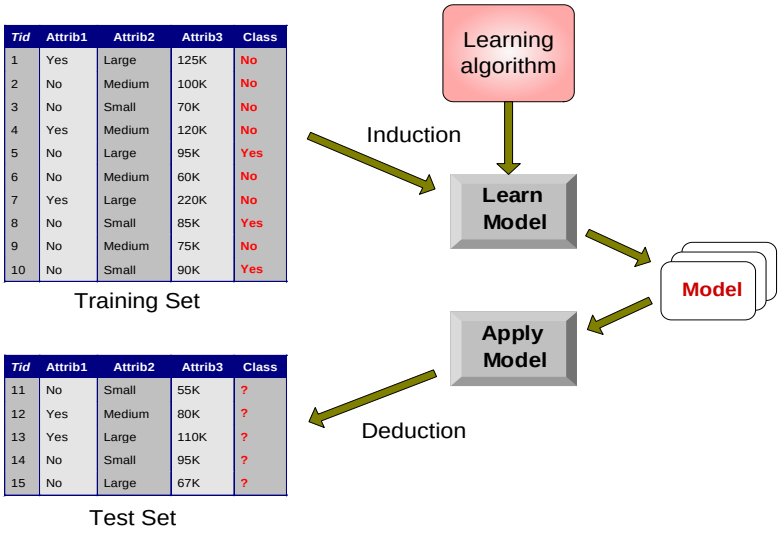
Kuliah 6

Klasifikasi: Konsep Dasar dan Pohon Keputusan (Decision Trees)

Klasifikasi: Definisi

- Diberikan sekumpulan *record* (himpunan data latih / *training set*)
  - Tiap *record* mengandung suatu himpunan atribut, salah satu atributnya adalah sebagai kelas/*class*.
- Temukan sebuah model yang menunjukkan suatu kelas atribut merupakan fungsi dari nilai atribut lainnya
- Tujuan: previously unseen records (record yang sebelumnya tak diketahui kelasnya) harus dapat ditetapkan kelasnya seakurat mungkin.
  - Suatu himpunan data uji (*test set)* digunakan untuk menentukan akurasi model. Umumnya, data set yang ada dibagi menjadi himpunan data latih (training set) dan himupnan data uji (test set), dengan data latih digunakan untuk membangun model dan data uji digunakan untuk melakukan validasi model

Ilustrasi Tahapan pada Klasifikasi



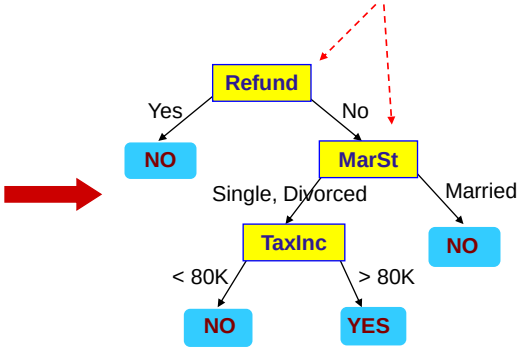
Teknik-teknik Klasifikasi

- Metode Berbasis Pohon Keputusan (Decision Tree based Methods)
- Metode Berbasis Aturan (Rule-based Methods)
- Penalaran Berbasis Memori (Memory based reasoning)
- Jaringan Syaraf Tiruan (Neural Networks)
- Naïve Bayes dan Bayesian Belief Networks
- Support Vector Machines

Contoh suatu Pohon Keputusan (Decision Tree)

Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

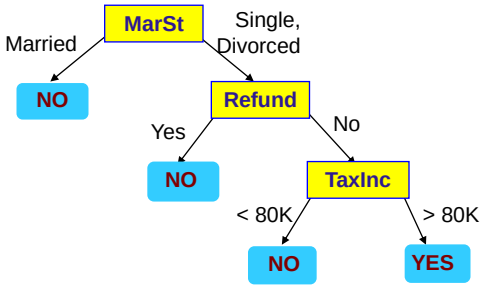
Data Latih (Training Data)



Model: Pohon Keputusan (Decision Tree)

Contoh Lain Pohon Keputusan (Decision Tree)

Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Ada lebih dari satu pohon (tree) yang dapat dibangun dari data set ini

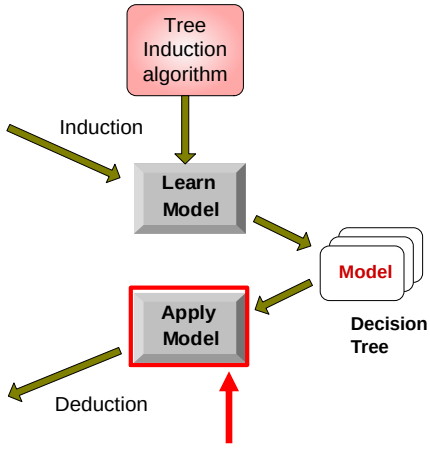
Tahapan Klasifikasi pada Pohon Keputusan (Decision Tree)

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

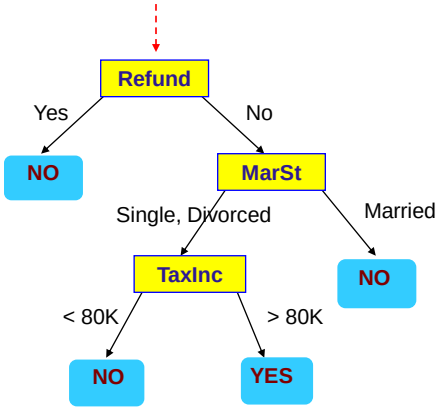
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Menerapkan Model pada Data Uji

