Nama : Muh.Ikhsan
NIM : H071191049

Classification (Klasifikasi)

A. Definisi

Klasifikasi adalah proses penemuan model (fungsi) yang menggambarkan dan membedakan kelas data atau konsep yang bertujuan agar bisa digunakan untuk memprediksi kelas dari objek yang label kelasnya tidak diketahui.

Proses klasifikasi data terdiri dari 2 tahap. Pertama adalah learning (fase training), dimana algoritma klasifikasi dibuat untuk menganalisa data training lalu direpresentasikan dalam bentuk rules. Proses kedua adalah tahap klasifikasi, dimana data testing digunakan untuk memperkirakan akurasi dari rules yang dihasilkan pada tahap learning

B. Penerapan Klasifikasi

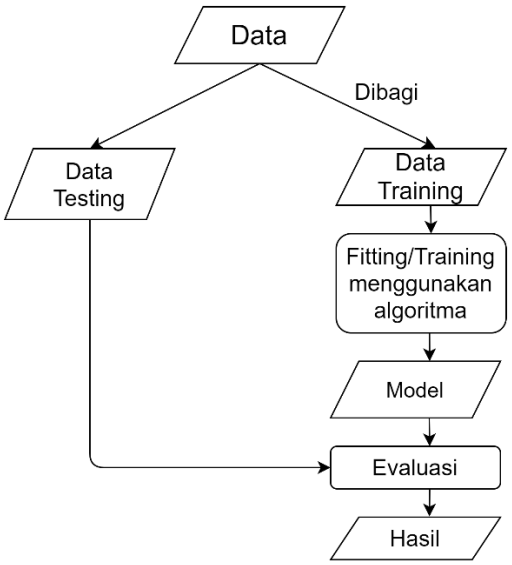
1. Persetujuan Kredit/Pinjama
2. Diagnosis Medis : Kanker, Covid-19
3. Deteksi Penipuan
4. Email Spam/Tidak
5. Dll

C. Komponen Klasifikasi

1. Kelas : variabel dependen yang berupa kategorikal yang merepresentasikan “label” yang terdapat pada objek. Contohnya: resiko penyakit jantung, resiko kredit, customer loyalty, jenis gempa.
2. Predictor : variabel independen yang direpresentasikan oleh karakteristik (atribut) data. Contohnya: merokok, minum alkohol, tekanan darah, tabungan, aset, gaji.
3. Training dataset : suatu dataset yang berisi nilai dari kedua komponen di atas yang digunakan untuk menentukan kelas yang cocok berdasarkan predictor.

4. Testing dataset : dataset yang berisi data baru yang akan diklasifikasikan oleh model yang telah dibuat dan akurasi klasifikasi dievaluasi.

D. Proses Klasifikasi



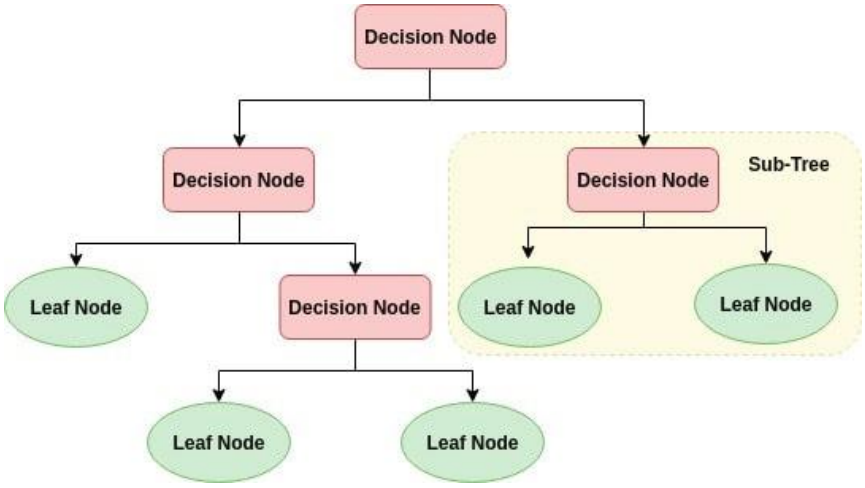
1. Data dibagi 2 Jenis : Data Training dan Data Testing (Proporsi berdasarkan kondisi data. Umumnya 80:20 atau 70:30)
2. Data Training digunakan untuk membangun model
3. Data Training dilakukan fitting/training menggunakan algoritma tertentu. Proses fitting akan menghasilkan model.
4. Data Testing digunakan untuk mengevaluasi model yang dihasilkan

