# BAB II

# LANDASAN TEORI

## 2.1. Sistem Pakar

### 2.1.1. Pengertian Sistem Pakar

Penggunaan konsep kecerdasan buatan, pada umumnya dilakukan untuk membuat software dalam bidang sistem pakar, yaitu suatu program yang bertindak sebagai penasehat atau konsultasi pintar. Sistem pakar merupakan sistem komputer dasar yang menggabungkan pengetahuan, fakta-fakta dan teknik penelusuran untuk memecahkan masalah yang biasanya memerlukan keahlian dari seorang pakar. Secara umum proses yang terjadi di dalam sistem pakar merupakan pengumpulan, representasi dan penyimpanan pengetahuan sistem pakar ke dalam komputer dan selanjutnya pengetahuan diakses oleh pemakai.

Salah satu aplikasi program kecerdasan buatan, sistem pakar menggabungkan pangkalan pengetahuan dengan sistem *inferensi*, berusaha menduplikasi fungsi seorang pakar dalam bidang keahlian tertentu. Sistem tidak bertujuan mengganti kedudukan seorang pakar, tetapi memasyarakatkan pengetahuan dan pengalaman seorang pakar.

### 2.1.2. Konsep Umum Sistem Pakar

Pengentahuan yang dimiliki system pakar direpresentasikan dalam beberapa cara. Salah satu metode yang paling umum digunakan adalah tipe *rule*s menggunakan format IF THEN. Banyak system pakar yang dibangun dengan mengekspresikan pengetahuan dalam bentuk *rules*. Bahkan,

pendekatan berbasis pengetahuan (*knowledge-base approach*) untuk membangun system pakar telah mamatahkan pendekatan awal yang digunakan pada sekitar tahun 1950-an dan 1960-an yang menggunakan teknik penalaran yang tidak mengandalkan pengetahuan.

Pengetahuan tidak tertulis yang dimiliki oleh seorang pakar harus diekstrasi melalui wawancara secara ekstensif oleh *knowledge engineer*. Proses pengembangan sistem pakar yang berhubungan dengan perolehan pengetahuan dari pakar maupun sumber lain dan kodingnya disebut sebagai *knowledge engineering* yang dilaksanakan oleh *knowledge engineer*.

Tahap awal, *knowledge* engineer melakukan diskusi dengan pakar untuk mengumpulkan pengetahuan yang dimiliki pakar yang bersangkutan. Tahap serupa dengan diskusi persyaratan atau kebutuhan yang dilakukan *system engineer* pada sistem konvensional dengan kliennya. Setelah itu *knowledge engineer* melakukan koding pengetahuan secara eksplisit ke dalam *knowledge base*. Pakar kemudian mengevaluasi system pakar dan memberikan kritik. Proses ini berlangsung secara interaktif hingga dinilai sesui oleh pakar.

System pakar umumnya dirancang dengan cara berbeda dengan system konvensional lain, terutama karena masalah yang dihadapi umumnya tidakmemiliki solusi algoritmik dan bergantung pada inferensi untuk mendapatkan solusi yang terbaik yang paling mungkin (*reasonable*). Oleh karena itu system pakar harus mampu menjelaskan inferensi yang dilakukan sehingga hasil yang diperoleh dapat diperiksa.

### 2.1.3. Manfaat dan kekurangan Sistem Pakar

**a. Manfaat Sistem Pakar**

System pakar menjadi sangat popular karena sangat banyak kemampuan dan manfaat yang diberikannya(T.Sutojo 2010), di antaranya:

1. Meningkatkan produktivitas, karena system pakar dapat bekerja lebih cepat daripada manusia.
2. Mebuat seseorang yang awam bekerja seperti layaknya seorang pakar.
3. Meningkatkan kualitas, dengan memberi nasehat yang konsisten dan mengurangi kesalahan.
4. Mampu menangkap pengetahuan dan kepakaran seseorang.
5. Memudahkan akses pengetahuan seorang pakar.
6. Bias digunakan sebagai media pelengkap dalam pelatihan. Pengguna pemula yang bekerja dengan system pakar akan menjadi lebih berpengalaman karena adanya fasilitas penjelas yang berfungsi sebagai guru.
7. Meningkatkan kemampuan untuk menyelesaikan masalah karena system pakar mengambil sumber pengetahuan dari banyak pakar.