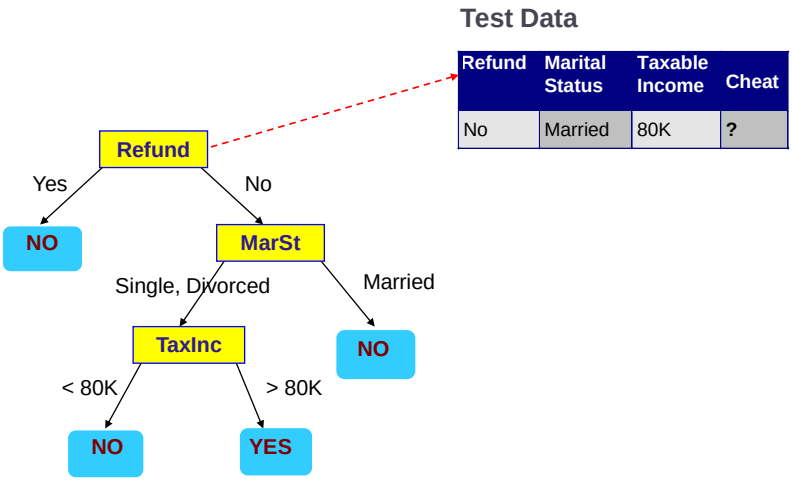
Mulai dari node akar (root of tree)



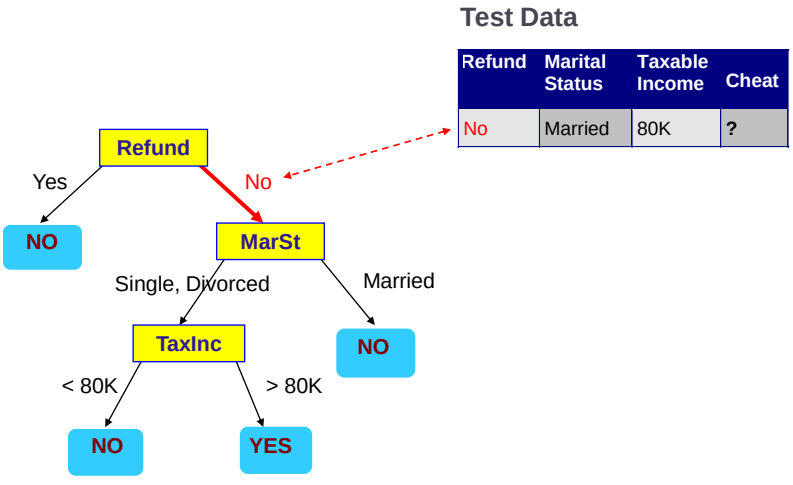
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

