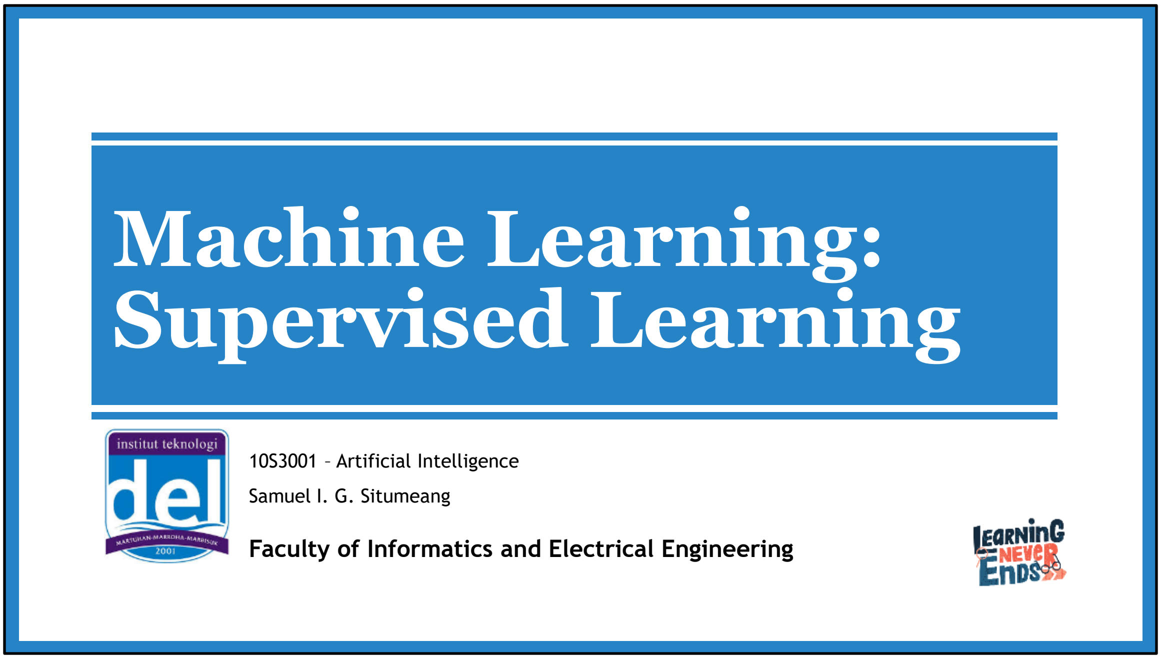


Machine Learning: Supervised Learning

10S3001 – Artificial Intelligence

Samuel I. G. Situmeang

Faculty of Informatics and Electrical Engineering

Objectives

Students are able:

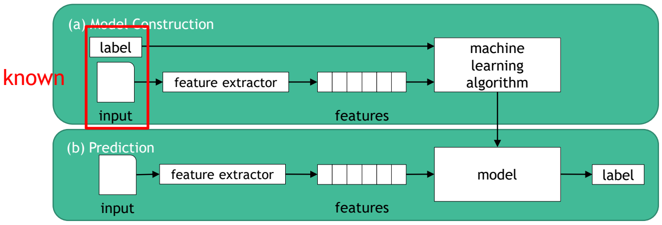
- to explain the concept of supervised learning, including its key components and applications.
- to define classification as a supervised learning task, and discuss common classification algorithms and their underlying principles.
- to describe the decision tree induction process, including algorithms like ID3 and C4.5, and their use in classification.
- to explain various metrics used to evaluate the performance of classification algorithms, such as accuracy, precision, recall, and F1-score.

3

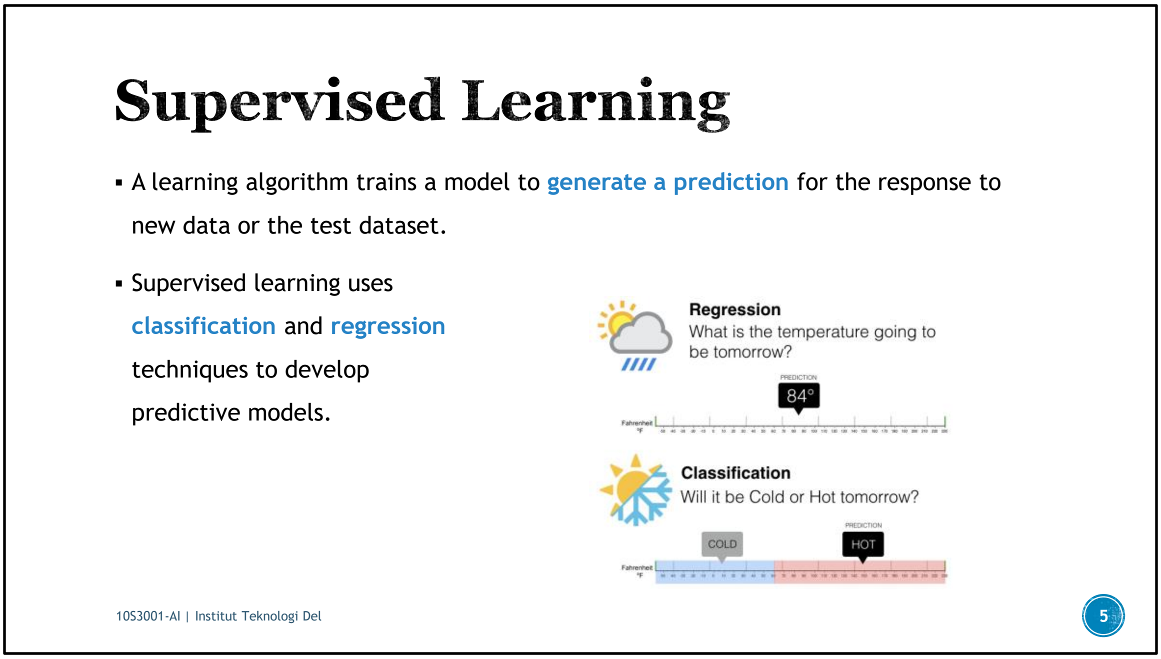
Supervised Learning

Supervised Learning

- Supervised learning is a learning model built to make a prediction, given an unforeseen input instance.



- A supervised learning algorithm takes a known set of input dataset and its known responses to the data (output) to learn a model.



Supervised Learning

- A learning algorithm trains a model to **generate a prediction** for the response to new data or the test dataset.
- Supervised learning uses **classification** and **regression** techniques to develop predictive models.

Regression
What is the temperature going to be tomorrow?

PREDICTION
84°
Fahrenheit °F

Classification
Will it be Cold or Hot tomorrow?

PREDICTION
COLD
HOT
Fahrenheit °F

Classification techniques predicts discrete responses. It is recommended if the data can be categorized, tagged, or separated into specific groups or classes. Classification models classify input data into categories. Popular or major applications of classification include bank credit scoring, medical imaging, and speech recognition. Also, handwriting recognition uses classification to recognize letters and numbers, to check whether an email is genuine or spam, or even to detect whether a tumor is benign or cancerous.

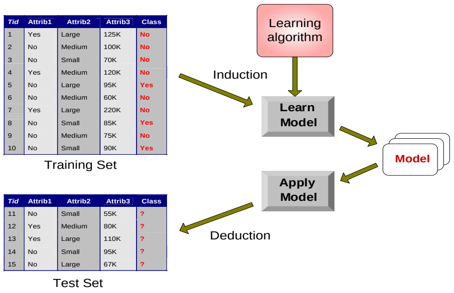
Regression techniques predict continuous responses. A linear regression attempts to model the relationship between two variables by fitting linear equation to observed data. For example, say, a data is collected about how happy people are after getting so many hours of sleep. In this dataset, sleep and happy people are the variables. By regression analysis, one can relate them and start making predictions.

