak pada atribut yang diprediksi berupa nilai yang kontinyu.

5. *Forecasting*

   Prediksi (*Forecasting*) berfungsi untuk melakukan kejadian yang akan datang berdasarkan data sejarah yang ada.

6. *Sequence Analysis*

   Tujuan dari metode ini adalah untuk mengenali pola dari data diskrit. Sebagai contoh adalah menemukan kelompok gen dengan tingkat ekpresi yang mirip.

7. *Deviation Analysis*

   Tujuan dari metode ini adalah untuk menemukan penyebab perbedaan antara data yang satu dengan data yang lain dan biasa disebut sebagai *oulier detection*. Sebagai contoh adalah apakah sudah terjadi penipuan terhadap pengguna kredit dengan melihat catatan transaksi yang tersimpan dalam basis data perusahaan kartu kredit.

2.3 Klasifikasi

Klasifikasi merupakan suatu pekerjaan menilai objek data untuk memasukkannya ke dalam kelas tertentu dari sejumlah kelas yang tersedia. Dalam klasifikasi ada dua pekerjaan utama yang dilakukan, yaitu (1) pembangunan model sebagai prototipe untuk disimpan sebagai memori dan (2) penggunaan model tersebut untuk melakukan pengenalan/klasifikasi /prediksi pada suatu objek data lain agar diketahui dikelas mana objek data tersebut dalam model yang sudah disimpannya. Model dalam klasifikasi mempunyai arti yang sama denga kotak hitam, dimana ada suatu model yang menerima masukan, kemudian mampu melakukan pemikiran terhadap

masukan tersebut dan memberikan jawaban sebagai keluaran dari hasil pemikiannya. (Prasetyo, E. 2012).

Tahapan dari klasifikasi dalam data mining terdiri dari(Han, J. dan M. Kamber. 2006) :

1. Pembangunan Model

Pada tahapan ini dibuat sebuah model untuk menyelesaikan masalah klasifikasi class atau atribut dalam data. Tahap ini merupakan fase pelatihan, dimana data latih dianalisis menggunakan algoritma klasifikasi, sehingga model pembelajaran direpresentasikan dalam bentuk aturan klasifikasi.

2. Penerapan Model

Pada tahapan ini model yang sudah dibangun sebelumnya digunakan untuk menentukan atribut/kelas dari sebuah data baru yang atribut/kelasnya belum diketahui sebelumnya. Tahap ini digunakan untuk memperkirakan keakuratan aturan klasifikasi terhadap data uji. Jika model dapat diterima, maka aturan dapat diterapkan terhadap klasifikasi data baru.

2.4 Decision Tree

2.4.1 Pengertian Decision Tree

Decision tree merupakan metode klasifikasi *data mining*. *Decision tree* dalam istilah pembelajaran merupakan sebuah struktur pohon dimana setiap *node* pohon mempresentasikan atribut yang telah diuji. Setiap cabang merupakan suatu pembagian hasil uji dan *node* daun (*leaf*) mempresentasikan kelompok kelas tertentu. (Jianwei, Han. 2001). Level *node* teratas dari sebuah *Decision Tree* adalah node akar (*root*) yang biasanya berupa atribut yang paling memiliki pengaruh terbesar pada suatu kelas tertentu. Pada umumnya *Decision Tree* melakukan strategi pencarian secara *top-down* untuk solusinya. Pada proses mengklasifikasi data yang tidak diketahui, nilai atribut akan diuji dengan cara melacak jalur dari *node* akar (*root*) sampai *node* akhir (daun) dan kemudian akan diprediksi kelas yang dimiliki oleh suatu data baru tertentu. (Santoso, Budi. 2007).

2.4.2 Jenis - Jenis Decision Tree

Beberapa model *decision tree* yang sudah dikembangkan antara lain C4.5 atau ID3 dan CART. Berikut ini akan dijelaskan model dari decision tree tersebut :

1. C4.5 atau ID3

   Decision Tree menggunakan algoritma ID3 atau C4.5, yang diperkenalkan dan dikembangkan pertama kali oleh Quinlan yang merupakan singkatan dari *Iterative Dichotomiser 3* atau *Induction of Decision 3.* Algoritma ID3 membentuk pohon keputusan dengan metode *divide-and-conquer* data secara rekursif dari atas ke bawah. Strategi pembentukan Decision Tree dengan algoritma ID3 adalah:

   A. Pohon dimulai sebagai *node* tunggal (akar/*root*) yang merepresentasikan semua data.
   B. Sesudah *node root* dibentuk, maka data pada *node* akar akan diukur dengan *information gain* untuk dipilih atribut mana yang akan dijadikan atribut pembaginya.
   C. Sebuah cabang dibentuk dari atribut yang dipilih menjadi pembagi dan data akan didistribusikan ke dalam cabang masing-masing.
   D. Algoritma ini akan terus menggunakan proses yang sama atau bersifat rekursif untuk dapat membentuk sebuah *Decision Tree*. Ketika sebuah atribut telah dipilih menjadi *node* pembagi atau cabang, maka atribut tersebut tidak diikutkan lagi dalam penghitungan nilai *information gain.*
   E. Proses pembagian rekursif akan berhenti jika salah satu dari kondisi dibawah ini terpenuhi :
      a. Semua data dari anak cabang telah termasuk dalam kelas yang sama.
      b. Semua atribut telah dipakai, tetapi masih tersisa data dalam kelas yang berbeda. Dalam kasus ini, diambil data yang mewakili kelas yang terbanyak untuk menjadi label kelas pada *node* daun. Tidak terdapat data pada anak cabang yang baru. Dalam kasus ini, *node*

daun akan dipilih pada cabang sebelumnya dan diambil data yang mewakili kelas terbanyak untuk dijadikan label kelas.

Metode C4.5 dan ID3 memiliki perbedaan dalam nilai tiap atribut. Metode C4.5 menggunakan atribut yang bernilai kategorikal dan numerikal, sedangkan metode ID3 menggunakan atribut yang bernilai kategorikal. Metode *decision tree C4.5* inilah yang digunakan dalam tugas akhir ini.

2. CART

CART adalah singkatan dari *Classification And Regression Tree*. Dalam CART ada dua langkah penting yang harus diikuti untuk mendapatkan *tree* dengan performansi yang optimal. Yang pertama adalah pemecahan objek secara berulang berdasarkan atribut tertentu. Yang kedua, *prunning* (pemangkasan) dengan menggunakan data validasi.

Misalkan kita mempunyai variabel independent x1,x2,x3,.....xn dan variabel dependent atau output y. Pemecahan secara berulang berarti kita bagi objek ke dalam kotak-kotak berdasarkan nilai variabel x1,x2 atau xr. Cara ini diulang sehingga dalam suatu kotak sebisa mungkin berisi observasi dalam kelompok atau kelas yang sama.