**b. Kekurangan Sistem Pakar**

Selain manfaat, ada juga beberapa kekurangan yang ada Sistem Pakar, diantaranya:

1. Biaya yang sangat mahal untuk membuat dan memeliharanya.

2. Sulit dikembangkan karena keterbatasan keahlian dan ketersediaan pakar.
3. System pakar tidak 100% bernilai benar.

### 2.1.4. Ciri-ciri Sistem Pakar

Ciri-ciri sitem pakar adalah sebagai berikut:

1. Terbatas pada *domain* keahlian tertentu.
2. Dapat memberikan panalaran untuk data yang tidak pasti.
3. Dapat mengemukakan rangkaian alasan yang diberikannya dengan cara yang dapat dipahami.
4. Bardasarkan kaidah atau *rule* tertentu.
5. Dirancang untuk dapat dikembangkan secara bertahap.
6. Pengetahuan dan mekanisme *inferensi* jelas terpisah.
7. Keluarannya bersifat anjuran.
8. System dapat mengaktifkan kaidah secara searah yang sesui yang dituntun oleh dialog dengan pemakai.

### 2.1.5. Klasifikasi Sistem Pakar

Klasifikasi system pakar berdasarkan kegunaannya menurut (Siswanto 2005)yaitu :

**a. Diagnosis:**

1. Digunakan untuk merekomendasikan: Obat untuk orang sakit, kerusakan mesin, kerusakan rangkaian elektronik.
2. Menemukan masalah/kerusakan yang terjadi.
3. Menggunakan pohon keputusan (*decision tree*) sebagai representasi pengetahuan

**b. Pengajaran:**

1. Digunakan untuk pengajaran, mulai dari SD sampai dengan PT.
2. Membuat diagnose apa penyebab kekurangan dari siswa, kemudian memberikan cara untuk memperbaikinya.

**c. Interpretasi:**

Untuk menganalisa data yang tidak lengkap, tidak teratur, dan data yang kontradiktif. Missal: untuk interpretasi citra.

**d. Prediksi:**

1. Contoh: bagaimana seorang pakar meteorology memprediksikan cuaca besok berdasarkan data-data sebelumnya.
2. Untuk peramalan cuaca.
3. Penentuan masa tanam.

**e. Perencanaan:**

1. Mulai dari perencanaan mesin-mesin sampai dengan manajemen bisnis.
2. Untuk menghemat biaya, waktu dan material, sebab pembuatan model.
3. Sudah tidak diperlukan.
4. Contoh: Sistem konfigurasi computer.

**f. Control:**

1. Digunakan untuk mengontrol kegiatan yang membutuhkan presisi waktu tinggi.
2. Missal: Pengontrolan pada industri-industri berteknologi tinggi.

## 2.2. Algoritma C4.5

Algoritma C4.5 merupakan algoritma yang digunakan untuk membentuk pohon keputusan. Didalam algoritma C4.5 ini, pohon-pohon keputusan yang dibentuk berdasarkan kriteria-kriteria pembentuk keputusan.

Pohon keputusan merupakan metode klasifikasi dan prediksi yang sangat kuat dan terkenal. Metode pohon keputusan mengubah fakta yang sangat besar menjadi pohon keputusan yang mempresentasikan aturan. Aturan dapat dengan mudah dipahami dengan Bahasa alami. Dan mereka juga dapat diekspresikan dalam bentuk Bahasa basis data seperti *Structured query language* untuk mencari record pada kategori tertentu.

Pohon keputusan juga berguna untuk mengeksplorasi data, menemukan hubungan yang tersembunyi antara sejumlah calon variable input dengan sebuah variable target. Karena pohon keputusan memadukan antara eksplorasi data dan pemodelan, dia sangat bagus sebagai langkah awal dalam pemrosesan pemodelan bahkan ketika dijadikan sebagai model akhir dari beberapa teknik lainnya.

Sebuah pohon keputusan adalah sebuah struktur yang dapat digunakan untuk memembagi kumpulan data yang besar menjadi himpunan-himpunan *record* yang lebih kecil dengan menerapkan serangkaian aturan keputusan. Dengan masing-masing rangkaian pembagian, anggota himpunan hasil menjadi mirip satu dengan yang lain (Berry & Linoff, 2004).

