Classification

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, yaitu:

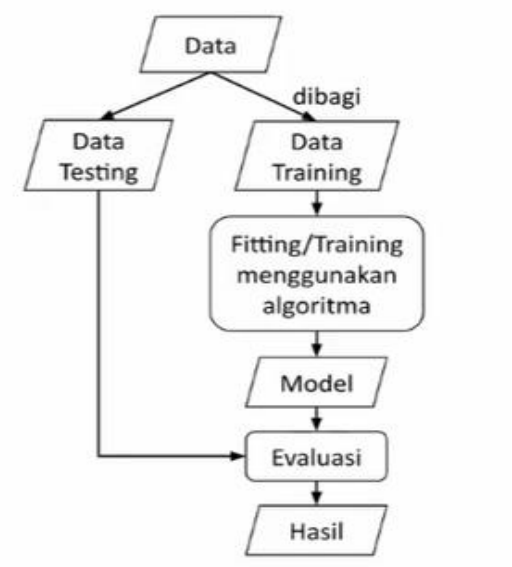
1. *Learning (fase training)*, dimana algoritma klasifikasi dibuat untuk menganalisa data training lalu direpresentasikan dalam bentuk rules.
2. *Tahap klasifikasi*, dimana data testing digunakan untuk memperkirakan akurasi dari rules yang dihasilkan pada tahap learning

Penerapan Klasifikasi :

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

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



Decision Tree

Decision tree (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 flowchart 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

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.

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

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.

Decision Tree : ID3 (Iterative Dichotomiser 3)

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.

Langkah-langkah Decision Tree dengan Algoritma ID3:

1. Pohon dimulai dengan sebuah simpul yang mereperesentasikan sampel data pelatihan yaitu dengan membuat simpul akar.
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.
3. Cabang akan dibuat untuk setiap nilai pada atribut dan data sampel akan dipartisi lagi.
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.
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.

Decision Tree : 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.