6

Classification

Classification

- The **training data** such as observations or measurements are accompanied by labels indicating the classes which they belong to.
- New data is classified based on the models built from the training set.



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

Classification

- Some examples of classification problems are:
  - speech recognition,
  - handwriting recognition,
  - bio metric identification,
  - document classification,
  - classification schemata in biology,
  - diagnostic sections in illness encyclopedias,
  - online troubleshooting section on software web pages.

Types of Classification

- **Binary**

  item to be classified into one of two classes

  $h : D \rightarrow C, C = \{c_1, c_2\}$

  - e.g., Spam/not spam, male/female, rel/irrel

- **Single-Label Multi-Class (SLMC)**

  item to be classified into only one of n possible classes.

  $h : D \rightarrow C, C = \{c_1 \ldots c_n\}$, where $n > 2$

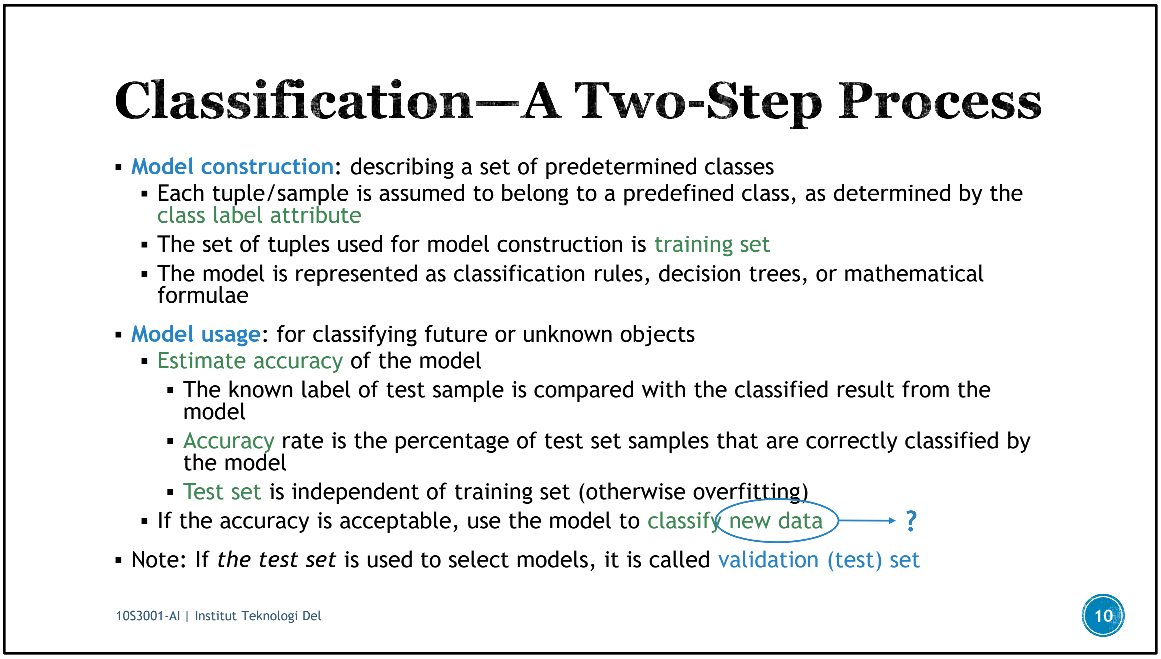
  - e.g., Sports/politics/entertainment, positive/negative/neutral

- **Multi-Label Multi-Class (MLMC)**

  item to be classified into one, two, or more classes

  $h : D \rightarrow 2^C, C = \{c_1 \ldots c_n\}$, where $n > 1$

  - e.g., Assigning CS articles to classes in the ACM Classification System
  - Usually be solved as n independent binary classification problems

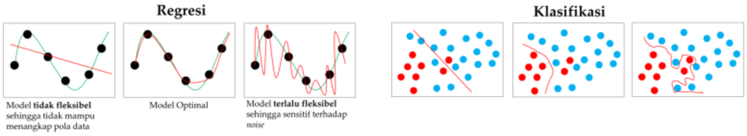


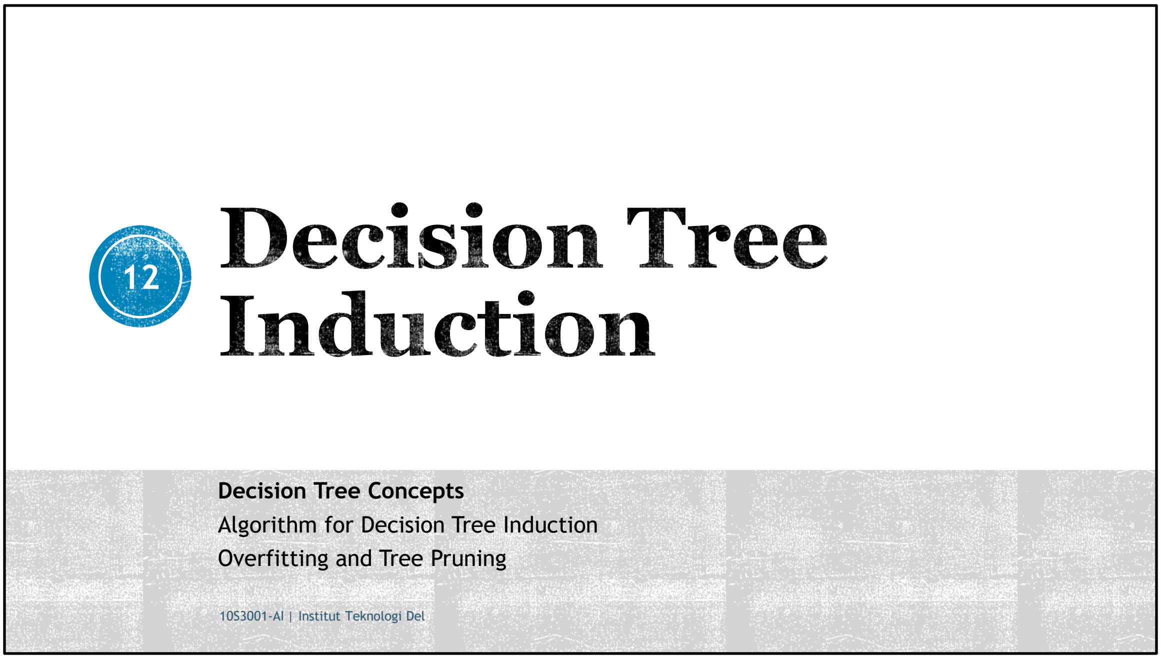
Classification—A Two-Step Process

- **Model construction**: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formulae
- **Model usage**: for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model
    - Test set is independent of training set (otherwise overfitting)
  - If the accuracy is acceptable, use the model to classify new data → ?
- Note: If *the test set* is used to select models, it is called validation (test) set

Overfitting and Underfitting

- **Overfitting**: keadaan ketika model memiliki kinerja baik hanya untuk *training data/seen examples* tetapi tidak memiliki kinerja baik untuk *unseen examples*.
  - Terjadi ketika model terlalu fleksibel (memiliki kemampuan yang terlalu tinggi untuk mengestimasi banyak fungsi) atau terlalu mencocokkan diri terhadap training data.
- **Underfitting**: keadaan ketika model memiliki kinerja buruk baik untuk *training data* dan *unseen examples*.
  - Terjadi akibat model yang telalu tidak fleksibel (memiliki kemampuan yang rendah untuk mengestimasi variasi fungsi.