### 2.2.1. Pohon Keputusan

Pohon keputusan merupakan metode klasifikasi dan prediksi yang sangat kuat dan terkenal. Metode pohon keputusan mengubah fakta yang sangat besar menjadi pohon keputusan yang merepresentasikan aturan. Aturan dapat dengan mudah dipahami dengan bahasa alami, Dan mereka juga dapat diekspresikan dalam bentuk bahasa basis data seperti *Structured Query Language* untuk mencari *record* pada kategori tertentu.

Pohon keputusan juga berguna untuk mengeksplorasi data, menemukan hubungan tersembunyi antara sejumlah calon variabel input dengan sebuah variabel target. Karena pohon keputusan memadukan antara eksplorasi data dan pemodelan, dia sangat bagus sebagai langkah awal dalam proses pemodelan bahkan ketika dijadikan sebagai model akhir dari beberapa teknik lain.

Sebuah pohon keputusan adalah sebuah struktur yang dapat digunakan untuk membagi kumpulan data yang besar menjadi himpunan-himpunan record yang lebih kecil dengan menerapkan serangkaian aturan keputusan. Dengan masing-masing rangkaian pembagian, anggota himpunan hasil menjadi satu dengan yang lain (Bern' & Linoff, 2004).

Sebuah model pohon keputusan terdiri dari sekumpulan aturan untuk membagi sejumlah populasi yang heterogen menjadi lebih lebih homogen dengan memperhatikan pada variabel tujuannya. Sebuah pohon keputusan mungkin dibangun dengan saksama secara manual atau dapat tumbuh secara otomatis dengan menerapkan salah satu atau beberapa

algoritma pohon keputusan untuk memodelkan himpunan data yang belum terklasifikasi.

Variabel tujuan biasanya dikelompokkan dengan pasti dan model pohon keputusan lebih mengarah pada perhitungan probabilitas dari tiap-tiap record terhadap kategori-kategori tersebut atau untuk mengklasifikasi record dengan mengelompokkannya dalam satu kelas, Pohon keputusan juga dapat digunakan untuk mengestimasi nilai dari variabel continue meskipun ada beberapa teknik yang lebih sesuai untuk kasus ini. Banyak algoritma yang dapat dipakai dalam pembentukan pohon keputusan, antara Iain 11)3, CART, dan C4S (Larose, 2005). Algoritma C4.5 merupakan pengembangan dari algoritma ID3 (Larose, 2005),

Data dalam pohon keputusan biasanya dinyatakan dalam bentuk tabel dengan atribut dan record. Atribut menyatakan suatu parameter yang dibuat sebagai kriteria dalam pembentukan pohon. Misalkan untuk menentukan main tenis, kriteria yang diperhatikan adalah cuaca, angin, dan temperatur. Salah satu atribut merupakan atribut yang menyatakan data solusi per item data yang disebut target atribut. Atribut memiliki nilai-nilai yang dinamakan dengan *instance*. Misalkan atribut cuaca mempunyai instance berupa cerah, berawan, dan hujan (Basuki & Syarif, 2003).

Proses pada pohon keputusan adalah mengubah bentuk data (tabel) menjadi model pohon, mengubah model pohon menjadi rule, dan menyederhanakan rule (Basuki & Syarif, 2003).

### 2.2.2. Algoritma

Untuk memudahkan penjelasan mengenai algoritma C4.5, berikut ini disertakan contoh kasus yang dituangkan dalam Tabel 2.1.

**Tabel 2.1** Keputusan Bermain Tenis

| NO | OUTLOOK | TEMPERATURE | HUMIDITY | WINDY | PLAY |
|---|---|---|---|---|---|
| 1 | Sunny | Hot | High | FALSE | No |
| 2 | Sunny | Hot | High | TRUE | No |
| 3 | Cloudy | Hot | High | FALSE | Yes |
| 4 | Rainy | Mild | High | FALSE | Yes |
| 5 | Rainy | Cool | Normal | FALSE | Yes |
| 6 | Rainy | Cool | Normal | TRUE | Yes |
| 7 | Cloudy | Cool | Normal | TRUE | Yes |
| 8 | Sunny | Mild | High | FALSE | No |
| 9 | Sunny | Cool | Normal | FALSE | Yes |
| 10 | Rainy | Mild | Normal | FALSE | Yes |
| 11 | Sunny | Mild | Normal | TRUE | Yes |
| 12 | Cloudy | Mild | High | TRUE | Yes |
| 13 | Cloudy | Hot | Normal | FALSE | Yes |
| 14 | Rainy | Mild | Hihg | TRUE | No |

Dalam kasus yang tertera pada Tabel 2.1 akan dibuat pohon keputusan untuk menentukan main tenis atau tidak dengan melihat keadaan cuaca, temperatur, kelembapan, dan keadaan angin.