Langkah berikutnya sesudah dilakukan pemecahan objek atau data secara berulang adalah melakukan *prunning*. Dalam *prunning* kita ingin memangkas *tree* yang mungkin terlalu besar dan terjadi fenomena *overfitting*. *Overvitting* merupakan sebuah satu buah pengelompokkan yang mungkin hanya berisi satu data yang memungkinkan data tersebut merupakan *noise* yang ada di data training dan bukan pola yang mungkin terjadi dalam data testing atau data validasi. *Prunning* terdiri dari beberapa langkah pemilihan secara berulang simpul yang akan dijadikan simpul daun. Dengan mengubah simpul menjadi simpul daun artinya tidak akan dilakukan pemecahan lagi sesudah itu. Dengan demikian ukuran *tree* akan berkurang. (Santoso, Budi. 2007).

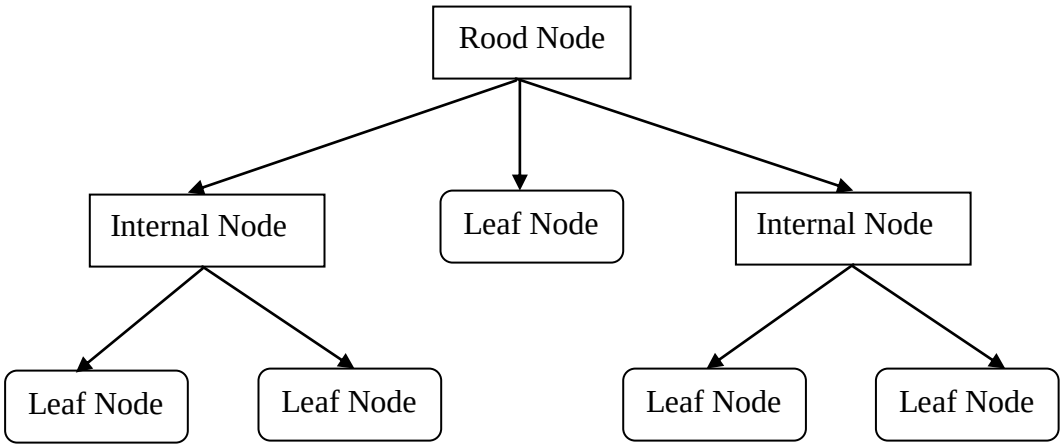
2.4.3 Model Decision Tree

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*.

Selain karena pembangunannya relatif cepat, hasil dari model yang dibangun mudah untuk dipahami. Pada *decision tree* terdapat 3 jenis *node*, yaitu:

a. *Root Node*, merupakan *node* paling atas, pada *node* ini tidak ada *input* dan bisa tidak mempunyai *output* atau mempunyai *output* lebih dari satu.
b. *Internal Node*, merupakan *node* percabangan, pada *node* ini hanya terdapat satu *input* dan mempunyai *output* minimal dua.
c. *Leaf node* atau *terminal node*, merupakan *node* akhir, pada *node* ini hanya terdapat satu *input* dan tidak mempunyai *output*.

Contoh dari model pohon keputusan yaitu seperti pada **gambar 2.1** berikut:



Gambar 2.1 Model *Decision Tree*

2.5 Donor Darah

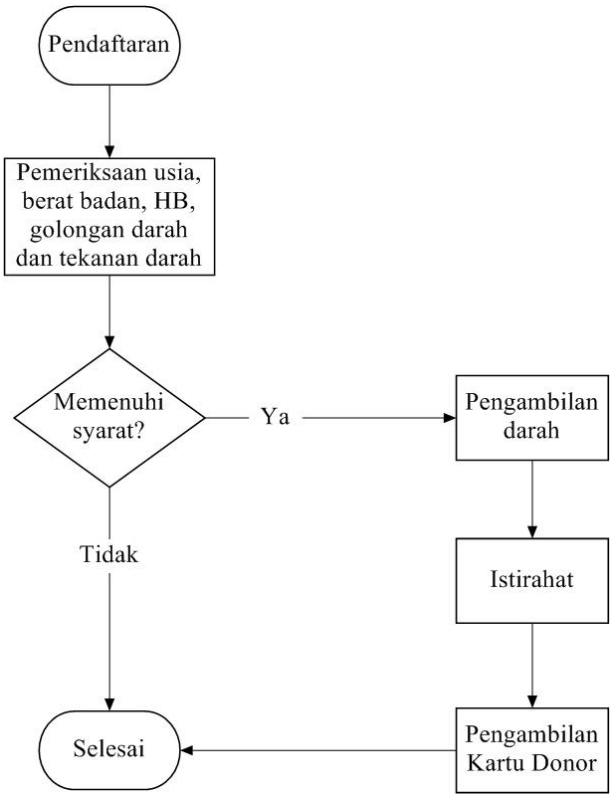
Donor darah merupakan proses pengambilan darah dari seseorang secara sukarela untuk disimpan di bank darah untuk kemudian dipakai pada transfusi darah bagi pasien yang membutuhkan.

Untuk dapat menyumbangkan darah, seorang donor darah harus memenuhi syarat sebagai berikut :

1. Berbadan sehat
2. Usia 17-60 tahun (pada usia 17 tahun diperbolehkan menjadi donor bila mendapat ijin tertulis dari orang tua. Sampai usia tahun donor masih dapat menyumbangkan darahnya dengan jarak penyumbangan 3 bulan atas pertimbangan dokter).
3. Berat badan minimum 45 kg.
4. Temperatur tubuh : 36,6 - 37,5 ℃ (oral).
5. Tekanan darah baik, yaitu:
   a. Sistole :110 - 160 mm Hg.
   b. Diastole : 60 - 100 mm Hg.
6. Denyut nadi : Teratur 50 - 100 kali/ menit.
7. Haemoglobin
   a. Wanita : minimal 12 gr %
   b. Pria : minimal 12,5 gr %
8. Jumlah penyumbangan pertahun paling banyak 4 kali dengan jarak penyumbangan sekurang-kurangnya 3 bulan. Keadaan ini harus sesuai dengan keadaan umum donor.
9. Bagi penyumbang darah wanita tidak sedang menstruasi, hamil atau menyusui.
10. Tidak dalam pengaruh obat-obatan seperti golongan narkotika dan alkohol.
11. Tidak menderita penyakit: jantung, hati, paru-paru, ginjal, kencing manis, penyakit kelainan darah, gangguan pembekuan darah, epilepsi, kanker atau penyakit kulit.

2.6 Prosedur Donor Darah

Sebelum dilakukan proses pengambilan darah, calon pendonor darah akan melalui beberapa tahap. Berikut ini akan dijelaskan secara lebih detail alur donor darah menggunakan *flowchart* yang disajikan pada **gambar 2.2**.