E. Algoritma Klasifikasi yang populer

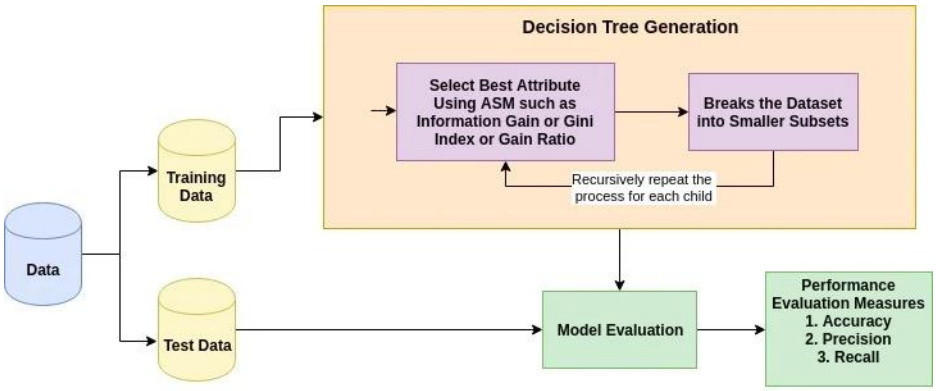
1. Decision/ pohon klasifikasi
2. Bayesian classifiers/Naive Bayes classifiers
3. Neural networks
4. Statistical analysis
5. Rough sets
6. k-nearest neighbor classifier
7. Rule-based methods
8. Memory based reasoning
9. Support vector machines.

F. Metode : Decision Tree

Decision tree atau Pohon Keputusan merupakan metode non parametrik yang digunakan untuk klasifikasi dan regresi. Proses pada Decision Tree adalah mengubah bentuk data (tabel) menjadi bentuk Tree, Mengubah Tree menjadi Rule, dan menyederhanakan Rule (basuki&syarif,2003) Decision Tree memiliki struktur pohon seperti *flow-chart* dimana simpul internal mewakili attribut, cabang mewakili aturan keputusan (decision rule), dan setiap simpul daun mewakili hasilnya. Node paling atas dikenal sebagai root node. Tujuan dari decision tree adalah membuat model yang memprediksi nilai variabel target dengan mengikuti aturan keputusan sederhana dari fitur data yang tersedia



- **Cara Kerja Algoritma Decision Tree :**
  1. Memilih atribut terbaik menggunakan Attribution Selection Measures (ASM) untuk membagi records data
  2. Membuat atribut menjadi decision node dan memecah dataset menjadi subsets yang lebih kecil
  3. Mulai membangun pohon dengan mengulangi proses ini secara rekursif, untuk setiap cabang sampai salah satu dari kondisi tersebut akan cocok.



- **Pseudocode Decision Tree**

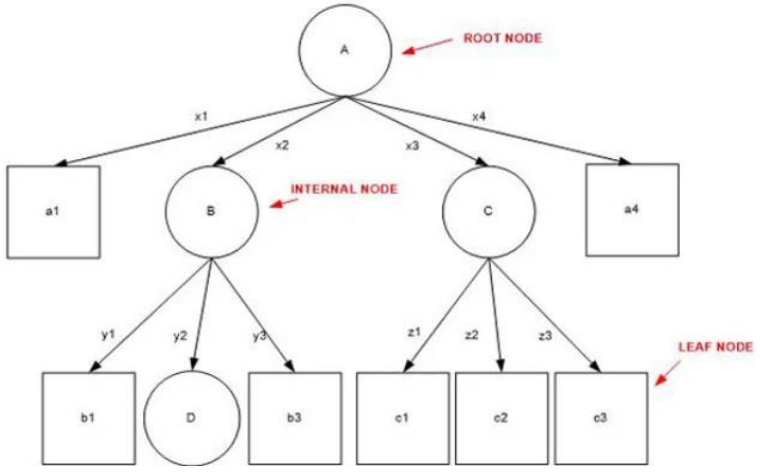
```
TreeGrowth (E,F)
1    if stopping_cond(E,F) = true then
2        leaf = createNode ().
3        leaf.label = Classify (E).
4        return leaf
5    Else
6        root = createNode()
7        root.test_cond = find_best_split(E,F)
8        let V = {v | v is a possible outcome of root.test_cond}
9        for each v ϵ V do
10           E_v = {e | root.test_cond(e) = v and e ϵ E}.
11           child = TreeGrowth (E_v , F).
12           and child as desendent of root and label the aged (root
     → child) as v.
13       end foor
14   and if
15   return root.
```

- **Arsitektur Decision Tree:**

  Arsitektur pohon keputusan dibuat menyerupai bentuk pohon, dimana pada umumnya sebuah pohon terdapat akar (root), cabang dan daun (leaf). Pada pohon keputusan juga terdiri dari tiga bagian sebagai berikut :

  1. Root node atau node akar merupakan node yang terletak paling atas dari suatu pohon.
  2. Internal Node ini merupakan node percabangan, dimana pada node ini hanya terdapat satu input dan mempunyai minimal dua output.
  3. Leaf Node ini merupakan node akhir, hanya memiliki satu input, dan tidak memiliki output. Pada pohon keputusan setiap leaf node menandai label kelas.

  Pada pohon keputusan di setiap percabangan menyatakan kondisi yang harus dipenuhi dan tiap ujung pohon menyatakan nilai kelas data. Gambar berikut merupakan bentuk arsitektur pohon keputusan.