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

a. Pilih atribut sebagai akar.

b. Buat cabang untuk tiap-tiap nilai.

c. Bagi kasus dalam cabang.

d. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama.

Untuk memilih atribut sebagai akar didasarkan pada nilai gain tertinggi dari atribut-atribut yang ada. Untuk menghitung gain digunakan rumus seperti tertera dalam persamaan 2.1 berikut.

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

Keterangan:

S : Himpunan Kasus

A : Atribut

n : jumlah partisi atribut A

|Si| : jumlah kasus pada partisi ke-i

|S| : Jumlah Kasus dalam S

Sementara itu, perhitungan nilai entropi dapat dilihat pada persaman 2.2 berikut.

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

Keterangan:

S : Himpunan Kasus

A : Fitur

n : jumlah partisi S

pi : proporsi dari $S_i$ terhadap S

Berikut ini adalah penjelasan lebih terperinci mengenai tiap-tiap langkah dalam pembentukan pohon keputusan dengan menggunakan algoritma C4.5 untuk menyelesaikan permasalahan pada Tabel 2.1.

a. Menghitung jumlah kasus, jumlah kasus untuk keputusan Yes, jumlah kasus untuk keputusan No, dan Entropy dari semua kasus dan kasus yang dibagi berdasarkan atribut OUTLOOK, TEMPERATURE, HUMIDITY, dan WINDY. Setelah itu, lakukan penghitungan Gain untuk setiap atribut. Hasil perhitungan ditunjukkan oleh Tabel 2.2.

**Table 2.2** Perhitungan Node 1

| Node 1 | | | Jml Kasus (S) | Tidak ($S_1$) | Ya ($S_2$) | Entropy | Gain |
|---|---|---|---|---|---|---|---|
| | TOTAL | | 14 | 4 | 10 | 0.863 | |
| | OUTLOOK | | | | | | 0.258 |
| | | CLOUDY | 4 | 0 | 4 | | |
| | | RAINY | 5 | 1 | 4 | 0.722 | |
| | | SUNNY | 5 | 3 | 2 | 0.971 | |
| | TEMPERATURE | | | | | | 0.184 |
| | | COOL | 4 | 0 | 4 | 0 | |
| | | HOT | 4 | 2 | 2 | 1 | |
| | | MILD | 6 | 2 | 4 | 0.918 | |
| | HUMIDITY | | | | | | 0.370 |
| | | HIGH | 7 | 4 | 3 | 0.985 | |
| | | NORMAL | | 0 | 7 | 0 | |
| | WINDY | | | | | | 0.006 |
| | | FALSE | 8 | 2 | 6 | 0.811 | |
| | | TRUE | 6 | 4 | 2 | 0.918 | |

Baris TOTAL kolom Entropy pada Tabel 2.2 dihitung dengan persamaan 2 sebagai berikut:

$$Entropy(Total) = \left(-\frac{4}{14} * log_2\left(\frac{4}{14}\right)\right) + \left(-\frac{10}{14} * log_2\left(\frac{10}{14}\right)\right)$$

*Entropy(Total)* =0.863

Sementara itu, nilai Gain pada baris *OUTLOOK* dihitung dengan menggunakan persamaan 1 sebagi berikut.

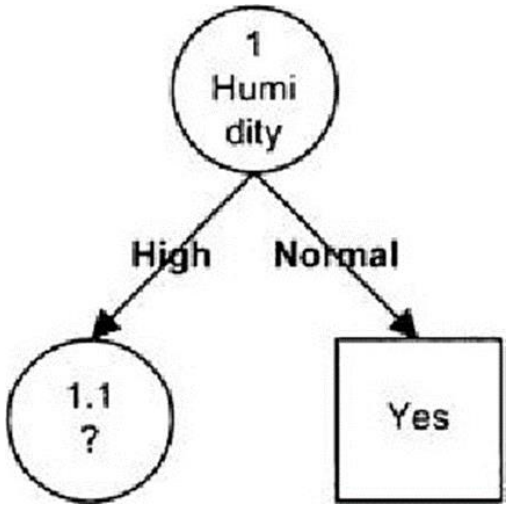
$$Gain(Total, Outlook)$$

$$= Entropy(Total) - \sum_{i=1}^{n} \frac{|\ Outlook\ |}{|\ Total\ |} * Entropy(Outlook_i)$$

$$Gain(Total, Outlook) = 0.863 - \left(\left(\frac{4}{14} * 0\right) + \left(\frac{5}{14} * 0.723\right) + \left(\frac{5}{14} * 0.97\right)\right)$$