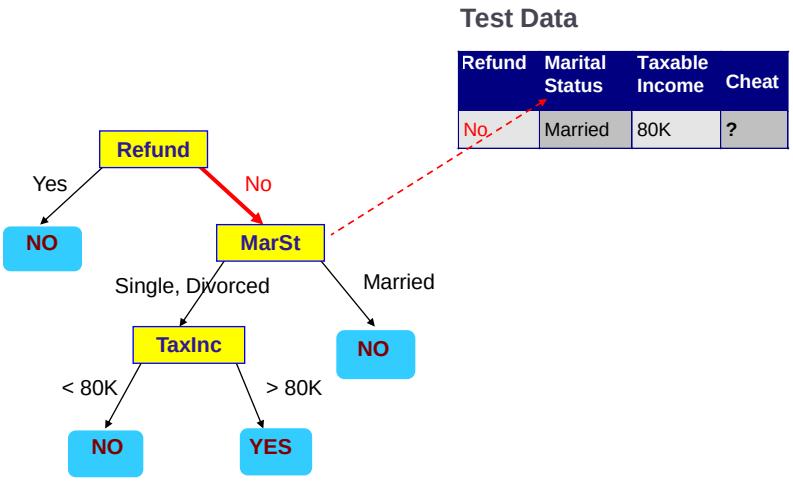
Menerapkan Model pada Data Uji



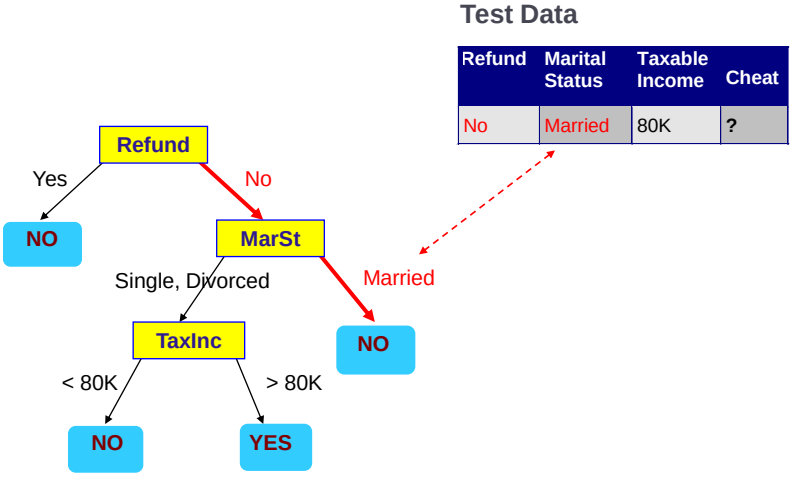
Menerapkan Model pada Data Uji



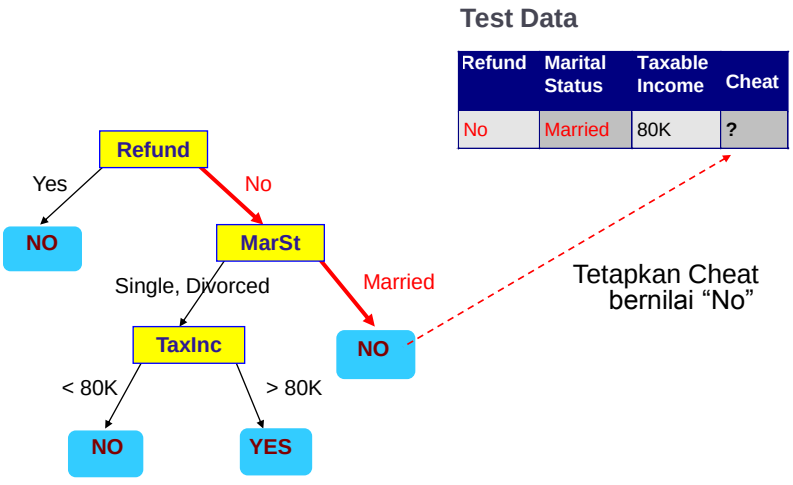
Menerapkan Model pada Data Uji



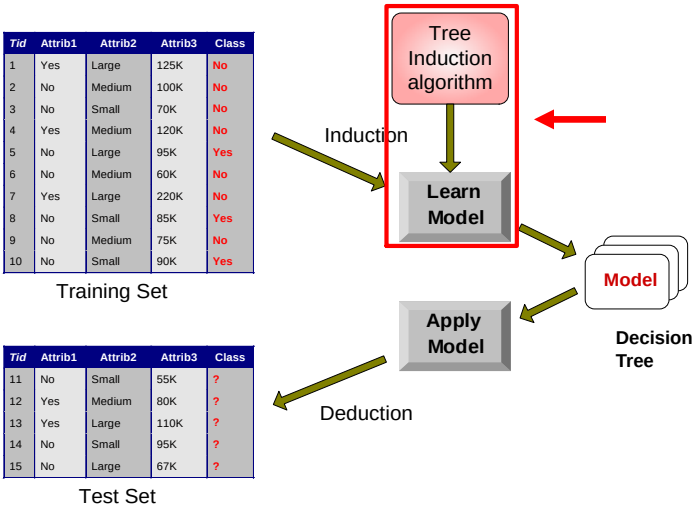
Menerapkan Model pada Data Uji



Menerapkan Model pada Data Uji



Tahapan Klasifikasi pada Pohon Keputusan (Decision Tree)



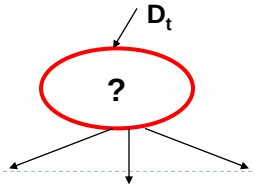
Induksi Pohon Keputusan (Decision Tree Induction)

- Ada banyak algoritme:
  - Hunt's Algorithm (salah satu yang paling awal)
  - CART
  - ID3, C4.5
  - SLIQ, SPRINT

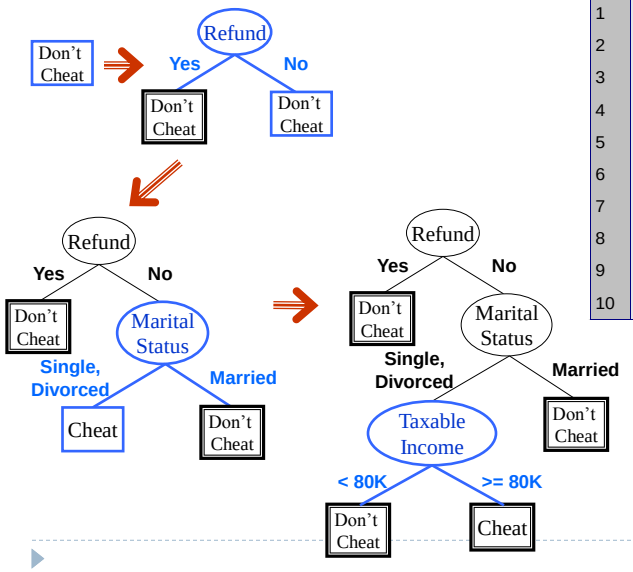
Struktur Umum Hunt's Algorithm

- Diketahui D_t adalah suatu rekord data latih (training records) yang mencapai suatu simpul (node) t (reach a node t)
- Prosedur umumnya:
  - Jika D_t mengandung records yang termasuk ke dalam kelas yang sama dengan y_t, maka t simpul daun (leaf node) yang diberi label dengan y_t
  - Jika D_t adalah suatu himpunan kosong, maka t adalah simpul daun (leaf node) yang diberi label dengan kelas default, y_d
  - Jika D_t mengandung record yang termasuk ke dalam lebih dari satu kelas, gunakan suatu uji atribut (attribute test) untuk memisahkan data ke dalam sub himpunan data (subset) yang lebih kecil. Lakukan secara rekursif prosedur di atas untuk setiap sub himpunan data (subset).

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Hunt's Algorithm



Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Induksi Pohon

- Strategi Greedy.
  - Bagi/pisahkan (Split) record berdasarkan suatu uji atribut (attribute test) yang mengoptimalkan kriteria yang ditetapkan
- Isu
  - Menentukan membagi (split) record
    - Bagaimana menspesifikasikan kondisi uji atribut (the attribute test condition)?
    - Bagaimana menentukan pemisahan terbaik (best split)?
  - Menentukan kapan berhenti melakukan pemisahan record (splitting)

Induksi Pohon

- StrategiGreedy .
  - Bagi/pisahkan (Split) record berdasarkan suatu uji atribut (attribute test) yang mengoptimalkan kriteria yang ditetapkan

- Isu
  - Menentukan membagi (split) record
    - Bagaimana menspesifikasikan kondisi uji atribut (the attribute test condition)?
    - Bagaimana menentukan pemisahan terbaik (best split)?
  - Menentukan kapan berhenti melakukan pemisahan record (splitting)

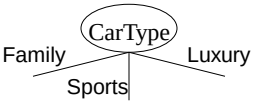
Bagaimana Menspesifikasikan Kondisi Uji Atribut (Attribute Test Condition)?

- Bergantung pada tipe atributnya
  - Nominal
  - Ordinal
  - Kontinu

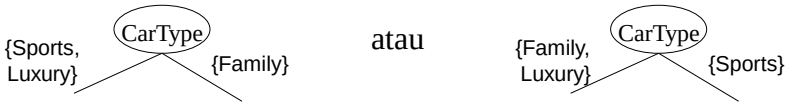
- Bergantung pada cara melakukan split
  - 2-way split
  - Multi-way split

Pemisahan (Splitting) Berdasarkan Atribut Nominal

- Multi-way split: Bagi ke dalam sebanyak mungkin partisi sesuai dengan nilai yang berbeda yang dimiliki oleh atribut tersebut

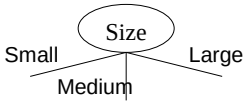


- Binary split: Bagi nilai ke dalam dua subhimpunan
  Perlu mencari pemartisian yang optimal