Gambar 2.2 *Flowchart* prosedur donor darah

Tahap pertama bagi calon pendonor darah yaitu mengambil *form* pendaftaran dan mengisikan data pribadi. Kemudian dilakukan pemeriksaan usia, berat badan, kadar HB, golongan darah dan tekanan darah oleh petugas UDD PMI. Jika memenuhi syarat, tahap berikutnya yaitu pengambilan darah. Sedangkan bagi yang tidak memenuhi syarat donor darah, maka pengambilan darah tidak dapat dilakukan. Setelah selesai, pendonor darah diberi waktu beristirahat sejenak. Apabila keadaan sudah membaik, petugas UDD PMI mempersilahkan pendonor darah untuk mengambil kartu donor. Kartu tersebut adalah bukti bahwa seseorang telah melakukan donor darah. Proses donor darahpun dinyatakan selesai.

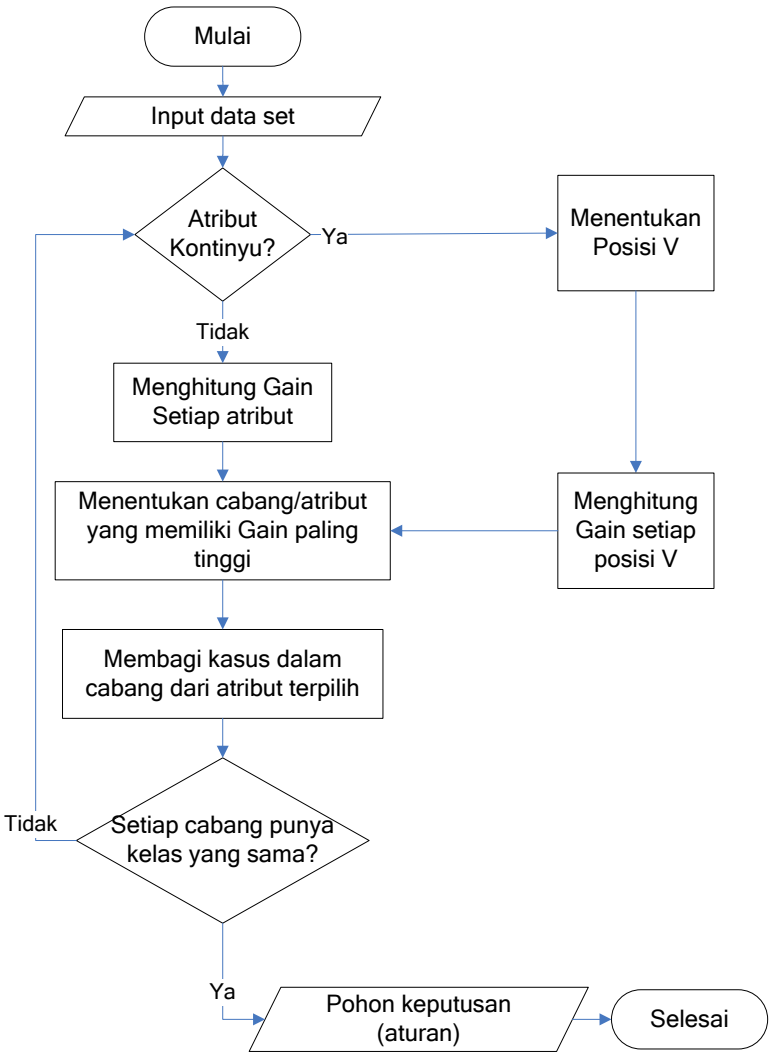
2.7 Algoritma Decision Tree C4.5

Secara umum algoritma C4.5 untuk membangun pohon keputusan adalah sebagai berikut:

1. Pilih atribut sebagai akar.

2. Buat cabang untuk tiap-tiap nilai.
3. Bagi kasus dalam cabang.
4. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama.

Berikut ini akan dijelaskan secara lebih detail algoritma C4.5 menggunakan *flowchart* yang disajikan pada **gambar 2.3**.



Gambar 2.3 *Flowchart* algoritma *Decision Tree C4.5*

Untuk memilih atribut sebagai simpul akar (*root node*) atau simpul dalam (*internal node*), didasarkan pada nilai *information gain* tertinggi dari atribut-atribut yang ada. Sebelum perhitungan *information gain*, akan dilakukan perhitungan *entropy*. *Entropy* merupakan distribusi probabilitas

dalam teori informasi dan diadopsi kedalam algoritma *C*4.5 untuk mengukur tingkat homogenitas distribusi kelas dari sebuah himpunan data (*data set*). Semakin tinggi tingkat *entropy* dari sebuah data maka semakin homogen distribusi kelas pada data tersebut. Perhitungan *information gain* menggunakan rumus 2.1, sedangkan *entropy* menggunakan rumus 2.2.

$$Gain(S,A) = Entropy\ (S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} * Entropy(S_i) \quad .....(2.1)$$

dimana,

S : Himpunan kasus

A : Atribut

n : Jumlah partisi atribut A

|Si|: Jumlah kasus pada partisi ke i

|S| : Jumlah kasus dalam S

$$Entropy(S) = -\sum_{i=1}^{n} p_i * log_2 p_i \quad .....(2.2)$$

dimana,

S : Himpunan kasus

n : Jumlah partisi S

p_i : Proporsi dari S_i terhadap S

Selain *Information Gain* kriteria yang lain untuk memilih atribut sebagai pemecah adalah *Rasio Gain*. Perhitungan rasio gain menggunakan rumus 2.3, sedangkan split information menggunakan rumus 2.4.

$$GainRasio\ (S,A) = \frac{Gain\ (S,A)}{SplitInf\ ormation\ (S,A)} \quad .....(2.3)$$

$$SplitInformation(S,A) = -\sum_{i=1}^{c} \frac{S_i}{S} log_2 \frac{S_i}{S} \quad .....(2.4)$$

dimana S1 sampai Sc adalah c subset yang dihasilkan dari pemecahan S dengan menggunakan atribut A yang mempunyai sebanyak c nilai.

2.8 Contoh Perhitungan

Berikut ini akan dijelaskan ilustrasi dari alur proses perhitungan algoritma *Decision Tree C4.5*. Data set yang digunakan pada contoh ini adalah data untuk melakukan prediksi "apakah harus bermain *baseball* ?" dengan menjawab Ya atau Tidak. Atribut yang digunakan ada 4 yaitu cuaca, suhu, kelembaban, dan angin. Dimana atribut suhu dan kelembaban bertipe kontinyu sedangkan cuaca dan angin bertipe kategorikal. Sedangkan kolom bermain adalah kelas tujuannya atau label kelasnya.