*Gain(Total,Outlook)* = 0.23

Dari hasil pada Tabel 2.2 dapat diketahui bahwa atribut dengan Gain tertinggi adalah HUMIDITY, yaitu sebesar 0.37, Dengan demikian, HUMIDITY dapat menjadi node akar, Ada dua nilai atribut dari HUMIDITY, yaitu HIGH dan NORMAL. Dari kedua nilai atribut tersebut, nilai atribut NORMAL sudah mengklasifikasikan kasus menjadi 1, yaitu keputusannya Yes, sehingga tidak perlu dilakukan perhitungan lebih lanjut, tetapi untuk nilai atribut HIGH masih perlu dilakukan perhitungan lagi. Dari hasil tersebut dapat digambarkan pohon keputusan sementaranya tampak seperti Gambar 2.1.



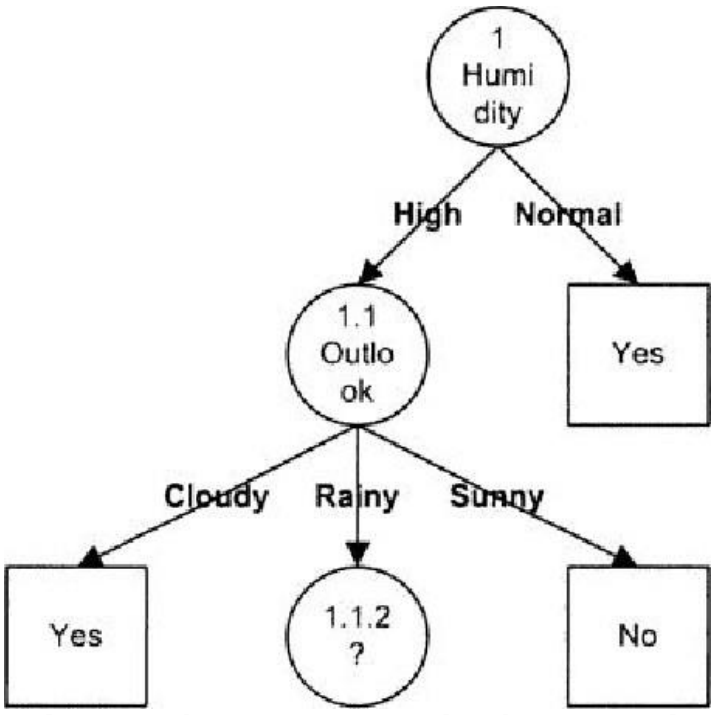

**Gambar 2.1** Pohon Keputusan Hasil Perhitungan Node 1

b. Menghitung jumlah kasus, jumlah kasus untuk keputusan ***Yes***, jumlah kasus untuk keputusan ***No***, dan Entropy dari semua kasus dan kasus yang dibagi berdasarkan atribut OUTLOOK, TEMPERATURE, dan WINDY yang dapat menjadi node akar dari nilai atribut HIGH. Setelah itu, lakukan penghitungan Gain untuk tiap-tiap atribut. Hasil perhitungan ditunjukkan oleh Tabel 2.3.

**Table 2.3** Perhitungan Node 1.1

| Node 1.1 | | | Jrnl Kasus (S) | Tidak (S1) | Ya (S2) | Entropy | Gain |
|---|---|---|---|---|---|---|---|
| | HUMIDITY-HIGH | | 7 | 4 | 3 | 0.985 | |
| | OUTLOOK | | | | | | 0.699 |
| | | CLOUDY | 2 | 0 | 2 | 0 | |
| | | RAINY | 2 | 1 | 1 | 1 | |
| | | SUNNY | 3 | 3 | 0 | 0 | |
| | TEMPERATURE | | | | | | 0.020 |
| | | COOL | 0 | 0 | 0 | 0 | |
| | | HOT | 3 | 2 | 1 | 0.918 | |
| | | MILD | 4 | 2 | 2 | 1 | |
| | WINDY | | | | | | 0.020 |
| | | FALSE | 4 | 2 | 2 | 1 | |
| | | TRUE | 3 | 2 | 1 | 0.918 | |

