6

Pohon Keputusan

> "Sometimes you make the right decision, sometimes you make the decision right."
>
> Phil McGraw

Bab ini akan menjelaskan salah satu varian pohon keputusan yaitu ID3 oleh Quinlan [29, 30] yang terinspirasi oleh teori informasi [31]. Algoritma ini sudah cukup tua, tetapi layak dimengerti. ID3 adalah salah satu varian dari *supervised learning*.

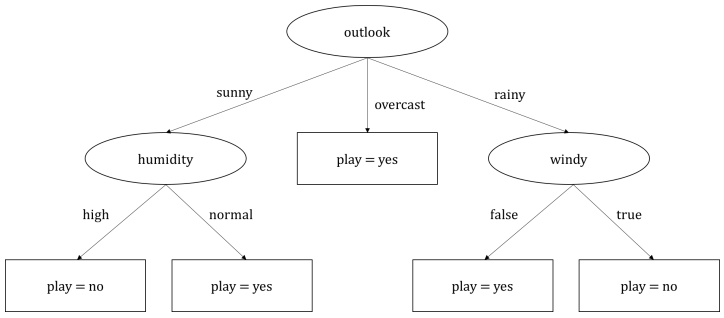
6.1 Inductive Learning

Salah satu bentuk "kecerdasan" sederhana kemungkinan adalah dalam bentuk aturan (*rule*) yang merepresentasikan pengetahuan. Misalkan, untuk menentukan apakah suatu pasien terserang penyakit tertentu, dokter mencari tahu gejala-gejala yang ada. Berdasarkan gejala-gejala yang ada, dokter memutuskan bahwa pasien memiliki suatu penyakit. Pada zaman dahulu, peneliti mentranskripsi aturan-aturan (*if then*) eksplisit (berdasarkan pengetahuan ahli) untuk membuat agen cerdas (*expert system*). Aturan sangat berguna, tetapi proses transkripsi pengetahuan sang ahli menjadi aturan formal (matematis) adalah hal yang sulit.

Pada era *big data* seperti sekarang, kita dapat mengotomatisasi hal tersebut dengan membuat aturan-aturan secara otomatis berdasarkan contoh data yang ada (*machine learning*). Pendekatan ini disebut *inductive learning*, yaitu mengembangkan aturan klasifikasi yang dapat menentukan kelas suatu *instans* berdasarkan nilai atributnya (*feature vector*). Cara paling sederhana diberikan pada subbab 3.3 yaitu mendaftarkan seluruh kemungkinan aturan yang ada, kemudian menghapus yang kurang cocok. Algoritma lebih baik adalah dengan membangun pohon keputusan (*decision tree*).

6.2 ID3

Seperti yang dijelaskan pada subbab sebelumnya, *decision tree* adalah varian dari *inductive learning*. ID3 adalah salah satu algoritma varian *decision tree* [30]. *Decision tree* dibangun berdasarkan asumsi bila atribut yang ada memberikan informasi yang cukup memadai maka kita mampu membangun *decision tree* yang mampu mengklasifikasikan seluruh instans di *training data* [30]. Akan tetapi, kita tentunya ingin melakukan generalisasi, yaitu *decision tree* yang juga mampu mengklasifikasikan objek dengan benar untuk instans yang tidak ada di *training data* (*unseen instances*). Oleh karena itu, kita harus mampu mencari hubungan antara kelas dan nilai atribut.



Gambar 6.1. *Final decision tree*

id	outlook	temperature	humidity	windy	play (class)
1	sunny	hot	high	false	no
2	sunny	hot	high	true	no
3	overcast	hot	high	false	yes
4	rainy	mild	high	false	yes
5	rainy	cool	normal	false	yes
6	rainy	cool	normal	true	no
7	overcast	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	overcast	mild	high	true	yes
13	overcast	hot	normal	false	yes
14	rainy	mild	high	true	no

Tabel 6.1. Contoh dataset *play tennis* (UCI machine learning repository)

Strategi pembangunan ID3 adalah berdasarkan *top-down* rekursif. Pertama, kita pilih atribut untuk *root* pohon, lalu membuat cabang untuk setiap nilai atribut yang mungkin. Untuk masing-masing cabang, kita buat *subtree*. Kita hentikan proses ini ketika kita sudah mencapai *leaf* (tidak bisa mencabang lebih jauh). *Leaf* ditandai apabila seluruh instans pada cabang tersebut memiliki kelas yang sama. Atribut yang sudah dipilih pada *ancestor* tidak akan dicoba pada percabangan di cabang tertentu.