Regresi
Klasifikasi
Model **tidak fleksibel** sehingga tidak mampu menangkap pola data
Model Optimal
Model **terlalu fleksibel** sehingga sensitif terhadap *noise*



12

Decision Tree Induction

Decision Tree Concepts

Algorithm for Decision Tree Induction

Overfitting and Tree Pruning

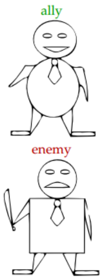
What is A Decision Tree

- Decision tree is a function that
  - takes a vector of attribute values as its input, and returns a “decision” as its output.
  - both input and output values can be measured on a nominal, ordinal, interval, and ratio scales, can be discrete or continuous.
- The decision is formed via a sequence of tests:
  - each **internal node** of the tree represents a test,
  - the **branches** are labeled with possible outcomes of the test, and
  - each **leaf node** represents a decision to be returned by the tree.

Attribute Description

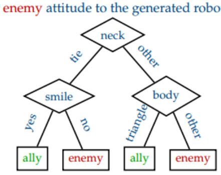
Example: A Computer Game

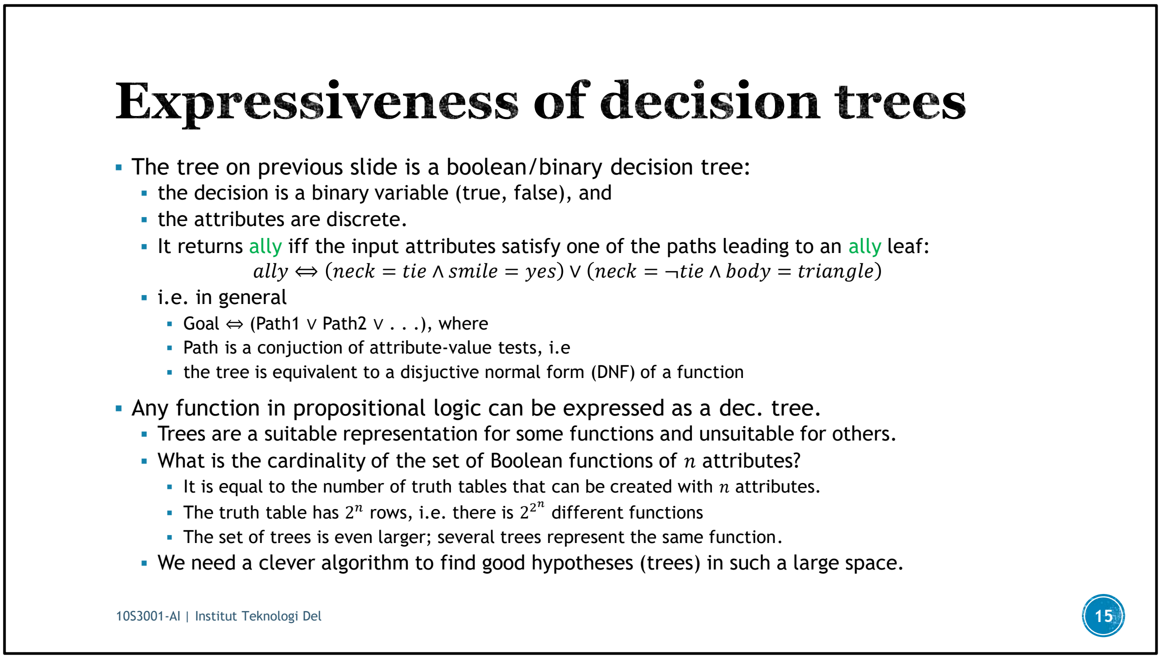
The main character of the game meets various robots along his way. Some behave like allies, others like enemies.



head	body	smile	neck	holds	class
circle	circle	yes	tie	nothing	ally
circle	square	no	tie	sword	enemy
…	…	…	…	…	…

The game engine may use e.g. the following tree to assign the ally or enemy attitude to the generated robots:





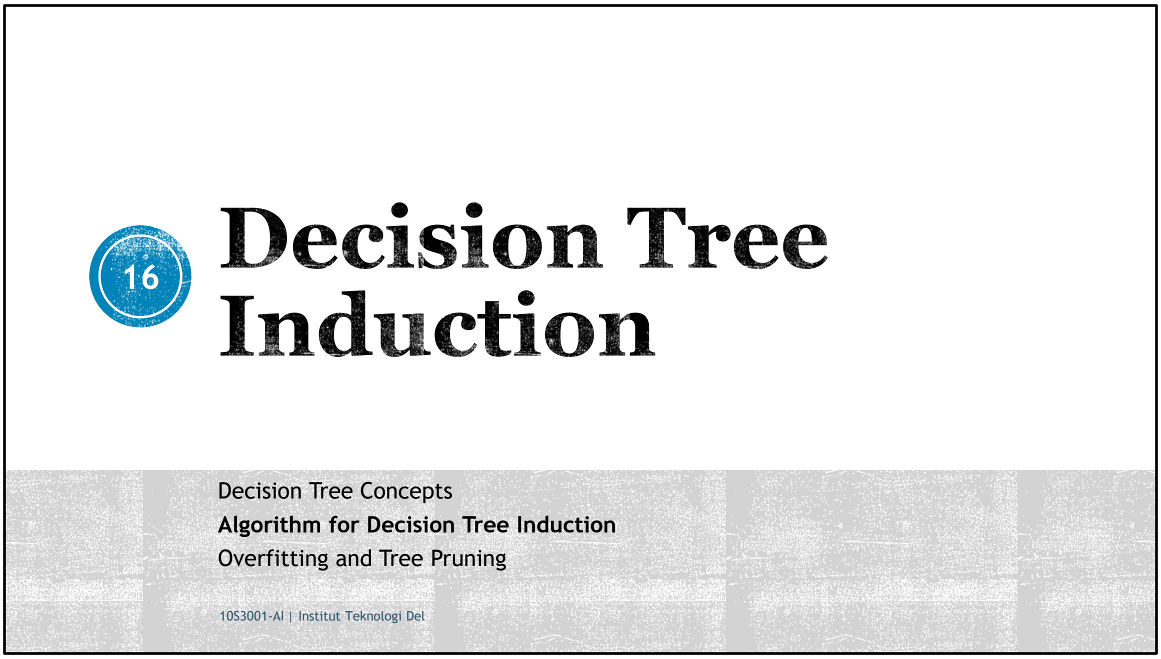
Expressiveness of decision trees

- The tree on previous slide is a boolean/binary decision tree:
  - the decision is a binary variable (true, false), and
  - the attributes are discrete.
  - It returns ally iff the input attributes satisfy one of the paths leading to an ally leaf:

$$ally \Leftrightarrow (neck = tie \wedge smile = yes) \vee (neck = \neg tie \wedge body = triangle)$$

  - i.e. in general
    - Goal ⇔ (Path1 ∨ Path2 ∨ . . .), where
    - Path is a conjuction of attribute-value tests, i.e
    - the tree is equivalent to a disjuctive normal form (DNF) of a function
- Any function in propositional logic can be expressed as a dec. tree.
  - Trees are a suitable representation for some functions and unsuitable for others.
  - What is the cardinality of the set of Boolean functions of n attributes?
    - It is equal to the number of truth tables that can be created with n attributes.
    - The truth table has 2^n rows, i.e. there is 2^{2^n} different functions
    - The set of trees is even larger; several trees represent the same function.
  - We need a clever algorithm to find good hypotheses (trees) in such a large space.

DNF - Disjunctive Normal Form: a standardization (or normalization) of a logical formula which is a disjunction of conjunctive clauses, it can also be described as an OR of ANDs.