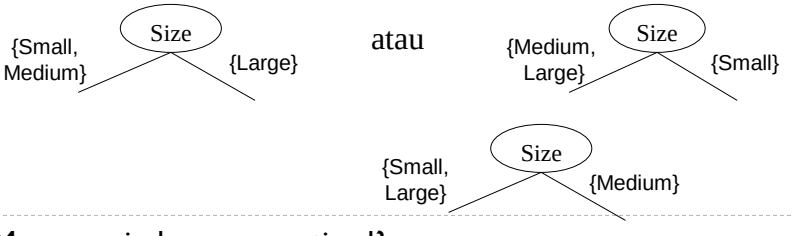


Pemisahan Berdasarkan Atribut Ordinal

- Multi-way split: Bagi ke dalam sebanyak mungkin partisi sesuai dengan nilai yang berbeda yang dimiliki oleh atribut tersebut



- Binary split: Bagi nilai ke dalam dua subhimpunan
  Perlu mencari pemartisian yang optimal

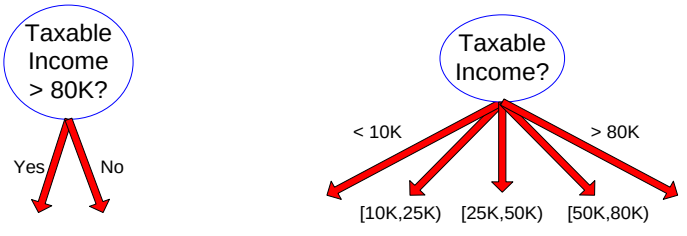


- Mana pemisahan yang optimal?

Pemisahan (Splitting) Berdasarkan atribut kontinu

- Ada dua cara penanganan yang berbeda
  - Diskritisasi untuk membentuk sebuah atribut kategorikal ordinal
    - Statis – melakukan diskritisasi pada tahap awal
    - Dinamis – jangkauan (ranges) dapat diperoleh dengan bucketing dengan interval yang sama equal interval bucketing, bucketing dengan frekuensi yang sama (equal frequency bucketing/percentiles), atau klustering.
  - Keputusan Biner (Binary Decision):$(A < v)$ or $(A \geq v)$
    - pertimbangkan semua kemungkinan pemisahan (splits) dan temukan yang terbaik
    - membutuhkan komputasi yang intensif

Pemisahan (Splitting) Berdasarkan Atribut Kontinu



(i) Binary split

(ii) Multi-way split

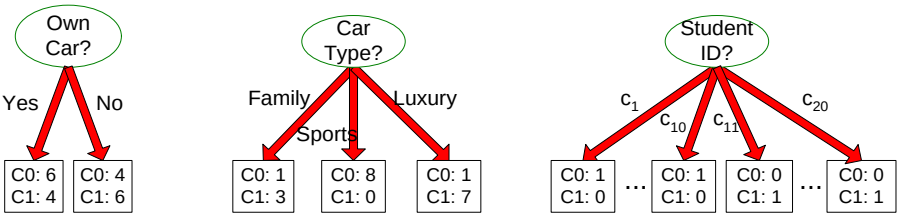
Induksi Pohon

- StrategiGreedy .
  - Bagi/pisahkan (Split) record berdasarkan suatu uji atribut (attribute test) yang mengoptimalkan kriteria yang ditetapkan

- Isu
  - Menentukan membagi (split) record
    - Bagaimana menspesifikasikan kondisi uji atribut (the attribute test condition)?
    - Bagaimana menentukan pemisahan terbaik (best split)?
  - Menentukan kapan berhenti melakukan pemisahan record (splitting)

Bagaimana menentukan pemisahan terbaik (Best Split)

Sebelum pemisahan (Splitting): ada 10 records yang termasuk kelas 0, dan 10 records yang termasuk kelas 1



Kondisi uji yang bagaimana yang terbaik?

Bagaimana menentukan pemisahan terbaik (Best Split)

- Pendekatan Greedy :
  - Simpul (nodes) dengan distribusi kelas yang homogen (homogeneous class distribution) lebih disukai (diutamakan)
- Diperlukan suatu ukuran ketidakhomogenan/ketidakmurnian mode (node impurity):

C0: 5
C1: 5

Tidak homogen (Non-homogeneous),

Ketidakhomogenan/ketidakmurniaan berderajat tinggi (High degree of impurity)

Kelas 0 ada 5 record

Kelas 1 ada 5 record

C0: 9
C1: 1

Homogen,

Ketidakhomogenan/ketidakmurniaan berderajat tinggi (Low degree of impurity)

Kelas 0 ada 9

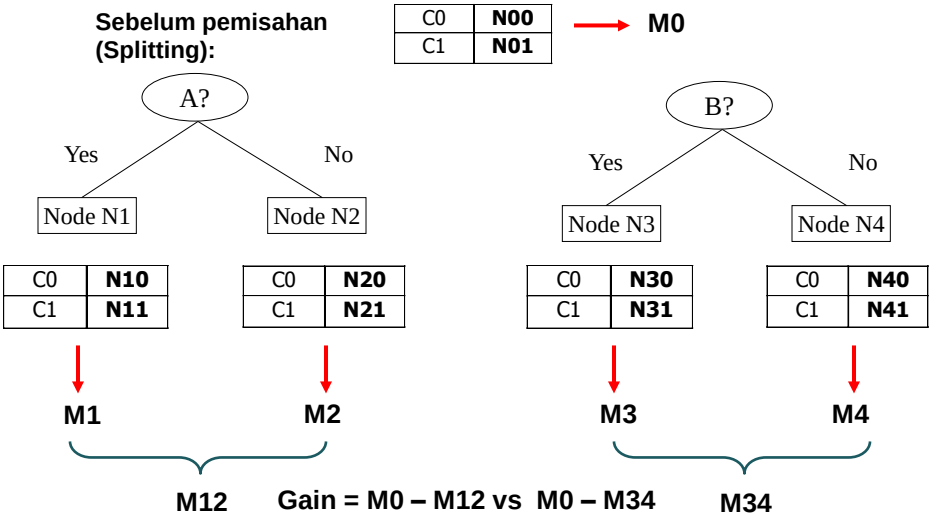
Kelas 1 ada 1

→ Distrbusi kelasnya relatif homogen

Ukuran untuk Ketidakhomogenan Simpul (Node Impurity)

- Gini Index
- Entropy
- Misclassification error

Bagaimana menentukan pemisahan terbaik (Best Split)



- N_{rc} = Jumlah record pada node ke-r yang termasuk dalam kelas c
  Di mana c bernilai 0 atau 1 (kelas biner)

Ukuran Impurity: GINI

- Gini Index untuk suatu node t :

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

(NOTE: $p(j \mid t)$ adalah frekuensi relatif dari kelas j pada node t).