- **Attribut Selection Measure (Pemilihan Atribut)**
  Attribut selection measure adalah sebuah heuristik dalam pemilihan kriteria spliyyin bagian-bagina data mejadi aturan yang sesuai. Hal ini juga menjadikan acuan spiltitin karena membantu kita untuk mengetahui breakpoint pada tuples yang diberikan oleh node. ASM memberikan tingkatan pada setiap fitur dengan menjelaskan dataset yang diberikan. Nilai terbaik dalam fitur akan dipilih sebagai fitur splitting.
  - Tujuan pemilihan atribut adalah untuk mendapatkan decision tree yang paling kecil ukurannya.
  - Pure (bersih) adalah apabila dalam satu cabang anggotanya berasal dari satu kelas . Semakin pure semakin cabang maka akan semakin baik.
  - *Impurty* adalah ukuran purity suatu cabang.
  - Contoh *Impurty* adalah Information Gain, Rasio Gain, Indeks Gain, dll
- **Entropy**
  Entropi adalah nilai informasi yang menyatakan ukuran ketidakpastian (impurity) dari attribut dari suatu kumpulan obyek data sample dalam satuan bit. Untuk mendapatkan nilai Information Gain dan Gain Ratio, terlebih dahulu kita harus menghitung nilai entropy.

$$Entropy(S) = -\sum_{i=1}^{k} P_i \log_2 P_i \ , \qquad p_i = \frac{|S_i|}{|S|}$$

  Dimana,
  S : Himpunan Kasus
  k : Jumlah partisi S
  Pi: Proporsi dari S_i terhadap S

- **Information Gain**
  Information gain menghitung perbedaan antara entropy sebelum pemisahan dan rata-rata entropy setelah pemisahan kumpulan data berdasarkan nilai atribut yang diberikan. Algoritma decision tree ID3 menggunakan information gain.

$$Info(S) = Entropy(S)$$

$$Info_A(S) = \sum_{i=1}^{k} \frac{|S_j|}{|S|} \ Info(S_j)$$

$$Gain(A) = Info(S) - Info_A(S)$$

- $Info(S)$ adalah jumlah rata-rata informasi yang dibutuhkan untuk mengidentifikasi kelas dari sebuah tuple di D.
- $\frac{|S_i|}{|S|}$ bertindak sebagai bobot partisi ke-i
- $Info_A(S)$ adalah informasi yang diharapkan yang diperlukan untuk menghasilkan tuple dari S berdasarkan partisi oleh A.

Attribut A dengan perolehan informasi tertinggi, Gain(A) dipilih sebagi atribut splitting pada node N()

- **Gain Ratio**

  Gain ratio menangani masalah normalisasi atau bias (Distorsi yang mana informasi yang didapat tidak representative terhadap situasi yang sebenarnya) dengan menormalkan informasi menggunakan Split Information. Algoritma C4.5 yang merupakan peningkatan dari ID3 menggunakan Gain Ratio.

$$SplitInfo_A(S) = Entropy(S) = -\sum_{j=1}^{k} \frac{|S_j|}{|S|} \times \log_2\left(\frac{|S_j|}{|S|}\right)$$

  Dimana :
  - $|D_j|/|D|$ bertindak sebagai bobot partisi ke-j.
  - k adalah jumlah nilai diskrit dalam atribut A.

  Gain Ratio dapat didefinisikan sebagai

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(S)}$$

  Attribut dengan gain ratio tertinggi dipilih sebagai attribut splitting.

- **Gini Index**

  Algoritma decision tree lainya CART menggunakan metode Gini untuk membuat titik-titik *split.*

$$Gini(S) = 1 - \sum_{j=1}^{k} Pi^2$$

  Dimana, pi adalah probabilitas bahwa sebuah tupel di D termasuk dalam kelas Ci.

  Indeks Gini mempertimbangkan pemisahan biner untuk setiap atribut. Kita dapat menghitung besar dari *Impurty* setiap partisi. Jika biner split

pada atribut A mempartisi data S menjadi S1 dan S2, indeks Gini dari S adalah:

$$Gini_A(S) = \frac{|S_1|}{|S|} Gini(S_1) + \frac{|S_2|}{|S|} Gini(S_2)$$

Secara umum dapat dihitung dengan

$$Gini_A(S) = \sum_{j=1}^{k} \frac{|S_j|}{|S|} \times Gini(S_j)$$

Dalam hal atribut bernilai diskrit, subset yang memberikan indeks gini minimum untuk yang dipilih dipilih sebagai atribut pemisah. Dalam kasus atribut bernilai kontinu, strateginya adalah memilih setiap pasangan nilai yang berdekatan sebagai titik pisah yang memungkinkan dan titik dengan indeks gini yang lebih kecil dipilih sebagai titik pemisah.

$$\Delta Gini(A) = Gini(S) - Gini_A(S)$$

Atribut dengan Gini minimum dipilih sebagai atribut pemisah.

- **Kelebihan Decision Tree**
  1. Mudah dipahim dan diinterpretasikan
  2. Dapat dengan mudah menangkap pola Non-linear
  3. Membutuhkan lebih sedikit *pra-processing* data dari pengguna, misalnya tidak perlu menormalisasi kolom
  4. Dapat digunakan untuk rekayasa fitur seperti memprediksi nilai yang hilang, cocok untuk pemilihan variabel
  5. Pohon keputusan tidak memiliki asusmis tentang distribusi karena algoritma yang non parametrik.
- **Kekurangan Decision Tree**
  1. Sensitif terhadap data yang bising. Hal ini dapat overfit data yang bising.
  2. Variasi kecil (atau varians) dalam data dapat menghasilkan pohon keputusan yang berbeda. Ini dapat dikurangi dengan mengantongi dan meningkatkan algoritma.
  3. Pohon keputusan bias dengan dataset ketidakseimbangan, sehingga disarankan untuk menyeimbangkan dataset sebelum membuat pohon keputusan.