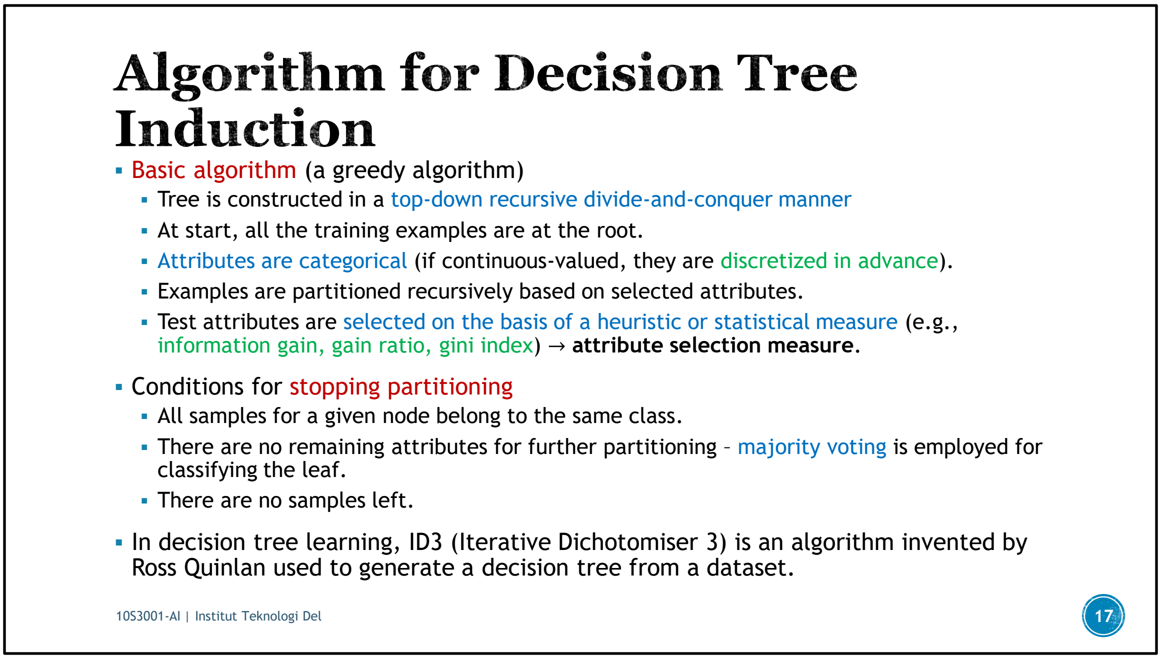
16

Decision Tree Induction

Decision Tree Concepts

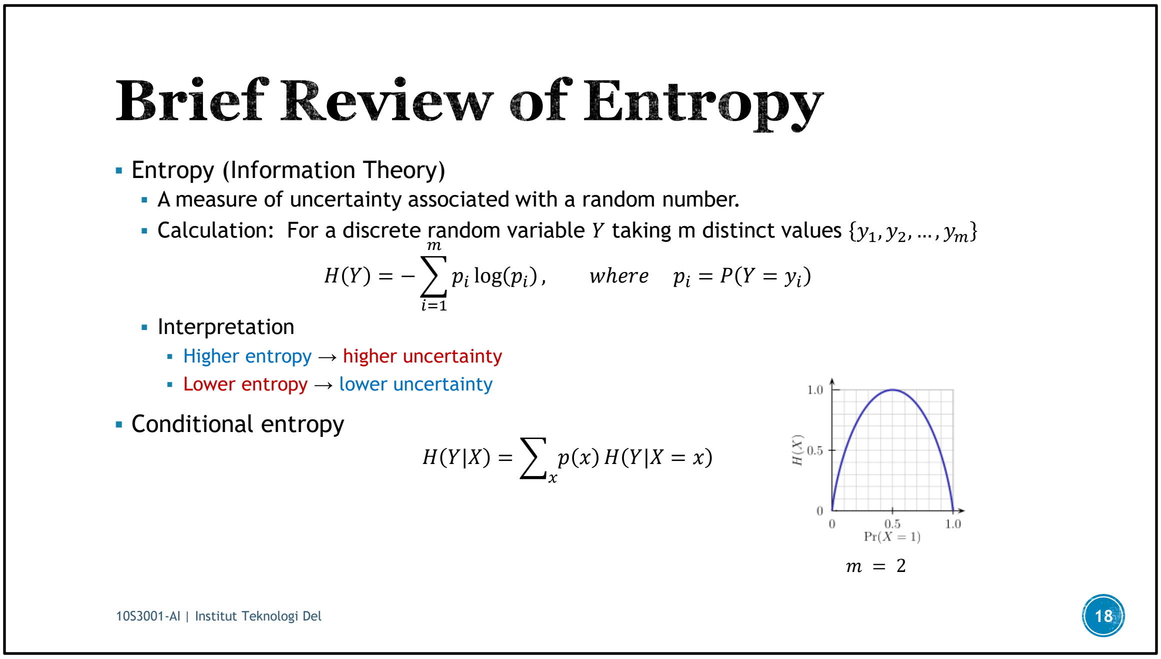
Algorithm for Decision Tree Induction

Overfitting and Tree Pruning



Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root.
  - Attributes are categorical (if continuous-valued, they are discretized in advance).
  - Examples are partitioned recursively based on selected attributes.
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain, gain ratio, gini index) → **attribute selection measure.**
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class.
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf.
  - There are no samples left.
- In decision tree learning, ID3 (Iterative Dichotomiser 3) is an algorithm invented by Ross Quinlan used to generate a decision tree from a dataset.



Brief Review of Entropy

- Entropy (Information Theory)
  - A measure of uncertainty associated with a random number.
  - Calculation: For a discrete random variable Y taking m distinct values $\{y_1, y_2, \dots, y_m\}$

$$H(Y) = -\sum_{i=1}^{m} p_i \log(p_i), \qquad where \quad p_i = P(Y = y_i)$$

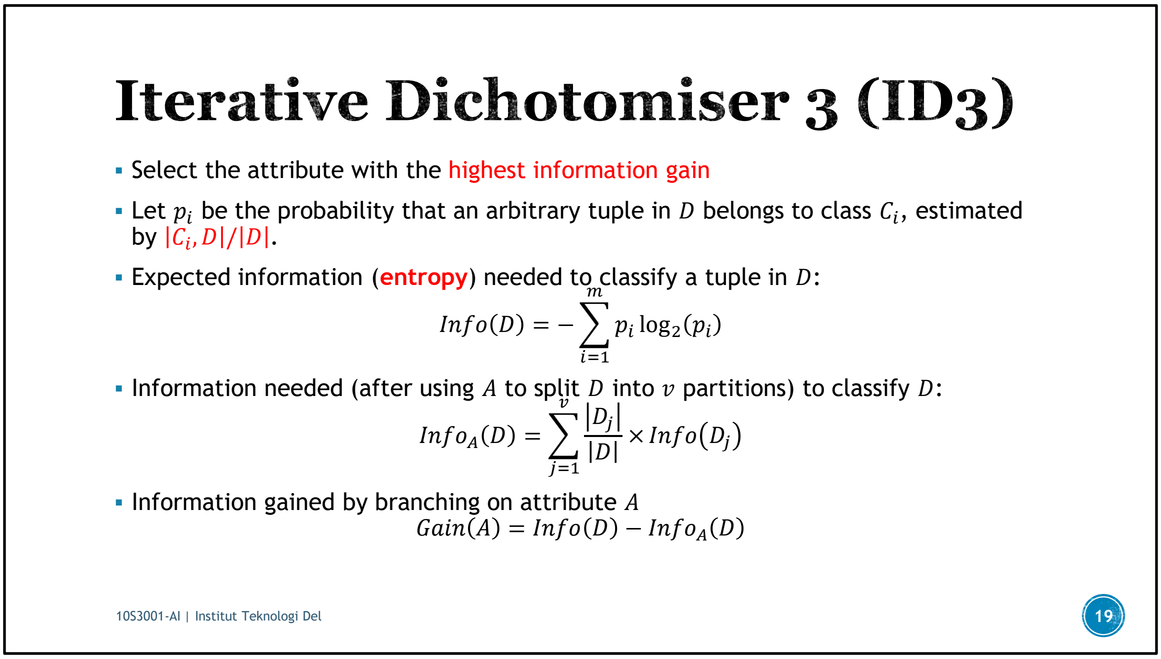
  - Interpretation
    - Higher entropy → higher uncertainty
    - Lower entropy → lower uncertainty
- Conditional entropy