Dari hasil pada Tabel 2.3 dapat diketahui bahwa atribut dengan Gain tertinggi adalah OUTLOOK, yaitu sebesar 0.67. Dengan demikian OUTLOOK dapat menjadi node cabang dari nilai atribut HIGH. Ada tiga nilai atribut dari OUTLOOK. yaitu CLOUDY, RAINY, dan SUNNY. Dari ketiga nilai atribut tersebut, nilai atribut CLOUDY sudah mengklasifikasikan kasus menjadi l, yaitu keputusannya Yes dan nilai atribut SUNNY sudah mengklasifikasikan kasus menjadi satu dengan keputusan No, sehingga tidak perlu dilakukan perhitungan lebih lanjut, tetapi untuk nilai atribut RAINY masih perlu dilakukan perhitungan lagi. Pohon keputusan yang terbentuk sampai tahap ini ditunjukkan pada Gambar 2.2 berikut.



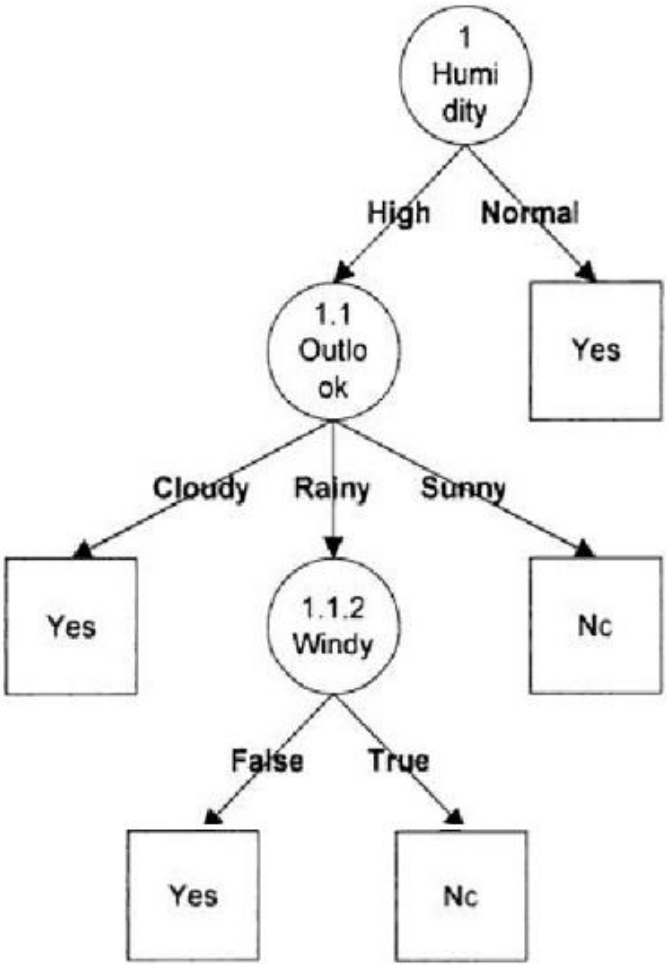

**Gambar 2.2** Pohon Keputusan Hasil Perhitungan Node 1.1

c. Menghitung jumlah kasus, jumlah kasus untuk keputusan ***Yes***, jumlah kasus untuk keputusan ***No***, dan Entropy dari semua kasus dan kasus yang dibagi berdasarkan atribut TEMPERATURE dan WINDY yang dapat menjadi node cabang dari nilai atribut RAINY. Setelah itu, lakukan penghitungan Gain untuk

tiap-tiap atribut, Hasil perhitungan ditunjukkan oleh Tabel 2.4. Dari hasil pada Tabel 2.4 dapat diketahui bahwa atribut dengan Gain tertinggi adalah WINDY, yaitu sebesar l. Dengan demikian, WINDY dapat menjadi node cabang dari nilai atribut RAINY. Ada dua nilai atribut dari WINDY, yaitu FALSE dan TRUE. Dari kedua nilai atribut tersebut, nilai atribut FALSE sudah mengklasifikasikan kasus menjadi 1, yaitu keputusannya ***Yes*** dan nilai atribut TRUE sudah mengklasifikasikan kasus menjadi satu dengan keputusan ***No***, sehingga tidak perlu dilakukan perhitungan lebih lamut untuk nilai atribut ini.