G. ID3 (Iterative Dichotomiser 3)

1. Pengeritan Algoritma ID3

Algoritma ini melakukan pencarian secara menyeluruh (greedy) pada semua kemungkinan pohon keputusan ID3 dikembangkan oleh J. Ross

Quinlan. Algoritma ID3 dapat diimplementasikan menggunakan fungsi rekursif (fungsi yang memanggil dirinya sendiri).

Decision tree menggunakan struktur hierarki untuk pembelajaran supervised. Proses dari decision tree dimulai dari root node hingga leaf node yang dilakukan secara rekursif. Di mana setiap percabangan menyatakan suatu kondisi yang harus dipenuhi dan pada setiap ujung pohon menyatakan kelas dari suatu data.

Proses dalam decision tree yaitu mengubah bentuk data (tabel) menjadi model pohon (tree) kemudian mengubah model pohon tersebut menjadi aturan (rule). Dengan pendekatan ini, salah satu kelemahan algoritma dari decision tree, adalah faktor skalabilitas dimana algoritma tersebut hanya dapat digunakan untuk menangani sampel-sampel yang dapat disimpan secara keseluruhan dan pada waktu yang bersamaan di memori.

Algoritma ID3 tidak pernah melakukan backtracking untuk merevisi keputusan pemilihan attribute yang telah dilakukan sebelumnya. ID3 hanya menangani nilai-nilai attribute yang sedikit dan diskret, tetapi algoritma modifikasinya, algoritma C4.5 (1993), selanjutnya mampu menangani nilai attribute kontinu.

2. **Langkah-langkah Konstruksi Decision Tree dengan Algoritma ID3**
   Adapun langkah-langkahnya sebagai berikut :
   Langkah 1 : Pohon dimulai dengan sebuah simpul yang mereperesentasikan sampel data pelatihan yaitu dengan membuat simpul akar.
   Langkah 2 : Jika semua sampel berada dalam kelas yang sama, maka simpul ini menjadi daun dan dilabeli menjadi kelas. Jika tidak, information gain akan digunakan untuk memilih atribut terbaik dalam memisahkan data sampel menjadi kelas-kelas individu.
   Langkah 3 : Cabang akan dibuat untuk setiap nilai pada atribut dan data sampel akan dipartisi lagi.
   Langkah 4 : Algoritma ini menggunakan proses rekursif untuk membentuk pohon keputusan pada setiap data partisi. Jika sebuah atribut sduah digunakan disebuah simpul, maka atribut ini tidak akan digunakan lagi di simpul anak-anaknya.
   Langkah 5 : Proses ini berhenti jika dicapai kondisi seperti berikut :
   - Semua sampel pada simpul berada di dalam satu kelas

- Tidak ada atribut lainnya yang dapat digunakan untuk mempartisi sampel lebih lanjut. Dalam hal ini akan diterapkan suara terbanyak. Ini berarti mengubah sebuah simpul menjadi daun dan melabelinya dnegan kelas pada suara terbanyak.

3. Latihan Soal

Data yang telah ada pada Tabel dibawah akan digunakan untuk membentuk pohon keputusan dimana Cmemiliku atribut-atribut seperti Cuaca, Suhu, Kelembaban, dan Berangin. Setiap atribut memiliki nilai. Sedangkan kelasnya ada pada kolom Main yaitu kelas "Tidak" dan kelas "Ya".

No	Cuaca	Suhu	Kelembaban	Berangin	Main
1	Cerah	Panas	Tinggi	Salah	Tidak
2	Cerah	Panas	Tinggi	Benar	Tidak
3	Berawan	Panas	Tinggi	Salah	Ya
4	Hujan	Sejuk	Tinggi	Salah	Ya
5	Hujan	Dingin	Normal	Salah	Ya
6	Hujan	Dingin	Normal	Benar	Ya
7	Berawan	Dingin	Normal	Benar	Ya
8	Cerah	Sejuk	Tinggi	Salah	Tidak
9	Cerah	Dingin	Normal	Salah	Ya
10	Hujan	Sejuk	Normal	Salah	Ya
11	Cerah	Sejuk	Normal	Benar	Ya
12	Berawan	Sejuk	Tinggi	Benar	Ya
13	Berawan	Panas	Normal	Salah	Ya
14	Hujan	Sejuk	Tinggi	Benar	Tidak

Langkah 1 : Menghitung nilai entropy dan nilai gain

Hitung nilai entripy dari total kasus :
S1 = 10 (Terdapat 10 "Ya")
S2 = 4 (Terdapat 4 "Tidak")
S = 14 (Total kasus)

$$Entropi(S) = \left(-\left(\frac{10}{14}\right)\right) \times \log_2\left(\frac{10}{14}\right) + \left(-\left(\frac{4}{14}\right) \times \log_2\left(\frac{4}{14}\right)\right)$$

$$= 0.863120569$$

Total Kasus	Sum(Ya)	Sum(Tidak)	Entropi Total
14	10	4	0.863120569

Hitung nilai entropy pada setiap atribut :

$$\text{Entropy}(\text{Cuaca} = \text{Berawan}) = \left(-\left(\frac{2}{2}\right)\right) \times \log_2\left(\frac{2}{2}\right) + \left(-\left(\frac{0}{2}\right) \times \log_2\left(\frac{0}{2}\right)\right)$$