Tabel 2.1 Contoh data set

Cuaca	Suhu	Kelembaban	Angin	Bermain
Cerah	85	85	Biasa	Tidak
Cerah	80	90	Kencang	Tidak
Mendung	83	78	Biasa	Ya
Hujan	70	96	Biasa	Ya
Hujan	68	80	Biasa	Ya
Hujan	65	70	Kencang	Tidak
Mendung	64	65	Kencang	Ya
Cerah	72	95	Biasa	Tidak
Cerah	69	70	Biasa	Ya
Hujan	75	80	Biasa	Ya
Cerah	75	70	Kencang	Ya
Mendung	72	90	Kencang	Ya
Mendung	81	75	Biasa	Ya
Hujan	71	80	Kencang	Tidak

Proses pertama adalah menghitung *entropy* untuk *node* akar (semua data) terhadap komposisi kelas.

Berikut adalah perhitungan *entropy* untuk semua data:

$$Entropy(S) = -\frac{9}{14} * log_2\left(\frac{9}{14}\right) - \frac{5}{14} * log_2\left(\frac{5}{14}\right)$$

$$= 0.9403$$

Selanjutnya, untuk fitur yang bertipe *numeric*, harus ditentukan posisi *v* yang terbaik untuk pemecahan. Dalam contoh ini, digunakan pemecahan biner. Hasil uji coba pada fitur Suhu dengan menghitung nilai Gainnya disajikan pada tabel 2.2. Nilai Gain tertinggi didapatkan pada posisi *v* = 70. Maka untuk fitur Suhu dilakukan diskretisasi pada *v* = 70 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.2 Posisi *v* untuk pemecahan fitur Suhu di *node* akar

Suhu	**70**		**75**		**80**	
	<=	>	<=	>	<=	>
Ya	4	5	7	2	7	2
Tidak	1	4	3	2	4	1
Gain	**0.0453**		**0.0251**		**0.0005**	

Hasil uji coba pada fitur Kelembaban dengan menghitung nilai *Gain* yang disajikan pada tabel 2.3. Nilai *Gain* tertinggi didapatkan pada posisi *v* = 80. Maka untuk fitur Kelembaban dilakukan diskretisasi pada *v* = 80 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.3 Posisi *v* untuk pemecahan fitur Kelembaban di *node* akar

Kelembaban	**70**		**75**		**80**		**85**	
	<=	>	<=	>	<=	>	<=	>
Ya	2	7	3	6	7	2	7	2
Tidak	1	4	1	4	2	3	3	2
Gain	**0.0005**		**0.0150**		**0.1022**		**0.0251**	

Selanjutnya dihitung entropy untuk setiap nilai fitur terhadap kelas, kemudian dihitung gain untuk setiap fitur. Hasilnya disajikan pada tabel 2.4.

Tabel 2.4 Hasil perhitungan *entropy* dan *gain* untuk *node* akar

Node			**Jumlah**	**Ya**	**Tidak**	**Entropy**	**Gain**
1	Total		14	9	5	0.9403	
	Cuaca						0.2467
		Cerah	5	2	3	0.9710	
		Mendung	4	4	0	0	
		Hujan	5	3	2	0.9710	
	Suhu						0.0453
		<=70	5	4	1	0.7219	
		>70	9	5	4	0.9911	
	Kelembaban						0.1022
		<=80	9	7	2	0.7642	
		>80	5	3	2	0.9710	
	Angin						0.0481
		Pelan	8	6	2	0.8113	
		Kencang	6	3	3	0.8113	

Hasil yang didapat di tabel 2.4 menunjukkan bahwa *Gain* tertinggi ada di fitur Cuaca, maka Cuaca dijadikan sebagai *node* akar. Selanjutnya, dihitung

posisi split untuk fitur Cuaca dengan menghitung *Rasio Gain*, selengkapnya disajikan pada tabel 2.5.

Hasil peritungan *rasio gain* posisi *split* untuk *opsi* satu sebagai berikut:

$$SplitInfo(Semua, cuaca) = \left(-\frac{5}{14} * log_2\left(\frac{5}{14}\right)\right) + \left(-\frac{4}{14} * log_2\left(\frac{4}{14}\right)\right) + \left(-\frac{5}{14} * log_2\left(\frac{5}{14}\right)\right)$$

$$= 1.5774$$

$$RasioGain(Semua, cuaca) = \frac{0.2467}{1.5774}$$

$$= 0.16$$

Dengan cara yang sama, akan didapatkan nilai *rasio gain* untuk *opsi* yang lain.

Hasil ditabel 2.5 menunjukkan bahwa *rasio gain* tertinggi ada di *opsi* 4 yaitu *split* {cerah, hujan} dengan {mendung}. Itu artinya, cabang untuk akar ada 2, yaitu: {cerah, hujan} dan {mendung}, seperti ditunjukkan pada gambar 2.3.

Tabel 2.5 Perhitungan *Rasio Gain* untuk fitur Cuaca

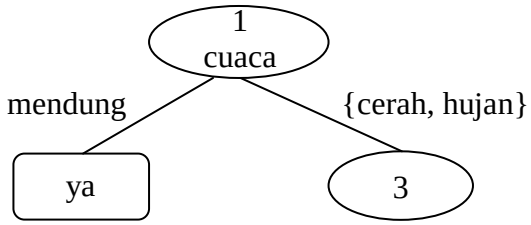
Node			**Jumlah**	**Entropy**	**Gain**	**Rasio Gain**
1	Total		14		0.2467	
Opsi 1	Cuaca	 Cerah Mendung Hujan	 5 4 5	1.5774		0.16
Opsi 2	Cuaca	 Cerah Mendung dan Hujan	 5 9	0.9403		0.26
Opsi 3	Cuaca	 Cerah, Mendung Hujan	 9 5	0.9403		0.26
Opsi 4	Cuaca	 Cerah, Hujan Mendung	 10 4	0.8631		0.29

Hasil pemisahan data menurut *node* akar disajikan pada tabel 2.6.

Tabel 2.6 Pemisahan data menurut *node* akar

Cuaca	**Suhu**	**Kelembaban**	**Angin**	**Bermain**
Cerah	85	85	Biasa	Tidak
Cerah	80	90	Kencang	Tidak
Hujan	70	96	Biasa	Ya
Hujan	68	80	Biasa	Ya
Hujan	65	70	Kencang	Tidak
Cerah	72	95	Biasa	Tidak
Cerah	69	70	Biasa	Ya
Hujan	75	80	Biasa	Ya
Cerah	75	70	Kencang	Ya
Hujan	71	80	Kencang	Tidak
Mendung	83	78	Biasa	Ya
Mendung	64	65	Kencang	Ya
Mendung	72	90	Kencang	Ya
Mendung	81	75	Biasa	Ya

Untuk *node* 2, nilai *Entropy* yang didapat adalah 0 (karena semua baris memiliki kelas yang sama) maka dipastikan bahwa node 2 menjadi daun, seperti ditunjukkan pada gambar 2.4.