  - Maximum (1 - $1/n_c$) terjadi ketika record terdistribusi secara merata/sama pada semua kelas yang menunjukkan bahwa informasi pada atribut tsb adalah yang paling tidak menarik (least interesting information)
  - Minimum (0.0) terjadi ketika semua records adalah angota suatu kelas yang menunjukkan bahwa informasi pada atribut tsb adalah paling penting/menarik

C1	0
C2	6
Gini=0.000	

C1	1
C2	5
Gini=0.278	

C1	2
C2	4
Gini=0.444	

C1	3
C2	3
Gini=0.500	

Contoh perhitungan GINI

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Gini = 1 – P(C1)² – P(C2)² = 1 – 0 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Gini = 1 – (1/6)² – (5/6)² = 0.278

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Gini = 1 – (2/6)² – (4/6)² = 0.444

Pemisahan (Splitting) berbasis GINI

- Digunakan pada CART, SLIQ, SPRINT.
- Ketika suatu node dipisahkan (split) ke dalam sejumlah k partisi (children), kualitas dari hasil pemisahan (split) ini dihitung dengan persamaan berikut:

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

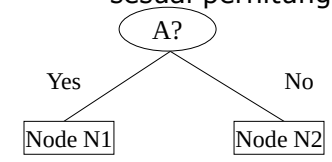
di mana, n_i = jumlah record pada child i,

n = jumlah record pada node p.

Atribut biner: Perhitungan GINI Index

- Pisahkan (Splits) ke dalam dua partisi
- Efek dari Effect of Weighing partitions:
  - Partisi yang lebih murni (pure) adalah B, sesuai perhitungan berikut:

	Parent
C1	6
C2	6
Gini = 0.500	

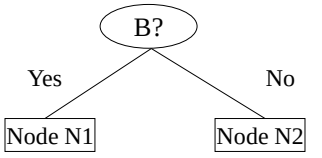


	N1	N2
C1	4	2
C2	3	3
Gini=0.486		

Gini(N1)
= 1 – $(4/7)^2$ – $(3/7)^2$
= 0.4898

Gini(N2)
= 1 – $(2/5)^2$ – $(3/5)^2$
= 0.48

Gini (children)
= 7/12 * 0.4898+ 5/12 * 0.48
= 0.486



	N1	N2
C1	1	5
C2	4	2
Gini=0.375		

Gini (B) < Gini (A)
Pilih Atribut B

Gini(N1)
= 1 – $(1/5)^2$ – $(4/5)^2$
= 0.32

Gini(N2)
= 1 – $(5/7)^2$ – $(2/7)^2$
= 0.408

Gini (children)
= 5/12 * 0.32+ 7/12 * 0.408
= 0.375

Atribut Kategorik(Categorical Attributes): Perhitungan Gini Index

- Untuk setiap nilai yang berbeda, dapatkan frekuensi (count) untuk setiap kelas dalam dataset
- Gunakan count matrix untuk membuat keputusan

Multi-way split

	CarType		
	Family	Sports	Luxury
C1	1	2	1
C2	4	1	1
Gini	0.393		

Two-way split
(find best partition of values)

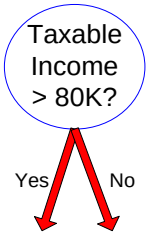
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	CarType	
	{Sports}	{Family, Luxury}
C1	2	2
C2	1	5
Gini	0.419	

Atribut Kontinu: Perhitungan Gini Index

- Gunakan keputusan biner berdasarkan satu nilai
- Ada beberapa pilihan untuk nilai pemisah (splitting value)
  - Jumlah nilai pemisah
    = jumlah nilai berbeda (distinct values) dari atribut tersebut
- Setiap nilai pemisah memiliki sebuah count matrix yang berasosiasi dengannya
  - Class counts pada setiap partisi, $A < v$ and $A \geq v$
- Metode sederhana untuk memilik nilai v terbaik
  - Untuk setiap v scan basis data untuk mendapatkan count matrix dan hitung Gini indexnya
  - Sangat tidak efisien secara komputasi Pekerjaan yang dilakukan berulang-ulang

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Atribut Kontinu: Komputasi Gini Index

- Untuk komputasi yang efisien: lakukan untuk setiap atribut hal berikut:
  - Urutkan nilai pada atribut kontinu
  - Scan nilai-nilai ini secara linear, setiap waktu lakukan pembaruan nilai pada count matrix dan lakukan perhitungan gini index
  - Pilih posisi pemisah (split position) yang memiliki gini index paling kecil

Cheat	No		No		No		Yes		Yes		Yes		No		No		No		No	
	Taxable Income																			
Sorted Values →	60		70		75		85		90		95		100		120		125		220	
Split Positions →	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		*0.300*		0.343		0.375		0.400		0.420	

Alternatif Kriteria pemecahan Berdasarkan INFO

- Entropy pada simpul (node) t didefinisikan sebagai:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

(NOTE: $p(j \mid t)$ adalah frekuensi relatif dari kelas j pada simpul (node) t).

- Ukuran kehomogenan dari suatu simpul (node)
  - Maximum (log n_c) terjadi ketika record terdistribusi secara merata/sama pada semua kelas yang menunjukkan informasi yang paling sedikit jumlahnya (implying least information)
  - Minimum (0.0) terjadi ketika semua record adalah anggota dari suatu kelas yang menunjukkan informasi yang paling banyak jumlahnya
- Komputasi berbasis Entropy mirip dengan komputasi berbasis GINI index

Contoh perhitungan Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

Pemisahan (Splitting) Berbasis INFO…

- Information Gain:

$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

Parent Node, p dipisahkan (split) ke dalam k partisi;

n_i jumlah records pada partisi i

  - Ukuran reduksi (Reduction) dalam Entropy diperoleh karena proses pemisahan (split). Pilih Split yang menghasilkan paling banyak reduksi (maksimumkan GAIN)
  - Digunakan di ID3 dan C4.5
  - Kerugian: Cenderung melakukan pemisahan yang menghasilkan sejumlah besar partisi, masing-masing berukuran kecil tapi homogen/murni/pure