… (Begitupula dengan atribut lainnya)

Node	Atribut	Nilai	Sum(N)	Sun(Ya)	Sum(Tidak)	Entropy
1	Cuaca	Berawan	4	4	0	0
		Hujan	5	4	1	0.721928095

		Cerah	5	2	3	0.970950594
	Suhu	Dingin	4	4	0	0
		Panas	4	2	2	1
		Sejuk	6	4	2	0.918295834
	Kelembaban	Tinggi	7	3	4	0.985228136
		Normal	7	7	0	0
	Berangin	Salah	8	6	2	0.811278124
		Benar	6	2	4	0.918295834

Setelah dicari nilai entropy pada setiap nilai dari atribut yang ada, kemudian lakukan perhitungan untuk mencari nilai gain dari setiap atribut.

$Gain\ (Cuaca)$

$$= 0.863120569 - ((4/14)\ x\ 0 + (5/14)\ x\ 0.721928095 + (5/14)\ x\ 0.970950594)$$

$$= 0.258521037$$

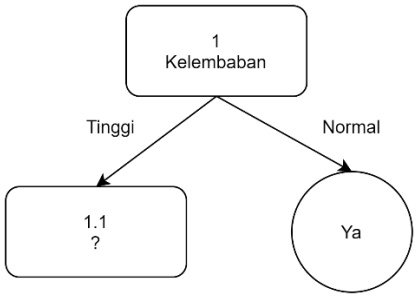
… Hitung pula Gain (Suhu), Gain (Kelembaban), dan Gain (Berangin).

Node	Atribut	Nilai	Sum(N)	Sun(Ya)	Sum(Tidak)	Entropy	Gain
1	Cuaca	Berawan	4	4	0	0	
		Hujan	5	4	1	0.721928095	
		Cerah	5	2	3	0.970950594	
							0.258521037
	Suhu	Dingin	4	4	0	0	
		Panas	4	2	2	1	
		Sejuk	6	4	2	0.918295834	
							0.183850925
	Kelembaban	Tinggi	7	3	4	0.985228136	
		Normal	7	7	0	0	
							0.370506501
	Berangin	Salah	8	6	2	0.811278124	
		Benar	6	2	4	0.918295834	
							0.005977711

Langkah 2 : Menentukan node akar

1. Karena nilai gain terbesar adalah Gain (Kelembaban), maka atribut "Kelembaban" menjadi node akar (root node).

Kemudian pada "Kelembaban" normal, memiliki 7 kasus dan semuanya memiliki jawaban Ya (Sum(Total) / Sum(Ya) = 7/7 = 1). Dengan demikian "Kelembaban" normal menjadi daun atau leaf.



Berdasarkan pembentukan pohon keputusan node 1 (root node), Node 1.1 akan dianalisis lebih lanjut. Untuk mempermudah, Tabel dibawah difilter, dengan mengambil data yang memiliki "Kelembaban" = Tinggi.

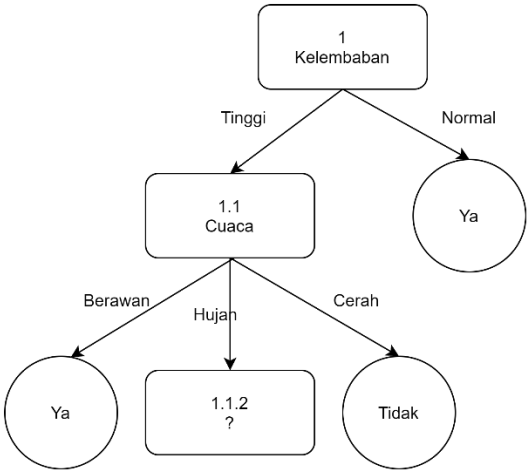
No	Cuaca	Suhu	Kelembaban	Berangin	Main
1	Cerah	Panas	Tinggi	Salah	Tidak
2	Cerah	Panas	Tinggi	Benar	Tidak
3	Berawan	Panas	Tinggi	Salah	Ya
4	Hujan	Sejuk	Tinggi	Salah	Ya
5	Cerah	Sejuk	Tinggi	Salah	Tidak
6	Berawan	Sejuk	Tinggi	Benar	Ya
7	Hujan	Sejuk	Tinggi	Benar	Tidak

Kemudian dihitung nilai entropi atribut "Kelembaban" Tinggi dan entropi setiap atribut serta gainnya. Setelah itu tentukan pilih atribut yang memiliki gain tertinggi untuk dibuatkan node berikutnya.