**Table 2.4** Perhitungan Node 1.1.2

| Node 1.1.2 | | | Jml Kasus (S) | Tidak (S1) | Ya (S2) | Entropy | Gain |
|---|---|---|---|---|---|---|---|
| | HUMIDITY-HIGH dan OUTLOOK-RAINY | | 2 | 1 | 1 | 1 | |
| | TEMPERATURE | | | | | | 0 |
| | | COOL | 0 | 0 | 0 | 0 | |
| | | HOT | 0 | 0 | 0 | 0 | |
| | | MILD | 2 | 1 | 1 | 1 | |
| | WINDY | | | | | | 1 |
| | | FALSE | 1 | 0 | 1 | 0 | |
| | | TRUE | 1 | 1 | 0 | 0 | |




**Gambar 2.3** Pohon Keputusan Hasil Perhitungan Node 1.1.2

Pohon keputusan yang terbentuk sampai tahap ini ditunjukkan pada Gambar 2.3. Dengan memperhatikan pohon keputusan pada Gambar 2.3, diketahui bahwa semua kasus sudah masuk dalam kelas. Dengan demikian, pohon keputusan pada Gambar 2.3 merupakan pohon keputusan terakhir yang terbentuk.

## 2.3. *Data base*

*Data base* adalah kumpulan terintegrasi dari elemen data yang secara logika saling berhubungan. *Database* mengonsolodasikan berbagai catatan dahulu disimpan dalam *file-file* terpisah kedalam satu gabungan umum elemen data yang menyediakan data untuk banyak aplikasi. Data yang disimpan dalam *data base* ketidak ketergantungan (independen) dari program aplikasi yang menggunakannya dan dari jenis peralatan penyimpanan tempat mereka disimpan.

## 2.4. *MySql*

*MySQL* adalah *Relational Database Management System* (RDBMS), yaitu *data base* relasi yang memiliki perintah standar adalah SQL (*Structured Query Language*). *MySQL* termasuk *Data base Server*, karna mendukung parintah SQL secara penuh dan dapat diakses dalam jaringan (bisa sebagai *Server* dan *Client*) (Nugroho 2006, Hal: 1).

MySQL adalah suatu database server merupakan  open source SQL database (Sunyoto 2007, Hal: 145). My SQL merupakan database server dimana pemrosesan data terjadi di server dan client hanya mengirim data dan memindah data. Pengaksesan dapat dilakukan dimana saja dan oleh siapa saja dengan catatan komputer telah terhubung ke server. Lain halnya dengan database dekstop dimana segala pemrosesan data seperti penambahan data ataupun penghapusan data harus dilakukan pada komputer yang bersangkutan. MySQL termasuk database yang terstruktur dalam pengolahan dan penampilan data. MySQL merupakan *Relational*

*Database Management System* (RDBMS) yaitu hubungan antar tabel yang berisi data-data pada suatu database (Kadir 2002, Hal: 353). Tabel-tabel tersebut di-link oleh suatu relasi yang memungkinkan untuk mengkombinasikan data dari beberapa tabel ketika seorang user menginginkan menampilkan informasi dari suatu database. Database MySQL merupakan sistem manajemen basis data SQL yang sangat terkenal dan bersifat Open Source. MySQL dibangun, didistribusikan dan didukung oleh MySQL AB. MySQL AB merupakan perusahaan komersial yang dibiayai oleh pengembang (*developer*) MySQL.

MySQL dapat didefenisikan sebagai :

1. MySQL merupakan sistem manajemen database. Database merupakan struktur penyimpanan data. Untuk menambah, mengakses, dan memproses data yang disimpan dalam sebuah database komputer, diperlukan sistem manajemen database seperti MySQL Server.
2. MySQL merupakan sistem manajemen database atau basis data terhubung (*relational database manajemen system*). Databaseterhubung menyimpan data pada tabel-tabel terpisah. Hal tersebut akan menambah kecepatan dan fleksibilitasnya. Kata SQL pada MySQL merupakan singkatan dari "*Structured Query Languange*". SQL merupakan bahasa standar yang digunakan untuk mengakses database dan ditetapkan oleh ANSI/ISO SQL standar.
3. MySQL merupakan software *open source*. *Open source* berarti semua orang diizinkan menggunakan dan memodifikasi software. Semua orang dapat men-download software MySQL dari internet dan

menggunakannya tanpa membayar. Anda dapat mempelajari *source Code* dan menggunakannya sesuai kebutuhan.