$$H(Y|X) = \sum_{x} p(x)\, H(Y|X = x)$$

1.0
0.5
0
$H(X)$
0
0.5
1.0
$\Pr(X = 1)$

$m = 2$

- Entropy is a measure of *unpredictability* of the state, or equivalently, of its *average information content*. To get an intuitive understanding of these terms, consider the example of a political poll. Usually, such polls happen because the outcome of the poll is not already known. In other words, the outcome of the poll is relatively *unpredictable*, and actually performing the poll and learning the results gives some new *information*; these are just different ways of saying that the *a priori* entropy of the poll results is large. Now, consider the case that the same poll is performed a second time shortly after the first poll. Since the result of the first poll is already known, the outcome of the second poll can be predicted well and the results should not contain much new information; in this case the *a priori* entropy of the second poll result is small relative to that of the first.



Iterative Dichotomiser 3 (ID3)

- Select the attribute with the highest information gain
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,} D|/|D|$.
- Expected information (**entropy**) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gained by branching on attribute A

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

In **decision tree** learning, **ID3** (Iterative Dichotomiser 3) is an algorithm invented by Ross Quinlan used to generate a **decision tree** from a dataset.

ID3 Algorithm

1. Siapkan *training data*
2. Pilih salah satu atribut sebagai akar dengan **Information Gain**

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

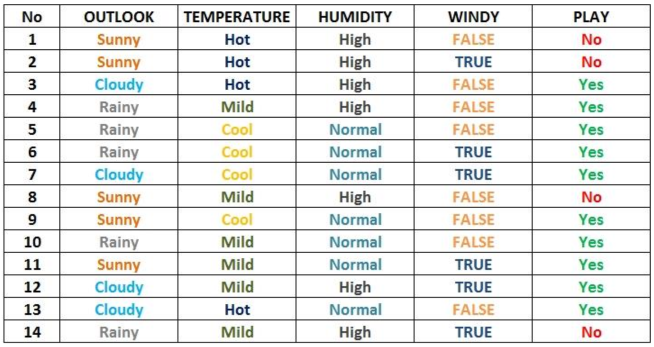
$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

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

3. Buat cabang untuk tiap-tiap nilai
4. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yg sama

Example of ID3 (1/19)

1. Siapkan *training data*

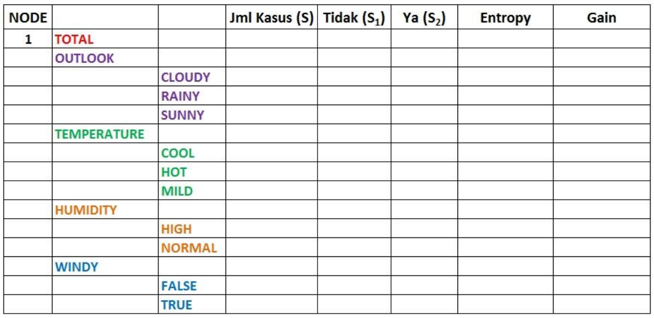


No	OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY
1	Sunny	Hot	High	FALSE	No
2	Sunny	Hot	High	TRUE	No
3	Cloudy	Hot	High	FALSE	Yes
4	Rainy	Mild	High	FALSE	Yes
5	Rainy	Cool	Normal	FALSE	Yes
6	Rainy	Cool	Normal	TRUE	Yes
7	Cloudy	Cool	Normal	TRUE	Yes
8	Sunny	Mild	High	FALSE	No
9	Sunny	Cool	Normal	FALSE	Yes
10	Rainy	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	High	TRUE	No

Example of ID3 (2/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**

Tips! Buat tabel untuk membantu proses perhitungan *Entropy* ($\mathrm{Info}(D)$) dan *Gain*



NODE			Jml Kasus (S)	Tidak (S_1)	Ya (S_2)	Entropy	Gain
1	TOTAL						
	OUTLOOK						
		CLOUDY					
		RAINY					
		SUNNY					
	TEMPERATURE						
		COOL					
		HOT					
		MILD					
	HUMIDITY						
		HIGH					
		NORMAL					
	WINDY						
		FALSE					
		TRUE					

Example of ID3 (3/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**
    - Hitung Entropy Total

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

$$Entropy(Total) = 0.863120569$$

    - Hitung Entropy (Outlook)

$$Entropy(Cloudy) = (-\frac{0}{4} * \log_2(\frac{0}{4})) + (-\frac{4}{4} * \log_2(\frac{4}{4})) \quad = 0.000000000$$

$$Entropy(Rainy) = (-\frac{1}{5} * \log_2(\frac{1}{5})) + (-\frac{4}{5} * \log_2(\frac{4}{5})) \quad = 0.721928095$$

$$Entropy(Sunny) = (-\frac{3}{5} * \log_2(\frac{3}{5})) + (-\frac{2}{5} * \log_2(\frac{2}{5})) \quad = 0.970950594$$

    - Hitung Entropy (Temperature)

$$Entropy(Cool) = (-\frac{0}{4} * \log_2(\frac{0}{4})) + (-\frac{4}{4} * \log_2(\frac{4}{4})) \quad = 0.000000000$$

$$Entropy(Hot) = (-\frac{2}{4} * \log_2(\frac{2}{4})) + (-\frac{2}{4} * \log_2(\frac{2}{4})) \quad = 1.000000000$$

$$Entropy(Mild) = (-\frac{2}{6} * \log_2(\frac{2}{6})) + (-\frac{4}{6} * \log_2(\frac{4}{6})) \quad = 0.918295834$$

Example of ID3 (4/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**
   - Hitung Entropy (Humidity)

$$Entropy(High) = (-\frac{4}{7} * \log_2(\frac{4}{7})) + (-\frac{3}{7} * \log_2(\frac{3}{7})) \ = 0.985228136$$

$$Entropy(Normal) = (-\frac{0}{7} * \log_2(\frac{0}{7})) + (-\frac{7}{7} * \log_2(\frac{7}{7})) \ = 0.000000000$$

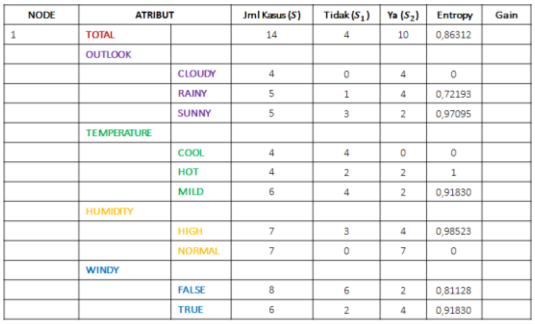
   - Hitung Entropy (Windy)

$$Entropy(False) = (-\frac{2}{8} * \log_2(\frac{2}{8})) + (-\frac{6}{8} * \log_2(\frac{6}{8})) \ = 0.811278124$$

$$Entropy(True) = (-\frac{4}{6} * \log_2(\frac{4}{6})) + (-\frac{2}{6} * \log_2(\frac{2}{6})) \ = 0.918295834$$

Example of ID3 (5/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**
   - Pindahkan hasil perhitungan ke dalam tabel.