Sebagai contoh, perhatikanlah Gambar 6.1 yang merupakan hasil ID3 untuk Tabel 6.1. Bentuk elips merepresentasikan nama atribut, sementara *edge* (panah) merepresentasikan nilai atribut. Bentuk segi empat merepresentasikan klasifikasi kelas (*leaf*). Pohon keputusan pada Gambar 6.1 dapat dikonversi menjadi kumpulan aturan klasifikasi berbentuk logika preposisi dengan menelusuri setiap cabang pada pohon tersebut, yaitu:

- if *outlook=sunny* and *humidity=high* then *play=no*
- if *outlook=sunny* and *humidity=normal* then *play=yes*
- if *outlook=overcast* then *play=yes*
- if *outlook=rainy* and *windy=false* then *play=yes*
- if *outlook=rainy* and *windy=true* then *play=no*

Pada setiap langkah membangun ID3, kita menggunakan ***information gain*** untuk memilih kandidat atribut terbaik. *Information gain* mengukur kemampuan suatu atribut untuk memisahkan *training data* berdasarkan kelas [7].

Sebelum masuk ke perumusan *information gain*, kami akan memperkenalkan **entropy** terlebih dahulu. Entropy (derajat ketidakteraturan) adalah informasi yang dibutuhkan untuk memprediksi sebuah kejadian, diberikan distribusi probabilitas. Secara matematis, entropy didefinisikan pada persamaan 6.1.

$$\text{entropy}(a, b) = -a \log a - b \log b \tag{6.1}$$

Kita juga definisikan **Info** sebagai persamaan 6.2.

$$\text{Info}(c_1, c_2, \ldots, c_N) = \text{entropy}(\frac{c_1}{\sum_j^N c_j}, \frac{c_2}{\sum_j^N c_j}, \ldots, \frac{c_N}{\sum_j^N c_j}) \tag{6.2}$$

dimana c_i adalah jumlah instans diklasifikasikan sebagai kelas ke-i (atau secara lebih umum, ke *node*-i). *Information gain* dihitung sebagai persamaan 6.3.

$$IG(c_1, c_2, \ldots, c_N) = \text{Info}(c_1, c_2, \ldots, c_N) - \sum_{v \epsilon V} \frac{c^v}{\sum_{i=1}^N c_i} \text{Info}({c_1}^v, {c_2}^v, \ldots, {c_N}^v) \tag{6.3}$$

dimana c_i adalah jumlah instans untuk kelas ke-i, v adalah nilai atribut, c^v adalah jumlah instans ketika dicabangkan dengan nilai atribut v, c_x^v adalah jumlah instans kelas saat percabangan. *Information gain* dapat dimaknai sebagai pengurangan entropy karena melakukan percabangan.

Sebagai contoh, mari kita hitung *Information gain* untuk atribut *outlook* sebagai *root*. Dari keseluruhan data terdapat 9 instans untuk $play = yes$ dan 5 instans untuk $play = no$. Kita hitung Info semesta sebagai (log basis 2)

$$\begin{aligned}\text{Info}([9,5]) &= \text{entropy}\left(\left[\frac{9}{14},\frac{5}{14}\right]\right)\\ &= -\frac{9}{14}\log\left(\frac{9}{14}\right) - \frac{5}{14}\log\left(\frac{5}{14}\right)\\ &= 0.940\end{aligned}$$

Kita hitung *entropy* untuk masing-masing nilai atribut *outlook* sebagai berikut:

- $outlook = sunny$
  Ada dua instans dengan $play = yes$ dan tiga instans dengan $play = no$ saat $outlook = sunny$, dengan demikian kita hitung Info-nya.