Gambar 2.4 Hasil pembentukan cabang di akar untuk kasus "apakah harus bermain *baseball* ?"

Selanjutnya, di *node* 3, harus dihitung dulu *entropy* untuk sisa data terhadap komposisi kelas yang tidak masuk dalam *node* 2.

Untuk fitur yang bertipe numerik, harus ditentukan lagi posisi *v* yang terbaik untuk pemecahan. Dalam contoh ini, digunakan pemecahan biner. Hasil uji coba pada fitur Suhu dengan menghitung nilai *Gain* yang disajikan pada tabel 2.7. Nilai *Gain* tertinggi didapatkan pada posisi $v = 75$. Maka untuk fitur Suhu dilakukan diskretisasi pada $v = 75$ ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.7 Posisi *v* untuk pemecahan fitur Suhu di *node* 3

Suhu	**70**		**75**		**80**	
	<=	>	<=	>	<=	>
Ya	3	2	5	0	5	0
Tidak	1	4	3	2	4	1
Gain	**0.1245**		**0.2365**		**0.1080**	

Hasil uji coba pada fitur Kelembaban dengan menghitung nilai *Gain* yang disajikan pada tabel 2.8. Nilai *Gain* tertinggi didapatkan pada posisi *v* = 80. Maka untuk fitur Kelembaban dilakukan diskretisasi pada *v* = 80 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.8 Posisi *v* untuk pemecahan fitur Kelembaban di *node* 3

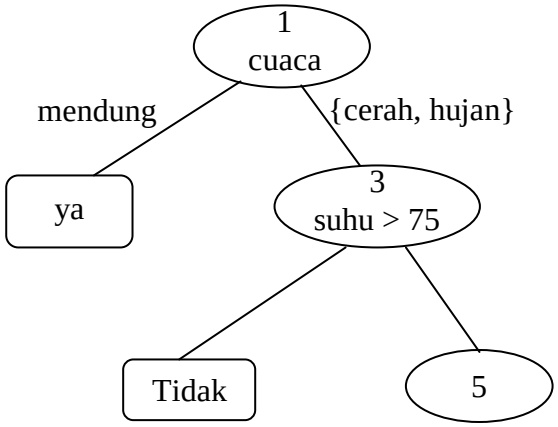
Kelembaban	**70**		**75**		**80**		**85**	
	<=	>	<=	>	<=	>	<=	>
Ya	2	3	2	3	4	3	4	1
Tidak	1	4	1	4	2	1	3	2
Gain	**0.0349**		**0.0349**		**0.1245**		**0.0349**	

Selanjutnya dihitung *entropy* untuk setiap nilai fitur terhadap kelas, kemudian dihitung *gain* untuk setiap fitur. Hasilnya disajikan pada tabel 2.9

Tabel 2.9 Hasil perhitungan *entropy* dan *gain* untuk *node* 3

Node			**Jumlah**	**Ya**	**Tidak**	**Entropy**	**Gain**
3	Total		10	5	5	1.0000	
	Cuaca						0.0290
		Cerah	5	2	3	0.9710	
		Hujan	5	3	2	0.9710	
	Suhu						0.2365
		<=75	8	5	3	0.9544	
		>75	2	0	2	0	
	Kelembaban						0.1245
		<=80	6	4	2	0.9183	
		>80	4	1	3	0.8113	
	Angin						0.1245
		Pelan	6	4	2	0.9183	
		Kencang	4	1	3	0.8113	

Hasil yang ditunjukkan pada tabel 2.9 menunjukkan bahwa *gain* tertinggi ada di fitur Suhu, berarti fitur Suhu dijadikan syarat kondisi di *node* 3, seperti ditunjukan pada gambar 2.5. Pemisahan datanya ditunjukkan pada tabel 2.10.



Gambar 2.5 Hasil pembentukan cabang di *node* 3 untuk kasus "apakah harus bermain *baseball*

Tabel 2.10 Pemisahan data menurut *node* 3

Cuaca	Suhu	Kelembaban	Angin	Bermain
Hujan	70	96	Pelan	Ya
Hujan	68	80	Pelan	Ya
Hujan	65	70	Kencang	Tidak
Cerah	72	95	Pelan	Tidak
Cerah	69	70	Pelan	Ya
Hujan	75	80	Pelan	Ya
Cerah	75	70	Kencang	Ya
Hujan	71	80	Kencang	Tidak
Cerah	85	85	Pelan	Tidak
Cerah	80	90	Kencang	Tidak
Mendung	83	78	Biasa	Ya
Mendung	64	65	Kencang	Ya
Mendung	72	90	Kencang	Ya
Mendung	81	75	Biasa	Ya

Node 4, untuk cabang suhu > 75 dimana label kelas bernilai tidak, dipastikan mempunyai entropy 0, maka *node* 4 (yang dituju) dijadikan daun. Seperti ditunjukkan pada gambar 2.4.

Selanjutnya pada *node* 5, untuk fitur numerik kembali dilakukan perhitungan posisi *v* yang terbaik untuk pemecahan. Dalam contoh ini, digunakan pemecahan biner. Hasil uji coba pada fitur Suhu dengan menghitung nilai *Gain* yang disajikan pada tabel 2.11. Nilai *Gain* tertinggi

didapatkan pada posisi *v* = 70. Maka untuk fitur Suhu dilakukan diskretisasi pada *v* = 70 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.11 Posisi *v* untuk pemecahan fitur Suhu di *node* 5

Suhu	**70**		**75**	
	<=	>	<=	>
Ya	3	2	5	0
Tidak	1	2	3	0
Gain	**0.0488**		**0**	

Hasil uji coba pada fitur Kelembaban dengan menghitung nilai *Gain* yang disajikan pada tabel 2.12. Nilai *Gain* tertinggi didapatkan pada posisi *v* = 80. Maka untuk fitur Kelembaban dilakukan diskretisasi pada *v* = 80 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.12 Posisi *v* untuk pemecahan fitur Kelembaban di *node* 5