Pemisahan (Splitting) Berbasis INFO…

- Gain Ratio:

$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO} \qquad SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

Parent Node, p dipisahkan (split) ke dalam k partisi;

n_i jumlah records pada partisi i

  - Sesuaikan Information Gain dengan entropy dari partisi (SplitINFO). Partisi dengan entropi yang lebih tinggi (sejumlah besar partisi kecil) dikenai penalti!
  - Used in C4.5
  - Dirancang untuk mengatasi kelemahan Information Gain

Kriteria Pemisahan Berdasarkan Kesalahan Klasifikasi (Classification Error)

- Classification error pada suatu node t, didefinisikan sebagai :

$$Error(t) = 1 - \max_i P(i \mid t)$$

- Ukuran misclassification error yang dari suatu node
  - Maximum ($1 - 1/n_c$) terjadi ketika record terdistribusi secara merata/sama pada semua kelas yang menunjukkan bahwa informasi pada atribut tsb adalah yang paling tidak menarik (least interesting information)
  - Minimum (0.0) terjadi ketika semua records adalah angota suatu kelas yang menunjukkan bahwa informasi pada atribut tsb adalah paling penting/menarik

Contoh Perhitungan Error

$$Error(t) = 1 - \max_i P(i \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Error = 1 – max (0, 1) = 1 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

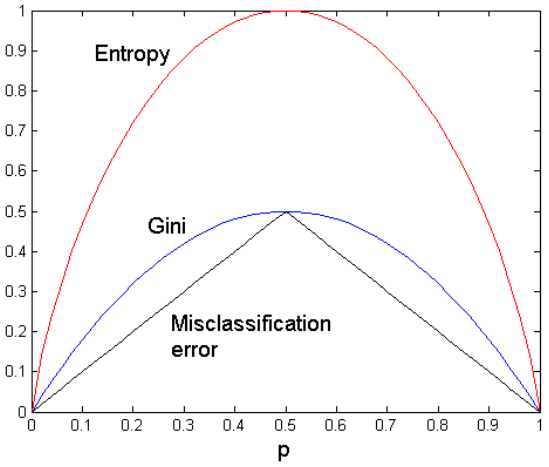
C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

Perbandingan antar Beberapa Kriteri Pemisahan

Untuk kasus 2-class (binary classification:



Induksi Pohon

- StrategiGreedy .
  - Bagi/pisahkan (Split) record berdasarkan suatu uji atribut (attribute test) yang mengoptimalkan kriteria yang ditetapkan

- Isu
  - Menentukan membagi (split) record
    - Bagaimana menspesifikasikan kondisi uji atribut (the attribute test condition)?
    - Bagaimana menentukan pemisahan terbaik (best split)?
  - Menentukan kapan berhenti melakukan pemisahan record (splitting)

Kriteria Berhenti pada Induksi Pohon (Tree Induction)

- Hentikan melakukan perluasan suatu node ketika semua record menjadi anggota kelas yang sama

- Hentikan memperluas suatu node ketika semua record memiliki nilai atribut yang mirip (similar)
- Penghentian sebelum salah satu kedua kondisi di atas terpenuhi (Early termination)

Klasifikasi Berbasis Pohon Keputusan

- Keuntungan:
  - Murah dalam menkontruksi modelnya
  - Sangat cepat ketika mengklasifikasikan record yang belum diketahui (data baru)
  - Mudah untuk mengintepretasikan hasilmuya jika pohon yang dihasilkan berukuran kecil
  - Akurasi dapat disejajarkan dengan teknik klasifikasi lain untuk dataset yang sederhana

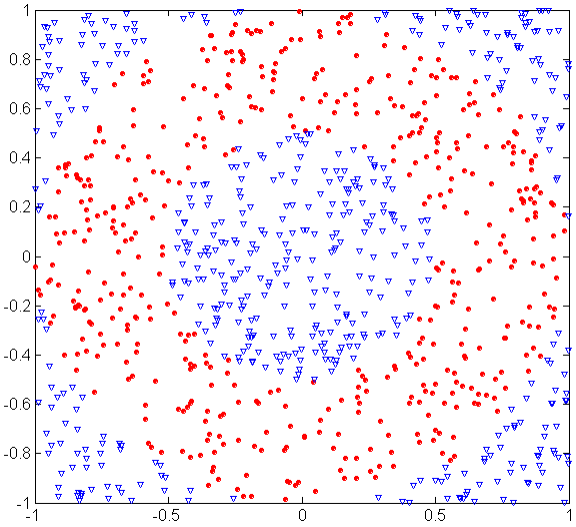
Contoh: C4.5

- Didasarkan pada teknik depth-first yang sederhana
- Menggunakan Information Gain
- Urutkan setiap atribut kontinu pada tiap node
- Perlu memuatkan semua data ke dalam memori
- Tidak sesuai untuk dataset berukuran besar
  - Perlu out-of-core sorting untuk menangani data berukuran sangat besar

Isu-isu Praktis Klasifikasi

- Underfitting dan Overfitting
- Missing Values
- Costs of Classification

Underfitting dan Overfitting (Contoh)



Ada 500 titik data yang ditandai dengan lingkaran dan 500 yang ditandai dengan segi tiga

Titik yang lingkaran

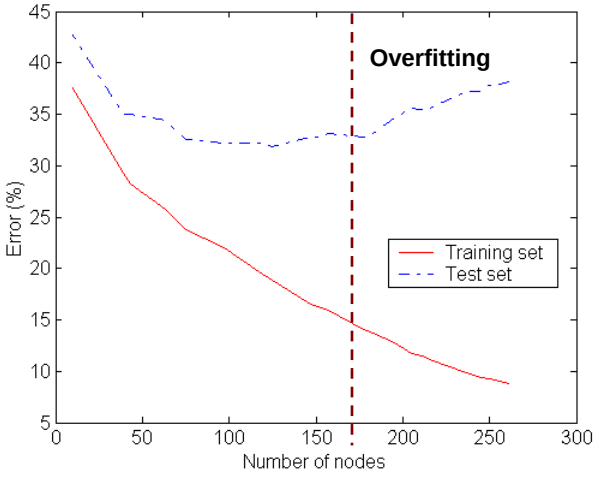
$0.5 \leq \text{sqrt}(x_1^2+x_2^2) \leq 1$