Kelembaban Tinggi	Sum(Ya)	Sum(Tidak)	Entropy
7	3	4	0.985228136

Node	Atribut	Nilai	Sum(N)	Sun(Ya)	Sum(Tidak)	Entropy	Gain
1	Cuaca	Berawan	2	2	0	0	
		Hujan	2	1	1	1	
		Cerah	3	0	3	0	
							0.69951385
	Suhu	Dingin	0	0	0	0	
		Panas	3	1	2	0.918295834	
		Sejuk	4	2	2	1	
							0.020244207
	Berangin	Salah	4	2	2	0	
		Benar	3	2	1	0.918295834	
							0.020244207

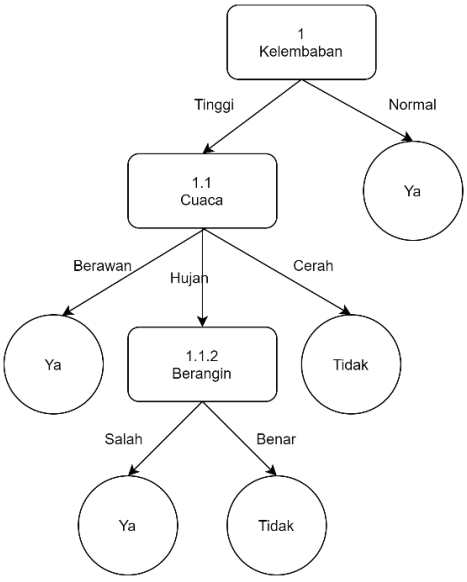
Gain tertinggi yang didapat ada pada atribut "Cuaca", dan Nilai yang dijadikan daun atau leaf adalah Berawan dan Cerah. Jika divualisasi maka pohon keputusan tampak seperti Gambar dibawah.
Untuk menganalisis node 1.1.2, lakukan lagi langkah-langkah yang sama seperti sebelumnya hingga semua node beberntuk node leaf.



No	Cuaca	Suhu	Kelembaban	Berangin	Main
1	Hujan	Sejuk	Tinggi	Salah	Ya
2	Hujan	Sejuk	Tinggi	Benar	Tidak

Kelembaban Tinggi & Hujan	Sum(Ya)	Sum(Tidak)	Entropy
2	1	1	1

Node	Atribut	Nilai	Sum(N)	Sun(Ya)	Sum(Tidak)	Entropy	Gain
	Suhu	Dingin	0	0	0	0	
		Panas	0	0	0	0	
		Sejuk	2	1	1	1	
							0
	Berangin	Salah	1	1	0	0	
		Benar	1	0	1	0	
							1



H. C4.5

Algoritma C4.5 merupakan pengembangan dari algoritma ID3. Algoritma C4.5 mempunyai prinsip dasar kerja yang sama dengan algoritma ID3. dimana pengembangan dilakukan dalam hal, bisa mengatasi missing data, bisa mengatasi data kontinu dan pruning.

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.

Latihan Soal :

Misalnya kita menggunakan data pada latihan ID3

Total Kasus	Sum(Ya)	Sum(Tidak)	Split Info
14	10	4	0.863120569

Node	Atribut	Nilai	Sum(N)	Sun(Ya)	Sum(Tidak)	Entropy	Gain
1	Cuaca	Berawan	4	4	0	0	
		Hujan	5	4	1	0.721928095	
		Cerah	5	2	3	0.970950594	
							0.258521037
	Suhu	Dingin	4	4	0	0	
		Panas	4	2	2	1	
		Sejuk	6	4	2	0.918295834	
							0.183850925
	Kelembaban	Tinggi	7	3	4	0.985228136	
		Normal	7	7	0	0	
							0.370506501
	Berangin	Salah	8	6	2	0.811278124	
		Benar	6	2	4	0.918295834	
							0.005977711

Langkah 1 : Menentukan Gain Ratio Berdasarkan Entropy dan Gain

Mencari Gain Ratio dari Gain dan Entropy pada latihan ID3

$$GainRatio(Cuaca) = \frac{0.258521037}{0.863120569} = 0.299519032$$

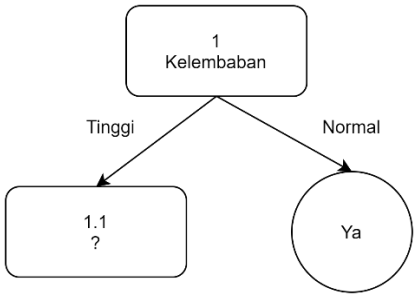
… (Begitu Pula Dengan Attribut Lainnya)

Node	Atribut	Gain	Gain Ratio
1	Cuaca	0.258521037	0.299519032
	Suhu	0.183850925	0.213007234
	Kelembaban	0.370506501	0.429263899
	Berangin	0.005977711	0.006925696

Langkah 2 :

Menentukan node akar

Karena nilai gain ratio terbesar adalah Gain Ratio (Kelembaban), maka atribut “Kelembaban” menjadi node akar (root node).