NODE	ATRIBUT		Jml Kasus (S)	Tidak (S_1)	Ya (S_2)	Entropy	Gain
1	TOTAL		14	4	10	0,86312	
	OUTLOOK						
		CLOUDY	4	0	4	0	
		RAINY	5	1	4	0,72193	
		SUNNY	5	3	2	0,97095	
	TEMPERATURE						
		COOL	4	4	0	0	
		HOT	4	2	2	1	
		MILD	6	4	2	0,91830	
	HUMIDITY						
		HIGH	7	3	4	0,98523	
		NORMAL	7	0	7	0	
	WINDY						
		FALSE	8	6	2	0,81128	
		TRUE	6	2	4	0,91830	

Example of ID3 (6/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**
   - Hitung Gain.

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

$$Gain(Total, Outlook) = 0.863120569 - \left(\left(\frac{4}{14} * 0.000000000\right) + \left(\frac{5}{14} * 0.721928095\right) + \left(\frac{5}{14} * 0.970950594\right)\right)$$

$$Gain(Total, Outlook) = 0.258521037$$

$$Gain(Total, Temperature) = Entropy(Total) - \sum_{i=1}^{n} \frac{|Temperature_i|}{|Total|} * Entropy\ (Temperature_i)$$

$$Gain(Total, Temperature) = 0.863120569 - \left(\left(\frac{4}{14} * 0.000000000\right) + \left(\frac{4}{14} * 1.000000000\right) + \left(\frac{6}{14} * 0.918295834\right)\right)$$

$$Gain(Total, Temperature) = 0.183850925$$

$$Gain(Total, Humidity) = Entropy(Total) - \sum_{i=1}^{n} \frac{|Humidity_i|}{|Total|} * Entropy\ (Humidity_i)$$

$$Gain(Total, Humidity) = 0.863120569 - \left(\left(\frac{7}{14} * 0.985228136\right) + \left(\frac{7}{14} * 0.000000000\right)\right)$$

$$Gain(Total, Humidity) = 0.370506501$$

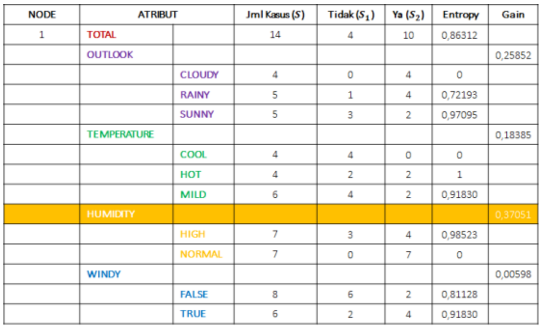
$$Gain(Total, Windy) = Entropy(Total) - \sum_{i=1}^{n} \frac{|Windy_i|}{|Total|} * Entropy\ (Windy_i)$$

$$Gain(Total, Windy) = 0.863120569 - \left(\left(\frac{8}{14} * 0.811278124\right) + \left(\frac{6}{14} * 0.918295834\right)\right)$$

$$Gain(Total, Windy) = 0.005977711$$

Example of ID3 (7/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**
   - Pindahkan hasil perhitungan ke dalam tabel.



NODE	ATRIBUT		Jml Kasus (S)	Tidak (S_1)	Ya (S_2)	Entropy	Gain
1	TOTAL		14	4	10	0,86312	
	OUTLOOK						0,25852
		CLOUDY	4	0	4	0	
		RAINY	5	1	4	0,72193	
		SUNNY	5	3	2	0,97095	
	TEMPERATURE						0,18385
		COOL	4	4	0	0	
		HOT	4	2	2	1	
		MILD	6	4	2	0,91830	
	HUMIDITY						0,37051
		HIGH	7	3	4	0,98523	
		NORMAL	7	0	7	0	
	WINDY						0,00598
		FALSE	8	6	2	0,81128	
		TRUE	6	2	4	0,91830	

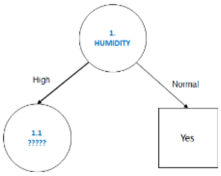
Gain terbesar

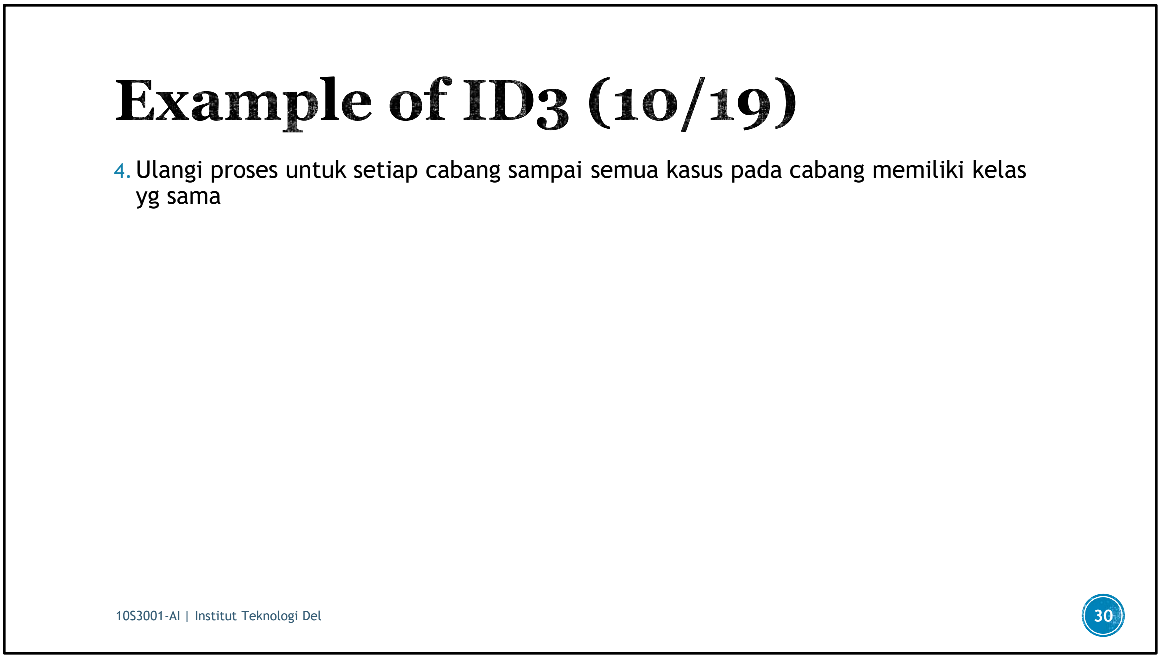
Example of ID3 (8/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**
   - Dari hasil pada Tabel Node 1, dapat diketahui bahwa atribut dengan Gain tertinggi adalah HUMIDITY yaitu sebesar 0.37051
     - Dengan demikian HUMIDITY dipilih menjadi node akar

Example of ID3 (9/19)