Titik yang segi tiga:

$\text{sqrt}(x_1^2+x_2^2) > 0.5$ or

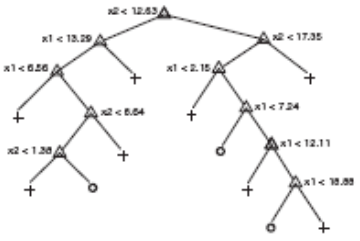
$\text{sqrt}(x_1^2+x_2^2) < 1$

Underfitting dan Overfitting

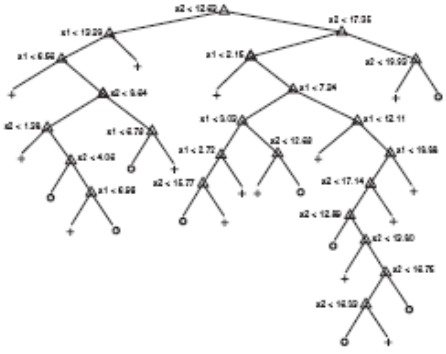


Underfitting: ketika model terlalu sederhana, kedua error hasil proses pelatihan (training) dan pengujian (testing) nilainya besar

Contoh

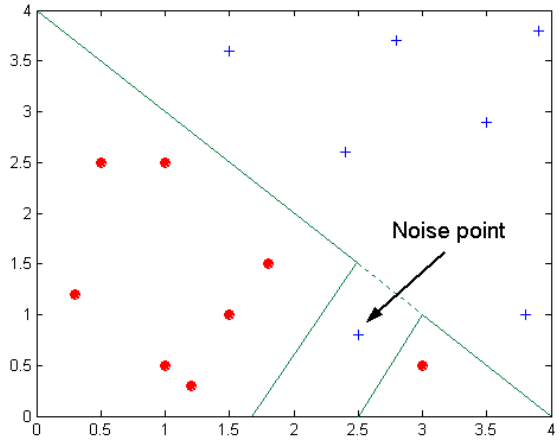


(a) Decision tree with 11 leaf nodes.



(b) Decision tree with 24 leaf nodes.

Overfitting disebabkan oleh Noise

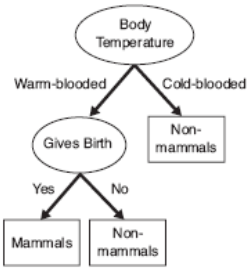


Batas keputusan (Decision boundary) terdistorsi karena adanya titik noise

Contoh

Training set

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
bat	warm-blooded	yes	no	yes	no*
whale	warm-blooded	yes	no	no	no*
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no



(b) Model M2

*** mislabeled**

Test set

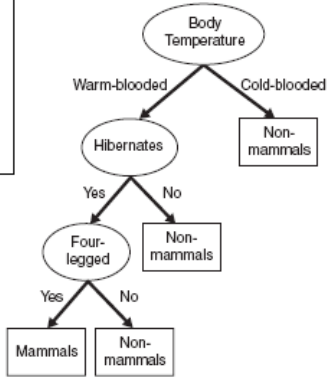
Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
human	warm-blooded	yes	no	no	yes
pigeon	warm-blooded	no	no	no	no
elephant	warm-blooded	yes	yes	no	yes
leopard shark	cold-blooded	yes	no	no	no
turtle	cold-blooded	no	yes	no	no
penguin	cold-blooded	no	no	no	no
eel	cold-blooded	no	no	no	no
dolphin	warm-blooded	yes	no	no	yes
spiny anteater	warm-blooded	no	yes	yes	yes
gila monster	cold-blooded	no	yes	yes	no

Contoh

Training set

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
bat	warm-blooded	yes	no	yes	no*
whale	warm-blooded	yes	no	no	no*
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no

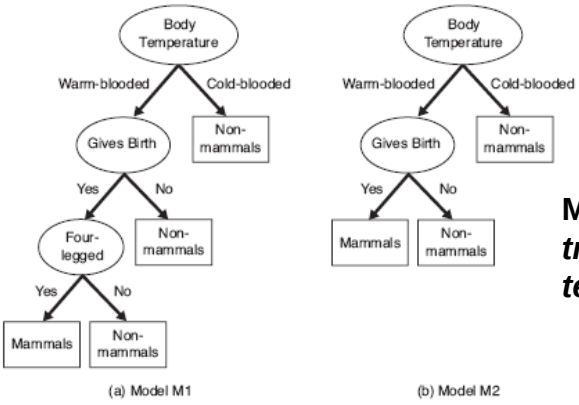
Model M1



Test set

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
human	warm-blooded	yes	no	no	yes
pigeon	warm-blooded	no	no	no	no
elephant	warm-blooded	yes	yes	no	yes
leopard shark	cold-blooded	yes	no	no	no
turtle	cold-blooded	no	yes	no	no
penguin	cold-blooded	no	no	no	no
eel	cold-blooded	no	no	no	no
dolphin	warm-blooded	yes	no	no	yes
spiny anteater	warm-blooded	no	yes	yes	yes
gila monster	cold-blooded	no	yes	yes	no

Pohon

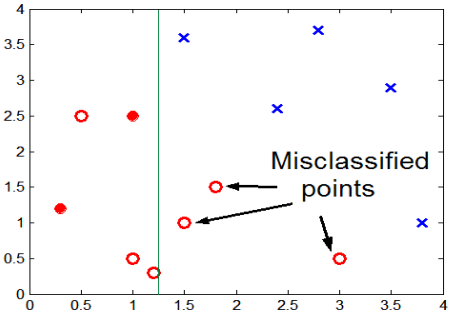


M1:
***training error* : 0%,**
***test error* : 30%**

M2:
***training error* : 20%,**
***test error* : 10%**

M1 overfitting training set

Overfitting disebabkan kurangnya data sampel (contoh)



Tidak adanya titik data pada setengah bagian yang lebih rendah pada diagram menyebabkan sulit untuk memprediksi secara benar label kelas dari daerah tersebut