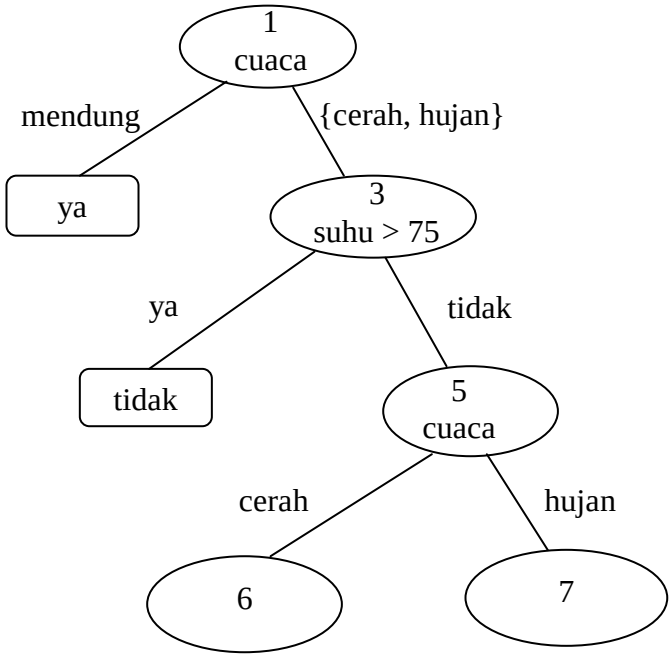
Kelembaban	**70**		**75**		**80**	
	<=	>	<=	>	<=	>
Ya	2	3	2	3	4	1
Tidak	1	2	1	2	2	1
Gain	**0.0032**		**0.0032**		**0.0157**	

Selanjutnya dihitung *entropy* untuk setiap nilai fitur terhadap kelas, kemudian dihitung *gain* untuk setiap fitur. Hasilnya disajikan pada tabel 2.13.

Tabel 2.13 Hasil perhitungan *entropy* dan *gain* untuk *node* 5

Node			**Jumlah**	**Ya**	**Tidak**	**Entropy**	**Gain**
5	Total		8	5	3	0.9544	
	Cuaca						0.2013
		Cerah	3	2	3	0.3900	
		Hujan	5	3	2	0.9710	
	Suhu						0.0488
		<=70	4	3	1	0.8113	
		>70	4	2	2	1.0000	
	Kelembaban						0.0157
		<=80	6	4	2	0.9183	
		>80	2	1	1	1.0000	
	Angin						0.1589
		Pelan	5	4	1	0.7219	
		Kencang	3	1	2	0.9183	

Hasil yang ditunjukkan pada tabel 2.13 menunjukkan bahwa *gain* tertinggi ada di fitur Cuaca, berarti fitur Cuaca dijadikan syarat kondisi di *node* 5, seperti ditunjukan pada gambar 2.6 Pemisahan datanya ditunjukkan pada tabel 2.14.



Gambar 2.6 Hasil pembentukan cabang di *node* 5 untuk kasus apakah harus bermain *baseball*

Tabel 2.14 Pemisahan data menurut *node* 5

Cuaca	**Suhu**	**Kelembaban**	**Angin**	**Bermain**
Cerah	72	95	Pelan	Tidak
Cerah	69	70	Pelan	Ya
Cerah	75	70	Kencang	Ya
Hujan	70	96	Pelan	Ya
Hujan	68	80	Pelan	Ya
Hujan	65	70	Kencang	Tidak
Hujan	75	80	Pelan	Ya
Hujan	71	80	Kencang	Tidak
Cerah	85	85	Pelan	Tidak
Cerah	80	90	Kencang	Tidak
Mendung	83	78	Biasa	Ya
Mendung	64	65	Kencang	Ya
Mendung	72	90	Kencang	Ya
Mendung	81	75	Biasa	Ya

Pada perhitungan berikutnya, fitur Cuaca tidak digunakan lagi karena kedua nilai berbeda yang tersisa sudah digunakan untuk syarat pengujian di *node* 5. Selanjutnya pada *node* 6, untuk fitur numerik kembali dilakukan perhitungan posisi *v* yang terbaik untuk pemecahan. Dalam contoh ini, digunakan pemecahan biner. Hasil uji coba pada fitur Suhu dengan menghitung nilai *Gain* yang disajikan pada tabel 2.15. Nilai *Gain* tertinggi didapatkan pada posisi *v* = 70. Maka untuk fitur Suhu dilakukan diskretisasi pada *v* = 70 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.15 Posisi *v* untuk pemecahan fitur Suhu di *node* 6

Suhu	**70**		**75**	
	<=	>	<=	>
Ya	1	1	2	0
Tidak	0	1	1	0
Gain	**0.2516**		0	

Hasil uji coba pada fitur Kelembaban dengan menghitung nilai *Gain* yang disajikan pada tabel 2.16. Nilai *Gain* didapatkan pada posisi *v* = 70. Maka untuk fitur Kelembaban dilakukan diskretisasi pada *v* = 70 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.16 Posisi *v* untuk pemecahan fitur Kelembaban di *node* 6

Kelembaban	**70**	
	<=	>
Ya	2	0
Tidak	0	1
Gain	**0.9183**	

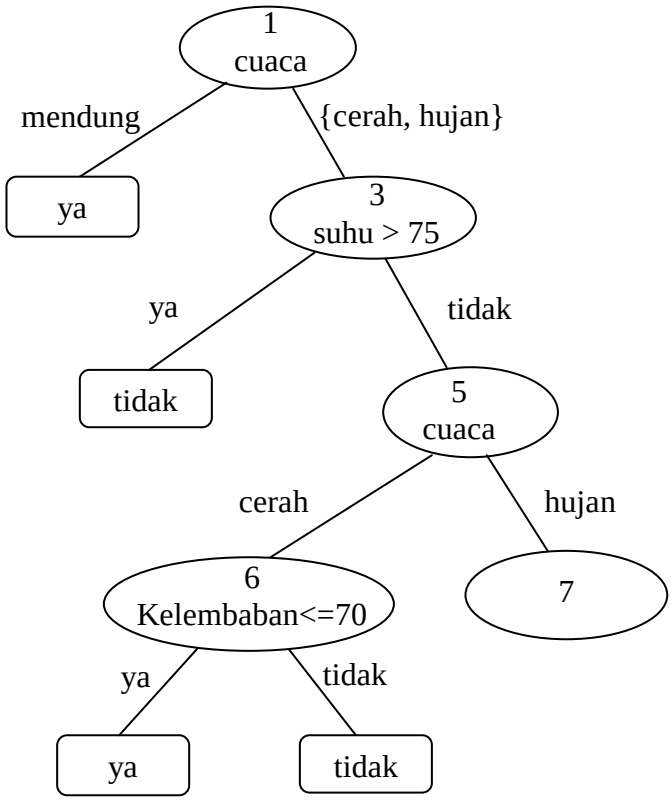
Selanjutnya dihitung *entropy* untuk setiap nilai fitur terhadap kelas, kemudian dihitung *gain* untuk setiap fitur. Hasilnya disajikan pada tabel 2.17.