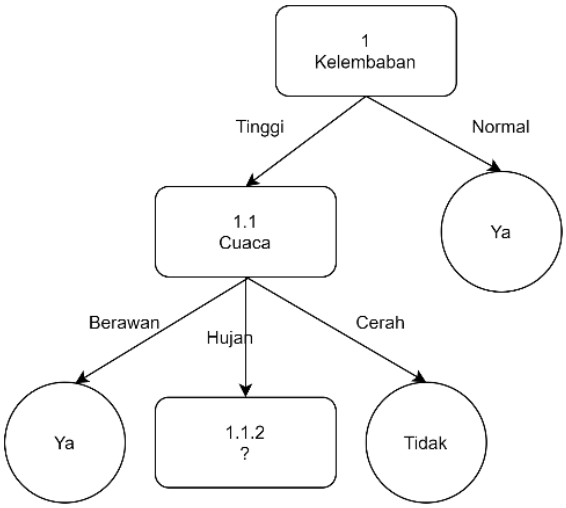


Karena rootnya sama dengan yang di ID3 maka untuk entropy dan gainnya juga akan sama

Kelembaban Tinggi	Sum(Ya)	Sum(Tidak)	Split Info
7	3	4	0.985228136

Atribut	Gain	Gain Ratio
Cuaca	0.699513850	0,710001902
Suhu	0.020244207	0,020547735
Berangin	0.020244207	0,020547735

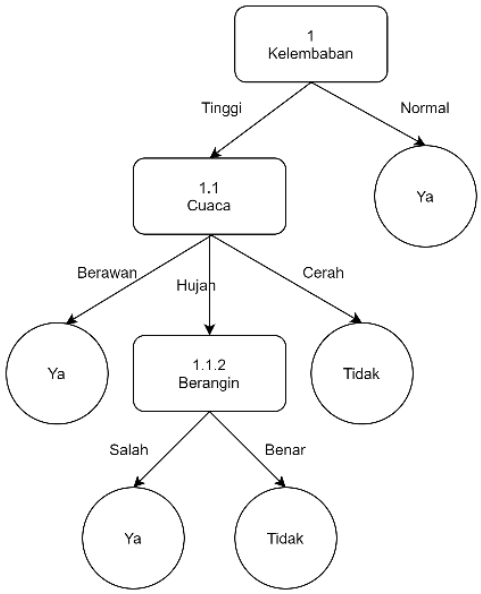
Gain Ratio tertinggi yang didapat ada pada atribut “Cuaca”, dan Nilai yang dijadikan daun atau leaf adalah Berawan dan Cerah.



Hasil ini juga sama pada ID3 jadi kita dapat langsung melihat tabelnya

Kelembaban Tinggi & Hujan	Sum(Ya)	Sum(Tidak)	Split Info
2	1	1	1

Atribut	Gain	Gain Ratio
Suhu	0	0
Berangin	1	1



I. C5.0

Algoritma C5.0 ini merupakan penyempurnaan algoritma sebelumnya yang dibentuk oleh Ross Quinlan pada tahun 1987, yaitu ID3 dan C4.5. Dalam algoritma ini pemilihan atribut diproses menggunakan gain ratio. Algoritma ini menghasilkan tree dengan jumlah cabang per node bervariasi.

C5.0 menghasilkan tree dengan jumlah cabang per node bervariasi. Perbedaan yang dapat dilihat secara fisik yaitu tree yang akan dihasilkan oleh Algoritma C5.0 akan lebih ringkas jika dibandingkan dengan Algoritma C4.5. Karena itulah pembangungan tree pada algoritma ini lebih cepat dibandingkan Algoritma C4.5.

J. Cart

Algoritma ini dapat digunakan untuk klasifikasi & regresi. Algoritma CART menggunakan kriteria Gini Index untuk membagi sebuah node menjadi sub-node. Ini dimulai dengan set pelatihan sebagai simpul akar, setelah berhasil membagi simpul akar menjadi dua, ia membagi subset menggunakan logika yang sama & lagi membagi sub-subset, secara rekursif hingga menemukan pemisahan lebih lanjut tidak akan memberikan sub-node murni atau jumlah maksimum daun dalam satu pohon yang sedang tumbuh atau disebut sebagai Tree pruning.

Latihan Soal

No	Cuaca	Suhu	Kelembaban	Berangin	Main
1	Cerah	Panas	Tinggi	Salah	Tidak
2	Cerah	Panas	Tinggi	Benar	Tidak
3	Berawan	Panas	Tinggi	Salah	Ya
4	Hujan	Sejuk	Tinggi	Salah	Ya
5	Hujan	Dingin	Normal	Salah	Ya
6	Hujan	Dingin	Normal	Benar	Ya
7	Berawan	Dingin	Normal	Benar	Ya
8	Cerah	Sejuk	Tinggi	Salah	Tidak
9	Cerah	Dingin	Normal	Salah	Ya
10	Hujan	Sejuk	Normal	Salah	Ya
11	Cerah	Sejuk	Normal	Benar	Ya
12	Berawan	Sejuk	Tinggi	Benar	Ya
13	Berawan	Panas	Normal	Salah	Ya
14	Hujan	Sejuk	Tinggi	Benar	Tidak

Langkah 1 : Mengukur Impurity Gini
(Impurity Gini dalam Cuaca)

$$Gini(Cerah) = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = 0.48$$

$$Gini(Berawan) = 1 - \left(\frac{4}{4}\right)^2 - \left(\frac{0}{4}\right)^2 = 0$$