$$\begin{aligned}\text{Info}([2,3]) &= \text{entropy}\left(\frac{2}{5},\frac{3}{5}\right)\\ &= -\frac{2}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right)\\ &= 0.971\end{aligned}$$

- $outlook = overcast$
  Ada empat instans dengan $play = yes$ dan tidak ada instans dengan $play = no$ saat $outlook = overcast$, dengan demikian kita hitung Info-nya.

$$\begin{aligned}\text{Info}([4,0]) &= \text{entropy}\left(\frac{4}{4},\frac{0}{4}\right)\\ &= -\frac{4}{4}\log\left(\frac{4}{4}\right) - \frac{0}{4}\log\left(\frac{0}{4}\right)\\ &= 0\end{aligned}$$

  Perhatikan log 0 pada matematika adalah tidak terdefinisi, tapi kita anggap 0 log 0 sebagai 0 dalam komputasi.
- $outlook = rainy$
  Ada tiga instans dengan $play = yes$ dan dua instans dengan $play = no$ saat $outlook = rainy$, dengan demikian kita hitung Info-nya.

$$
\begin{aligned}
Info([3,2]) &= entropy\left(\frac{3}{5},\frac{2}{5}\right) \\
&= -\frac{3}{5}\log\left(\frac{3}{5}\right) - \frac{2}{5}\log\left(\frac{2}{5}\right) \\
&= 0.971
\end{aligned}
$$

Kita hitung *information gain* untuk atribut *outlook* sebagai

$$
\begin{aligned}
IG(outlook) &= \text{Info}\left([9,5]\right) - \\
&\quad \left(\frac{5}{14} \times \text{Info}\left([3,2]\right) + \frac{4}{14} \times \text{Info}([4,0]) + \frac{5}{14} \times \text{Info}([3,2])\right) \\
&= 0.940 - \left(\frac{5}{14} \times 0.971 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.971\right) \\
&= 0.940 - 0.693 \\
&= 0.247
\end{aligned}
$$

Dengan metode yang sama, kita hitung *information gain* untuk atribut lainnya.

- $IG(temperature) = 0.029$
- $IG(humidity) = 0.152$
- $IG(windy) = 0.048$

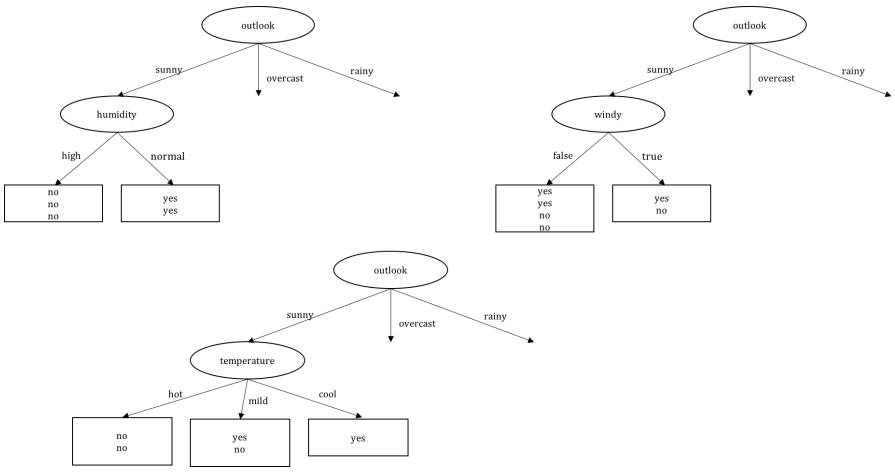
Dengan demikian, kita memilih atribut *outlook* sebagai *root*.

Kita lanjutkan lagi membuat *subtree* setelah memilih atribut *outlook* sebagai *root*. Kita hitung atribut yang tepat pada cabang $outook = sunny$, seperti diilustrasikan pada Gambar 6.2.