- Tidak mencukupinya jumlah records untuk proses latih (training records) pada daerah tersebut menyebabkan pohon keputusan (decision tree) memprediksi sampel uji menggunakan record/data latih yang lain yang tidak relevan untuk melakukan klasifikasi

Catatan untuk Overfitting

- Overfitting menghasilkan pohon keputusan (decision trees) yang lebih kompleks dari yang dibutuhkan
- Training error tidak lagi bisa memberikan perkiraan/estimasi yang bagus mengenai performa/kinerja dari pohon/tree yang dihasilkan dalam mengklasifikasikan record baru (unknown records)
- Perlu cara lain untuk memperkirakan error

Mengestimasi Generalization Errors

- Re-substitution errors: error pada training (Σ e(t))
- Generalization errors: error pada testing (Σ e'(t))
- Metode untuk mengestimasi generalization errors:
  - Pendekatan Optimistis (Optimistic approach): e'(t) = e(t)
  - Pendekatan Pesimistis (Pessimistic approach):
    - For each leaf node: e'(t) = (e(t)+0.5)
    - Total errors: e'(T) = e(T) + N $\times$ 0.5 (N: number of leaf nodes)
    - For a tree with 30 leaf nodes and 10 errors on training (out of 1000 instances):
      Training error = 10/1000 = 1%
      Generalization error = (10 + 30$\times$0.5)/1000 = 2.5%
  - Reduced error pruning (REP):
    - Menggunakan data testing (validasi) untuk mengestimasi generalizationerror

Bagaimana Mengatasi Overfitting

- Pre-Pruning (Early Stopping Rule)
  - Hentikan algoritme sebelum menjadi "a fully-grown tree" (pohon lengkap)
  - Kondisi berhenti (Stopping) suatu node:
    - Hentikan jika semua instans (records) memiliki kelas label yang sama
    - Hentikan ketika semua nilai atribut adalah sama/similar
  - Kondisi yang lebih ketat:
    - Hentikan ketika jumlah instan/records dalam suatu node lebih kecil dari threshold yang didefinisikan oleh user/pengguna
    - Hentikan ketika distribusi kelas dari instan-2nya/record-recordnya indpenden dari fitur-fitur yang ada (e.g., gunakan χ^2 test)
    - Hentikan ketika perluasan suatu node tidak memperbaiki derajat homogenitas (imputiry) (gunakan Gini atau information gain).

Bagaimana mengatasi Overfitting…

- Post-pruning
  - Tumbuhkan pohon keputusan (decision tree) sampai menjadi pohon lengkap
  - Potong simpul-simpul (node-2) pada pohon keputusan (decision tree) di mulai dari bawah (bottom-up fashion)
  - Hitung generalization error nya. Jika generalization errornya menjadi lebih baik setelah proses pemotongan (trimming), ganti sub-pohon (sub-tree) dengan sebuah simpul daun (leaf node)
  - Label kelas dari simpul daun (leaf node) ditentukan dari label kelas mayoritas dari instan-instan (record) pada sub-pohon tersebut
  - Dapat menggunakan MDL (Minimum Desription Length) untuk post-pruning

Contoh Post-Pruning

Class = Yes	20
Class = No	10
Error = 10/30	

Training Error (Before splitting) = 10/30

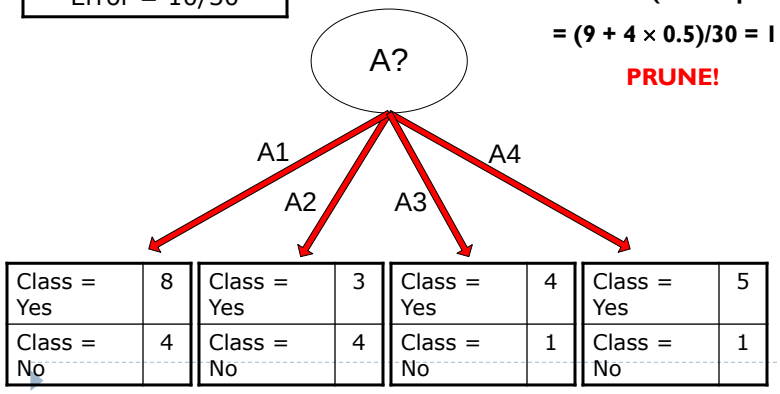
Pessimistic error = (10 + 0.5)/30 = 10.5/30

Training Error (After splitting) = 9/30

Pessimistic error (After splitting)

= (9 + 4 × 0.5)/30 = 11/30

PRUNE!

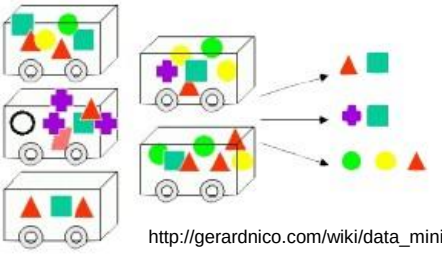


Scalable Decision Tree Induction Methods

- SLIQ (EDBT'96 — Mehta et al.)
  - Membangun indeks untuk tiap atribut dan hanya list kelas dan list atribut yang disimpan di memori
- SPRINT (VLDB'96 — J. Shafer et al.)
  - Membangun struktur data list untuk atribut
- PUBLIC (VLDB'98 — Rastogi & Shim)
  - Mengintegrasikasikan pemisahan pohon (tree splitting) dan pemangkasan pohon (tree pruning): menghentikan pertumbuhan pohon lebih awal
- RainForest (VLDB'98 — Gehrke, Ramakrishnan & Ganti)
  - Membangun sebuah AVC-list (attribute, value, class label)
- BOAT (PODS'99 — Gehrke, Ganti, Ramakrishnan & Loh)
  - Menggunakan bootstrapping untuk menciptakan beberapa sampel berukuran kecil

Catatan

- Semua slide diambil dari
  - Tan P., Michael S., & Vipin K. 2006. *Introduction to Data mining*. Pearson Education, Inc.
  - Han J & Kamber M. 2006. *Data mining – Concept and Techniques.* Morgan-Kauffman, San Diego
  - Dan sumber lainnya yang relevan
- Topik selanjutnya: *Support Vector Machine*



http://gerardnico.com/wiki/data_mining/association