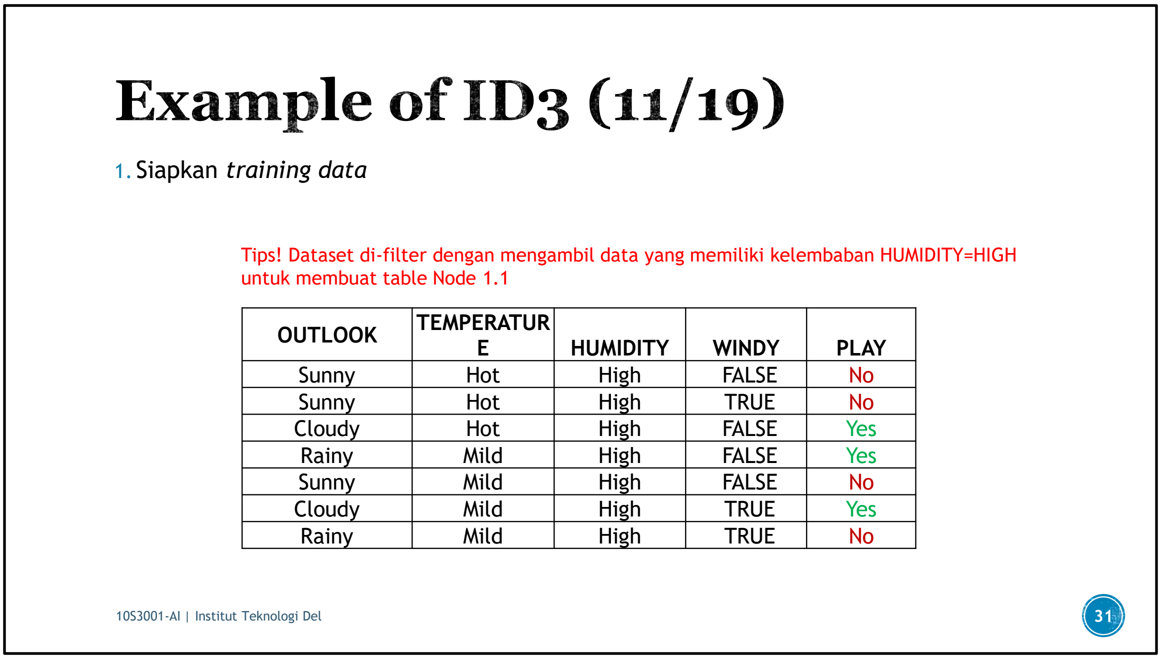
3. Buat cabang untuk tiap-tiap nilai
   - Ada 2 nilai atribut dari HUMIDITY yaitu HIGH dan NORMAL. Dari kedua nilai atribut tersebut, nilai atribut NORMAL sudah mengklasifikasikan kasus menjadi 1 yaitu keputusan-nya Yes, sehingga tidak perlu dilakukan perhitungan lebih lanjut
     - Tetapi untuk nilai atribut HIGH masih perlu dilakukan perhitungan lagi





Example of ID3 (10/19)

4. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yg sama



Example of ID3 (11/19)

1. Siapkan *training data*

Tips! Dataset di-filter dengan mengambil data yang memiliki kelembaban HUMIDITY=HIGH untuk membuat table Node 1.1

OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY
Sunny	Hot	High	FALSE	No
Sunny	Hot	High	TRUE	No
Cloudy	Hot	High	FALSE	Yes
Rainy	Mild	High	FALSE	Yes
Sunny	Mild	High	FALSE	No
Cloudy	Mild	High	TRUE	Yes
Rainy	Mild	High	TRUE	No

Example of ID3 (12/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**

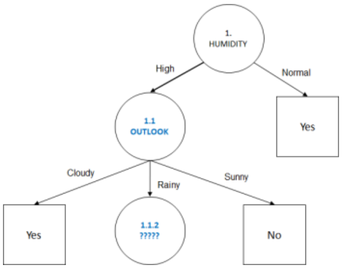
NODE	ATRIBUT		Jml Kasus (S)	Tidak (S_1)	Ya (S_2)	Entropy	Gain
1.1	HUMIDITY		7	4	3	0,98523	
	OUTLOOK						0,69951
		CLOUDY	2	0	2	0	
		RAINY	2	1	1	1	
		SUNNY	3	3	0	0	
	TEMPERATURE						0,02024
		COOL	0	0	0	0	
		HOT	3	2	1	0,91830	
		MILD	4	2	2	1	
	WINDY						0,02024
		FALSE	4	2	2	1	
		TRUE	3	2	1	0,91830	

Example of ID3 (13/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**
   - Dari hasil pada Tabel Node 1.1, dapat diketahui bahwa atribut dengan Gain tertinggi adalah OUTLOOK yaitu sebesar 0.69951
     - Dengan demikian OUTLOOK dapat menjadi node kedua

Example of ID3 (14/19)

3. Buat cabang untuk tiap-tiap nilai
   - Artibut CLOUDY = YES dan SUNNY= NO sudah mengklasifikasikan kasus menjadi 1 keputusan, sehingga tidak perlu dilakukan perhitungan lebih lanjut
     - Tetapi untuk nilai atribut RAINY masih perlu dilakukan perhitungan lagi



Example of ID3 (15/19)

4. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yg sama

Example of ID3 (16/19)

1. Siapkan *training data*

Tips! Dataset di-filter dengan mengambil data yang memiliki kelembaban HUMIDITY=HIGH untuk membuat table Node 1.1.2

OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY
Rainy	Mild	High	FALSE	Yes
Rainy	Mild	High	TRUE	No

Example of ID3 (17/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**

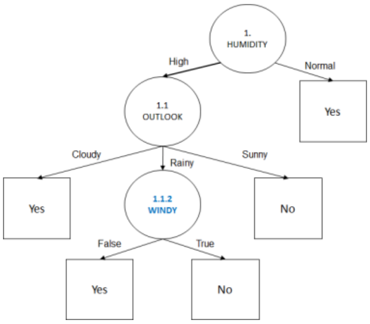
NODE	ATRIBUT		Jml Kasus (S)	Tidak (S_1)	Ya (S_2)	Entropy	Gain
1.1.2	HUMIDITY HIGH & OUTLOOK RAINY		2	1	1	1	
	TEMPERATURE						0
		COOL	0	0	0	0	
		HOT	0	0	0	0	
		MILD	2	1	1	1	
	WINDY						1
		FALSE	1	0	1	0	
		TRUE	1	1	0	0	

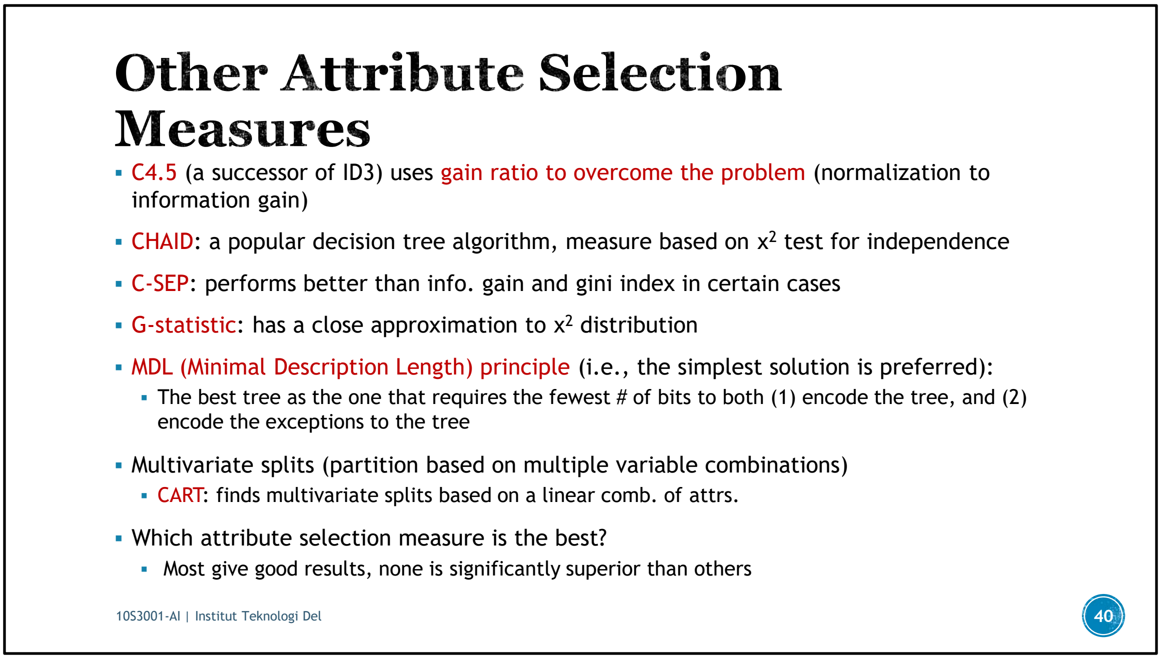
Example of ID3 (18/19)