Pada $outlook = sunny$, terdapat dua instans dengan kelas $play = yes$ dan tiga instans dengan kelas $play = no$. Kita hitung *information gain* saat melanjutkan cabang dengan atribut *humidity*.

$$
\begin{aligned}
IG(humidity) &= \text{Info}([2,3]) - \left(\frac{3}{5} \times \text{Info}([0,3]) + \frac{2}{5} \times \text{Info}([2,0])\right) \\
&= 0.971 - 0 \\
&= 0.971
\end{aligned}
$$

Untuk setiap kedalaman, kita coba menggunakan atribut yang belum pernah dicoba pada level-level lebih atas, seperti yang sudah diilutrasikan. Proses ini dilanjutkan sampai kita tidak bisa atau tidak perlu mencabang lagi.



Gambar 6.2. Percabangan

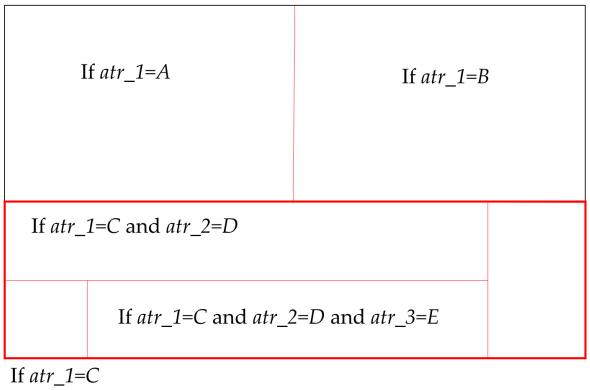
6.3 Isu pada ID3

Pada algoritma *decision tree* secara umum, terdapat beberapa isu diantara lain [7, 32]:

1. Mudah overfitting
2. Masalah menangani atribut kontinu
3. *Information gain* memiliki bias terhadap atribut yang memiliki banyak nilai (*highly-branching attributes*)
4. Data dengan *missing value*. Beberapa sel pada tabel dataset tidak terisi.
5. Data dengan *unseen value*. Misal nilai atribut yang tidak pernah dilihat pada *training data*, muncul saat *testing*.

6.4 Pembagian Ruang Konsep

Ada hubungan antara algorithma *decision tree* dan model linear. Pada model linear, kita semacam membagi-bagi ruang konsep (semesta data) menjadi ruang per kelas menggunakan garis pembatas linear. *Decision tree* melakukan hal yang hampir sama, karena percabangan *decision tree* dapat dianggap sebagai linear. Sebagai contoh perhatikan ilustrasi Gambar 6.3, dimana semesta adalah suatu ruang konsep. Tiap ruang merepresentasikan suatu cabang (dari *root* sampai *leaf*) pada *decision tree*. Garis-garis yang membentuk ruang-ruang pemisah disebut sebagai ***decision boundary***.



Gambar 6.3. Ilustrasi pembagian ruang konsep

Soal Latihan

6.1. Isu
Pada subbab 6.3, telah disebutkan isu-isu yang ada pada ID3, sebutkan dan jelaskan bagaimana cara menangani masing-masing isu tersebut!

6.2. Gain Ratio
Selain *information gain*, kita dapat menggunakan cara lain untuk memilih atribut bernama *gain ratio*. Jelaskan perbedaan keduanya! Yang mana lebih baik?

6.3. C4.5
ID3 disempurnakan kembali oleh pembuat aslinya menjadi C4.5. Jelaskanlah perbedaan ID3 dan C4.5, beserta alasan strategi penyempurnaan!

6.4. Pruning
Jelaskan apa itu *pruning* dan bagaimana cara melakukan *pruning* untuk *decision tree*!

6.5. Association Rule
Terdapat beberapa algoritma *association rule* yang membentuk aturan-aturan seperti *decision tree*.
(a) Jelaskanlah algoritma PRISM dan Apriori!
(b) Jelaskan perbedaan *association rule* dan *inductive learning*!

6.6. Final Decision Tree
Lanjutkanlah proses konstruksi ID3 untuk Tabel 6.1 hingga membentuk *decision tree* akhir seperti pada Gambar 6.1!

6.7. Variant
Jelaskanlah *Random Forest*, *Bagging* dan *Boosting* pada *Decision Tree*!