Tabel 2.17 Hasil perhitungan *entropy* dan *gain* untuk *node* 6

Node			**Jumlah**	**Ya**	**Tidak**	**Entropy**	**Gain**
6	Total		3	2	1	0.9183	
	Suhu						0.2516
		<=70	1	1	0	0	
		>70	2	1	1	1.0000	
	Kelembaban						0.9183
		<=70	2	2	0	0	
		>70	1	0	1	0	

	Angin	 Pelan Kencang	 2 1	 1 1	 1 0	 1.0000 0	0.2516

Hasil yang ditunjukkan pada tabel 2.17 menunjukkan bahwa *gain* tertinggi ada di fitur Kelembaban, berarti fitur Kelembaban dijadikan syarat kondisi di *node* 6, seperti ditunjukan pada gambar 2.7. Pemisahan datanya ditunjukkan pada tabel 2.18.



Gambar 2.7 Hasil pembentukan cabang di *node* 6 untuk kasus "apakah harus bermain *baseball*

Tabel 2.18 Pemisahan data menurut *node* 6

Cuaca	**Suhu**	**Kelembaban**	**Angin**	**Bermain**
Hujan	70	96	Pelan	Ya
Hujan	68	80	Pelan	Ya
Hujan	65	70	Kencang	Tidak
Hujan	75	80	Pelan	Ya
Hujan	71	80	Kencang	Tidak
Cerah	69	70	Pelan	Ya
Cerah	75	70	Kencang	Ya
Cerah	72	95	Pelan	Tidak

Cerah	85	85	Pelan	Tidak
Cerah	80	90	Kencang	Tidak
Mendung	83	78	Biasa	Ya
Mendung	64	65	Kencang	Ya
Mendung	72	90	Kencang	Ya
Mendung	81	75	Biasa	Ya

Jika diamati tabel 2.18, untuk *node* 8 dan 9 (cabang dari *node* 6) dipastikan menjadi daun karena nilai *entropy* 0, dimana masing-masing cabang jatuh pada label kelas yang sama. Proses berikutnya dilanjutkan untuk *node* 7.

Selanjutnya pada *node* 7, untuk fitur numerik kembali dilakukan perhitungan posisi *v* yang terbaik untuk pemecahan. Dalam contoh ini, digunakan pemecahan biner. Hasil uji coba pada fitur Suhu dengan menghitung nilai *Gain* yang disajikan pada tabel 2.19. Nilai *Gain* tertinggi didapatkan hanya pada posisi *v* = 70. Maka untuk fitur Suhu dilakukan diskretisasi pada *v* = 70 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.19 Posisi *v* untuk pemecahan fitur Suhu di *node* 7

Suhu	70	
	<=	>
Ya	2	1
Tidak	0	1
Gain	**0.0200**	

Hasil uji coba pada fitur Kelembaban dengan menghitung nilai *Gain* yang disajikan pada tabel 2.20. Nilai *Gain* didapatkan hanya pada posisi *v* = 80. Maka untuk fitur Kelembaban dilakukan diskretisasi pada *v* = 80 ketika menghitung *Entropy* dan *Gain* pada semua fitur.

Tabel 2.20 Posisi *v* untuk pemecahan fitur Kelembaban di *node* 7

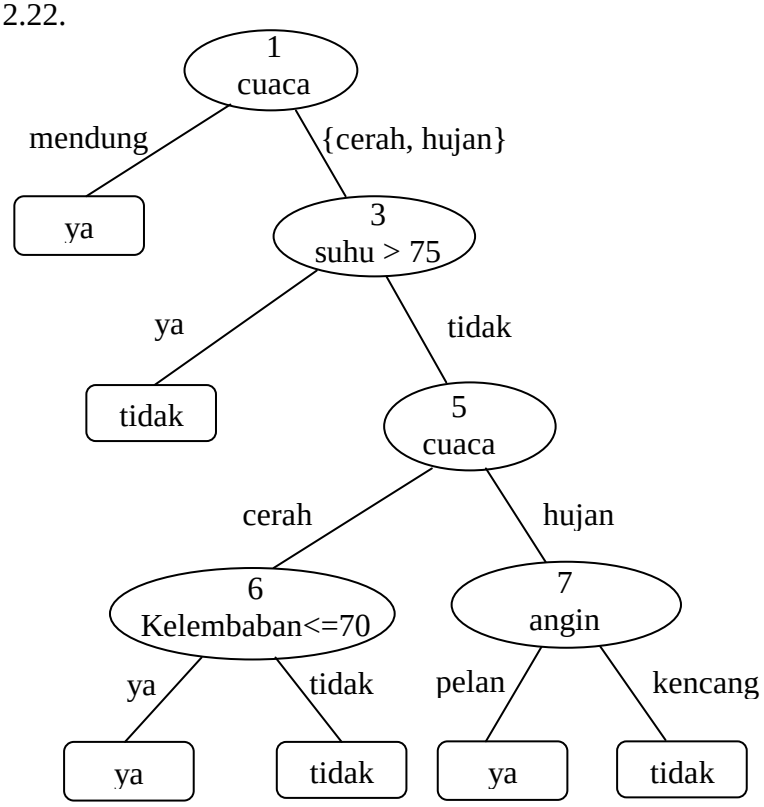
Kelembaban	**80**	
	<=	>
Ya	2	1
Tidak	2	0
Gain	**0.1710**	

Selanjutnya dihitung *entropy* untuk setiap nilai fitur terhadap kelas, kemudian dihitung *gain* untuk setiap fitur. Hasilnya disajikan pada tabel 2.21.

Tabel 2.21 Hasil perhitungan *entropy* dan *gain* untuk node 7

Node			**Jumlah**	**Ya**	**Tidak**	**Entropy**	**Gain**
7	Total		5	3	2	0.9710	
	Suhu						0.0200
		<=70	3	2	1	0.9183	
		>70	2	1	1	1.0000	
	Kelembaban						0.1710
		<=80	4	2	2	1.0000	
		>80	1	1	0	0	
	Angin						0.9710
		Pelan	3	3		0	
		Kencang	2	0	2	0	

Hasil yang ditunjukkan pada tabel 2.21 menunjukkan bahwa *gain* tertinggi ada di fitur Angin, berarti fitur Angin dijadikan syarat kondisi di *node* 7, seperti ditunjukan pada gambar 2.8. Pemisahan datanya ditunjukkan pada tabel 2.22.