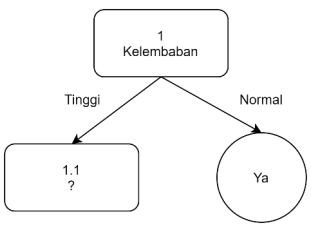
$$Gini(Hujan) = 1 - \left(\frac{4}{5}\right)^2 - \left(\frac{1}{5}\right)^2 = 0.32$$

$$Gini_A(Cuaca) = \left(\frac{5}{14}\right) \times 0.48 + \left(\frac{4}{14}\right) \times 0.375 + \left(\frac{5}{14}\right) \times 0.32 = 0.398$$

… Lakukan hal yang sama untuk Suhu Kelembaban dan Berangin

Node	Atribut	Nilai	Sum(N)	Sum(Ya)	Sum(Tidak)	Gini(Nilai)	Gini(Attribut)
1	Cuaca	Berawan	4	4	0	0	
		Hujan	5	4	1	0.32	
		Cerah	5	2	3	0.48	
							0.28571429
	Suhu	Dingin	4	4	0	0	
		Panas	4	2	2	0.5	
		Sejuk	6	4	2	0.4	
							0.33333333
	Kelembaban	Tinggi	7	3	4	0.49	
		Normal	7	7	0	0	
							0.24489796
	Berangin	Salah	8	6	2	0.375	
		Benar	6	2	4	0.444	
							0.4047619

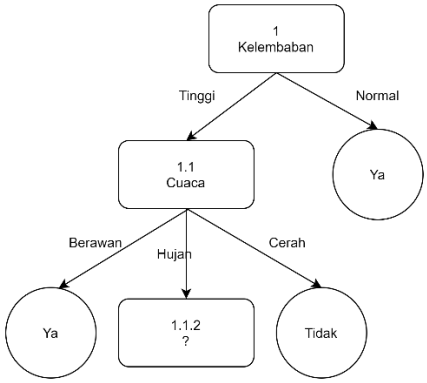
Kelembaban memiliki Gini Impurity paling rendah, jadi gunakan di root node.



Kemudian dihitung Gini untuk atribut kelembaban tinggi

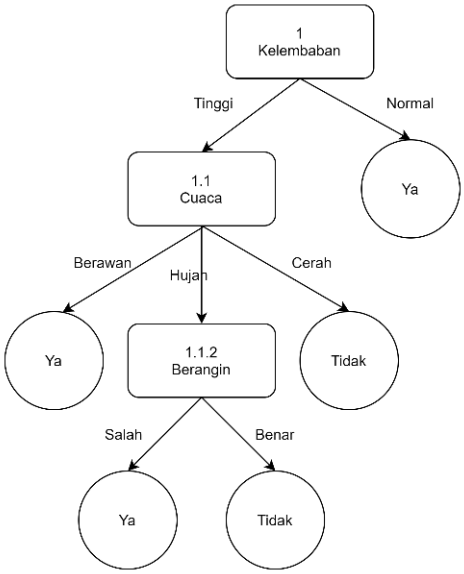
Node	Atribut	Nilai	Sum(N)	Sun(Ya)	Sum(Tidak)	Gini(Nilai)	Gini(Atribut)
1	Cuaca		2	2	0	0	
			2	1	1	0.5	
			3	0	3	0	
							0.14285714
	Suhu		0	0	0	0	
			3	1	2	0.444	
			4	2	2	0.5	
							0.47619048
	Berangin		4	2	2	0.5	
			3	2	1	0.44	
							0.47619048

Karena Cuaca Memiliki Impurity Gini terendah, maka dijadikan leaf



Node	Atribut	Nilai	Sum(N)	Sun(Ya)	Sum(Tidak)	Gini(Nilai)	Gini(Attribut)
	Suhu	Dingin	0	0	0	0	
		Panas	0	0	0	0	
		Sejuk	2	1	1	0.5	
							0.5
	Berangin	Salah	1	1	0	0	
		Benar	1	0	1	0	
							0

Berangin Memiliki Impurity Gini Terendah



K. Prunning

Prunning adalah proses yang dilakukan untuk memotong atau menghilangkan beberapa cabang (branches) yang tidak diperlukan.

L. Confusion Matrix

Confusion matrix merupakan sebuah metode untuk evaluasi yang menggunakan tabel matrix. Hasil Evaluasi dengan menggunakan confusion matrix menghasilkan nilai akurasi, serta laju error. Akurasi menyatakan jumlah data yang diklasifikasikan benar setelah dilakukan proses pengujian, sedangkan laju error digunakan untuk menghitung kesalahan identifikasi. Untuk menghitung akurasi adalah

$$Akurasi = \frac{TP + TN}{TP + FN + FP + FN} \times 100$$

Dimana TP adalah True Positive yaitu jumlah data positif yang terklasifikasi dengan benar oleh sistem, TN adalah True Negative yaitu jumlah data negatif yang terklasifikasi dengan benar oleh sistem, FN False Negative yaitu jumlah data negatif namun terklasifikasi salah oleh system dan FP adalah False Positive, yaitu jumlah data positif namun terklasifikasi salah oleh system. Laju error dapat dihitung sebagai berikut :

$$Laju\ error = \frac{Jumlah\ data\ diidentifikasi\ salah}{Jumlah\ seluruh\ data} \times 100\%$$