4. Server database MySQL mempunyai kecepatan akses tinggi, mudah digunakan, dan andal. MySQL dikembangkan untuk menangani database yang besar secara cepat dan telah sukses digunakan selama bertahun-tahun. Konektivitas, kecepatan, dan keamanannya membuat server MySQL cocok untuk mengakses database di internet.
5. MySQL server bekerja di klien/server atau sistem *embedded*. Software database MySQL merupakan sistem klien/server yang terdiri atas *multithread* SQL server yang mendukung software klien dan library yang berbeda, *tool administratif*, dan sejumlah *Application Programming Interfaces* (APIs).
6. MySQL tersedia dalam beberapa macam bahasa. (wahana komputer 2006, Hal: 182).

## 2.5. PHP

### 2.5.1. Pengertian PHP

Menurut Anhar (2010) Pengertian PHP yaitu *Hypertext Preprocessor* adalah pemrograman web *server-side* yang bersifat open source. PHP merupakan script yang terintegrasi dengan HTML dan berada pada server (*Server side HTML embedded scripting*). PHP adalah *script* yang digunakan untuk membuat halaman website dinamis. Dinamis berarti halam yang akan ditampilkan dibuat saat halam itu diminta oleh *client.* Mekanisme ini menyebabkan informasi yang diterima *client* selalu yang

terbaru. Semua *script* PHP dieksekusi pada *server* dimana tersebut dijalankan (Anhar 2010).

PHP adalah salah satu bahasa pemrograman yang berjalan pada sebuah web server dan berfungsi sebagai pengolah data pada sebuah server. Sintak PHP mirip dengan bahasa Perl dan C. PHP biasanya sering digunakan bersama web server Appache di beragam sistem operasi. PHP juga men-support ISAPI dan dapat digunakan bersamadengan Microsoft IIS di Windows (Sunyoto 2007). Secara khusus PHP dirancang untuk web dinamis. Artinya PHP dapat membentuk suatu tampilan berdasarkan permintaan terkini. Misalnya dapat menampilkan isi database ke halaman web. Pada prinsipnya PHP memiliki fungsi yang sama dengan skrip-skrip seperti ASP (*Active Server Page*), Cold Fusion ataupun Perl.

### 2.5.2. Kelebihan PHP

PHP memiliki beberapa kelebihan, antara lain:

1. Mudah dibuat dan dijalankan
2. Mampu berjalan pada web server dengan sistem operasi yang berbeda-beda: PHP mampu berjalan dengan sistem operasi UNIX, keluarga windows dan machintos
3. PHP bisa didapatkan secara gratis
4. Dapat berjalan pada web server yang berbeda: PHP mampu berjalan pada web server yang berbeda-beda, seperti Microsoft personal Web Server, Apache, IIS, Xitami
5. Dapat di-*embeded*: PHP dapat diletakan dalam tag HTML.

## 2.6. Pengertian WAMPServer

WAMP adalah singkatan dari *Windows and the principal components of the package*:Apache, MySQL and PHP ( Perl or Python). Apache adalah *Web server*, MySQL adalah database, sedangkan PHP adalah bahasa scripting yang dapat memanipulasi informasi yang dibuat di database dan menghasilkan halaman web dinamis konten setiap waktu diminta oleh browser.

Wamp merupakan sebuah aplikasi yang dapat menjadikan komputer kita menjadi sebuah server. Kegunaan **Wamp*server*** adalah untuk membuat jaringan local sendiri dalam arti kita dapat membuat website secara offline untuk masa coba-coba di komputer sendiri. Jadi fungsi dari wamp server itu sendiri merupakan server website kita untuk cara memakainya. Mengapa harus menjadi server? Karena dalam hal ini komputer yang akan kita pakai harus memberikan pelayanan untuk pengaksesan web, untuk itu komputer kita harus menjadi server.

Biasanya para perancang web atau web master jika akan merencanakan (planing), kemudian membangun (buliding) dilakukan di komputer local atau bisa juga di jaringan local, tidak langsung di host –> internet. Oleh karena itu perlu dikomputer kita di jadikan server sehingga kita seolah olah sedang meng update di hostnya ( tempat penyimpanan file-file yang diperlukan website ) –> internet. Dengan di tempatkannya file pendukung website di komputer, kita tidak perlu on line via internet, sehingga hal ini mengurangi presentasi waktu dan biaya (Sibero 2013).