Gambar 2.8 Hasil pembentukan cabang di *node* 7 untuk kasus "apakah harus bermain *baseball*

Tabel 2.22 Pemisahan data menurut *node* 7

Cuaca	Suhu	Kelembaban	Angin	Bermain
Hujan	70	96	Pelan	Ya
Hujan	68	80	Pelan	Ya
Hujan	75	80	Pelan	Ya
Hujan	65	70	Kencang	Tidak
Hujan	71	80	Kencang	Tidak
Cerah	69	70	Pelan	Ya
Cerah	75	70	Kencang	Ya
Cerah	72	95	Pelan	Tidak
Cerah	85	85	Pelan	Tidak
Cerah	80	90	Kencang	Tidak
Mendung	83	78	Biasa	Ya
Mendung	64	65	Kencang	Ya
Mendung	72	90	Kencang	Ya
Mendung	81	75	Biasa	Ya

Jika diamati tabel 2.21, untuk *node* 10 dan 11 (cabang dari *node* 7) dipastikan menjadi daun karena nilai *entropy* 0, dimana masing-masing cabang jatuh pada label kelas yang sama.

Karena tidak ada lagi node yang harus diproses, maka induksi *decision tree* dinyatakan selesai. Hasil akhir *decision tree* seperti disajikan pada gambar 2.7.

Bentuk aturan *IF THEN* untuk *decision tree* sebagai berikut:

IF cuaca= mendung *THEN playball* = ya

IF cuaca= {cerah, hujan} *AND* suhu > 75 *THEN playball* = tidak

IF cuaca= cerah *AND* suhu <=75 *AND* kelembaban<=70 *THEN playball* = ya

IF cuaca=cerah *AND* suhu<=75 *AND* kelembaban >70 *THEN playball*= tidak

IF cuaca= hujan *AND* suhu <=75 *AND* angin = pelan *THEN playball* = ya

IF cuaca= hujan *AND* suhu<=75 *AND* angin= kencang *THEN playball* = tidak

2.9 Penelitian Sebelumnya

Penelitian sebelumnya yang menggunakan metode *decision tree C4.5* adalah penelitian yang berjudul *"Sistem Prediksi Prestasi Akademik Mahasiswa Menggunakan Metode Decision Tree C4.5"* oleh Aunur Rasyid. Adapun data yang diambil dalam penelitian ini adalah data mahasiswa

Teknik Informatika Universitas Muhammadiyah Gresik semester 6 angkatan 2010 sebanyak 98 data. Atribut yang digunakan adalah instansi sekolah asal (SMA/SMK/MA), status sekolah asal (Negeri/Swasta), jurusan sekolah asal (IPA/IPS/Bahasa/Teknik/Administrasi), nilai rata-rata UN, status kerja (Sudah/Belum), dan pihak yang mempengaruhi mahasiswa dalam memilih kuliah (Sendiri/Orang Tua/Orang Lain). Pengujian sistem dilakukan sebanyak dua belas kali percobaan. Pohon keputusan terbaik yang digunakan untuk memprediksi prestasi akademik mahasiswa adalah pohon keputusan pada percobaan ke-8 dengan nilai akurasi 90%.

Liliana Swastina melakukan penelitian dengan judul *"Penerapan Algoritma C4.5 Untuk Penentuan Jurusan Mahasiswa"*. Setelah dilakukan *preprocessing*, data dihitung dengan metode *decision tree C4.5*. Adapun data yang diambil dalam penelitian ini adalah data mahasiswa baru STMIK Indonesia Banjarmasin tahun 2008 s.d 2009. Atribut yang digunakan adalah atribut nama, jenis kelamin, umur, asal sekolah, jurusan asal sekolah, nilai UAN, IPK semester 1, IPK semester 2. Sebanyak 90 % data akan digunakan untuk membangun struktur pohon keputusan melalui metode C4.5. Sedangkan 10 % lainnya digunakan sebagai data uji. Uji pertama melalui data sample yaitu data angkatan 2008, field data NRP, nama, tempat lahir dan tanggal lahir dihilangkan untuk mendapatkan akurasi yang lebih tinggi. Selain itu, Untuk membentuk pohon keputusan maka atribut IPK Semester 1 dan IPK Semester 2 perlu di klasifikasi menjadi: IPK>= 3,00 tergolong kelas A, IPK>= 2,75 tergolong kelas B, dan IPK< 2,75 tergolong kelas C. Kesimpulan dari penelitian tersebut, yakni: dari hasil uji, algoritma *decision tree C4.5* memprediksi lebih akurat dari pada algoritma *naive bayes* dalam penentuan kesesuaian jurusan dan rekomendasi jurusan mahasiswa dan algoritma *Decision Tree C4.5* akurat diterapkan untuk penentuan kesesuaian jurusan mahasiswa dengan tingkat keakuratan 93,31 % dan akurasi rekomendasi jurusan sebesar 82,64%.

Penelitian lain tentang metode *Decision Tree* dilakukan oleh Selly Artaty Zega yang berjudul "*Penggunaan Pohon Keputusan untuk Klasifikasi*

Tingkat Kualitas Mahasiswa Berdasarkan Jalur Masuk Kuliah". Adapun data yang diambil dalam penelitian ini adalah mahasiswa Politeknik Negeri Batam Program Studi Teknik Informatika angkatan 2007 s.d 2009 sebanyak 331 data. Atribut yang digunakan adalah NIM, data jalur masuk mahasiswa, dan data akademis mahasiswa yang meliputi: IP semester 1, IPK, Surat Peringatan (SP), mata kuliah yang mengulang, tidak naik tingkat dan waktu tunggu kerja (2007). Metode yang digunakan dalam pemilihan *data training* dan *data testing* adalah *K-fold cross validation.* Sehingga jumlah data untuk setiap iterasi = 33 data, kecuali pada S10 jumlah data = 34 data. Kesimpulan dari penelitian tersebut,yakni: Proses *learning* dan *classification* menghasilkan 2 model *decision tree* yang memenuhi persyaratan, yaitu iterasi 4 dan 7. Berdasarkan model *decision tree* iterasi ke 7, PMDK memiliki persentase yang lebih besar dalam menghasilkan mahasiswa yang berkualitas yaitu sebesar 90%, sedangkan melalui UMPB sebesar 78,96%. Hal ini sesuai dengan hasil angket bahwa 7 dari 13 responden memilih lajur masuk PMDK yang menghasilkan mahasiswa berkualitas. Berdasarkan model *decision tree* iterasi ke 4, PMDK memiliki persentase yang lebih kecil dalam menghasilkan mahasiswa yang berkualitas yaitu sebesar 57,89%, sedangkan melalui UMPB sebesar 80,77%. Hal ini bertolak belakang dengan hasil klasifikasi menggunakan *rule* iterasi 7. Jalur masuk kuliah memiliki pengaruh dalam mengklasifikasikan tingkat kualitas mahasiswa, namun hanya untuk penggunaan *rule* iterasi 7. Berdasarkan *decision tree* iterasi 4 dan 7 terlihat bahwa jalur masuk PMDK memiliki presentasi yang lebih besar dalam menghasilkan mahasiswa yang berkualitas yaitu 90% pada iterasi 7 dibandingkan dengan UMPB pada iterasi 4 sebesar 80,77%.