2. Pilih salah satu atribut sebagai akar dengan **Information Gain**
   - Dari tabel, Gain Tertinggi adalah WINDY yaitu sebesar 1 dan menjadi node cabang dari atribut RAINY

Example of ID3 (19/19)

3. Buat cabang untuk tiap-tiap nilai
   - Karena semua kasus sudah masuk dalam kelas
   - Jadi, pohon keputusan pada Gambar merupakan pohon keputusan terakhir yang terbentuk

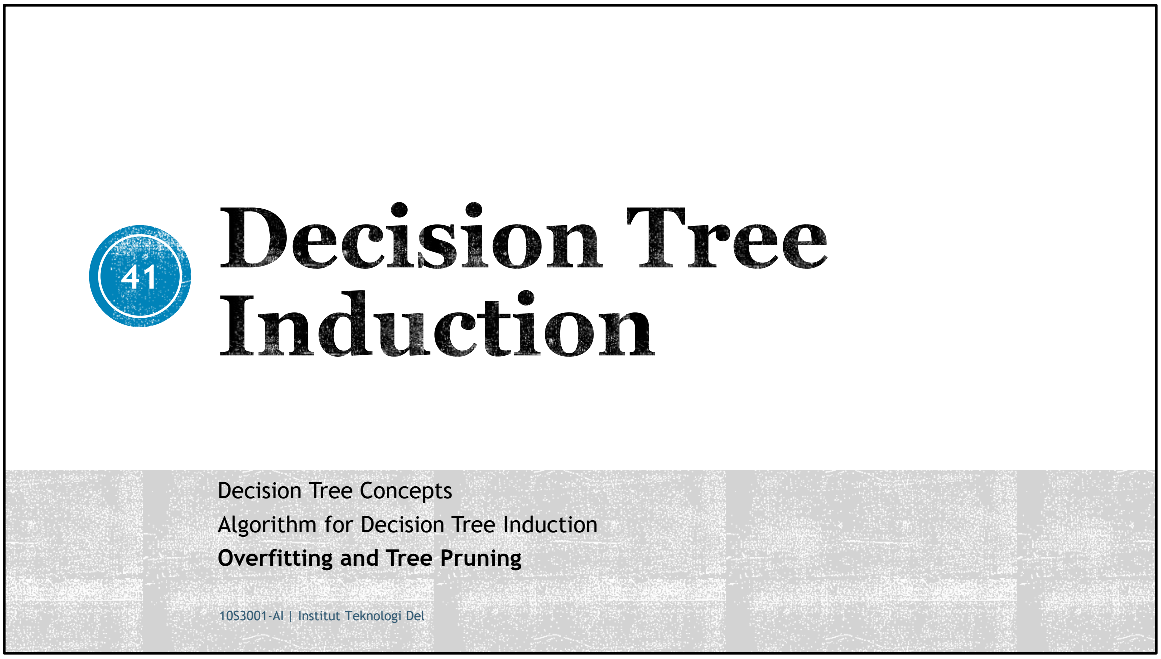




Other Attribute Selection Measures

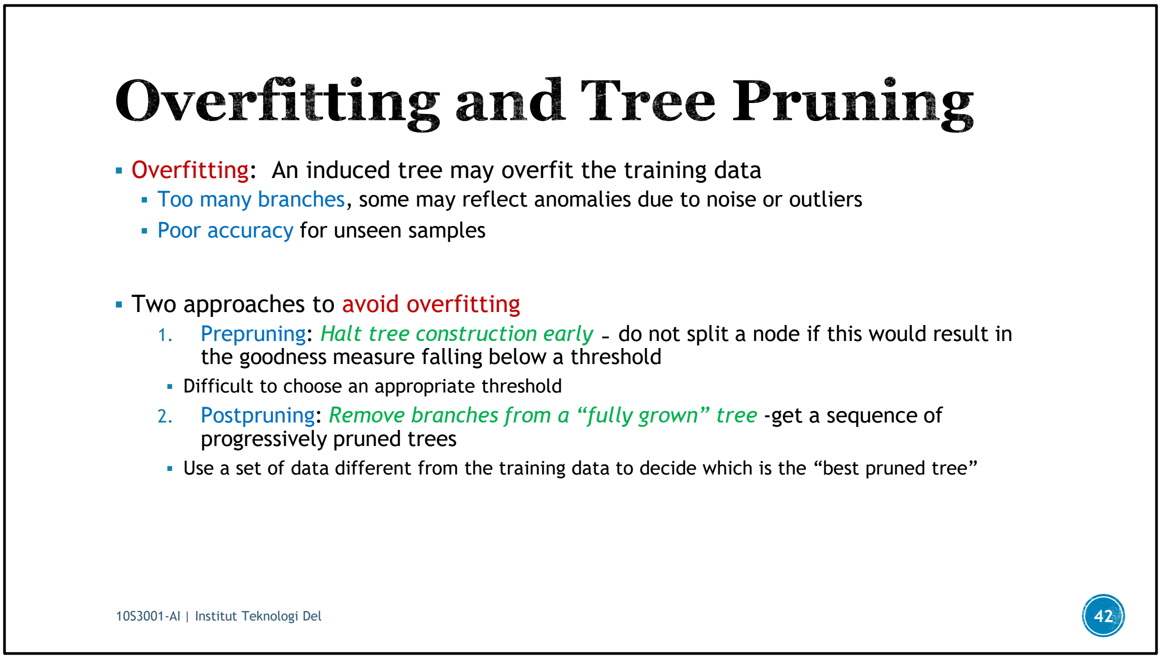
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)
- CHAID: a popular decision tree algorithm, measure based on x^2 test for independence
- C-SEP: performs better than info. gain and gini index in certain cases
- G-statistic: has a close approximation to x^2 distribution
- MDL (Minimal Description Length) principle (i.e., the simplest solution is preferred):
  - The best tree as the one that requires the fewest # of bits to both (1) encode the tree, and (2) encode the exceptions to the tree
- Multivariate splits (partition based on multiple variable combinations)
  - CART: finds multivariate splits based on a linear comb. of attrs.
- Which attribute selection measure is the best?
  - Most give good results, none is significantly superior than others

- CHAID (Chi-square Automatic Interaction Detector)
- C-SEP (Core Selective Evaluation Process)
- CART (Classification And Regression Tree)



Decision Tree Induction

Decision Tree Concepts
Algorithm for Decision Tree Induction
Overfitting and Tree Pruning

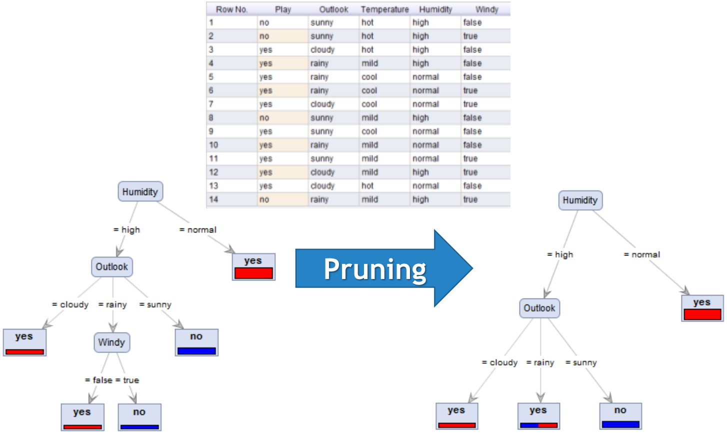


Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples

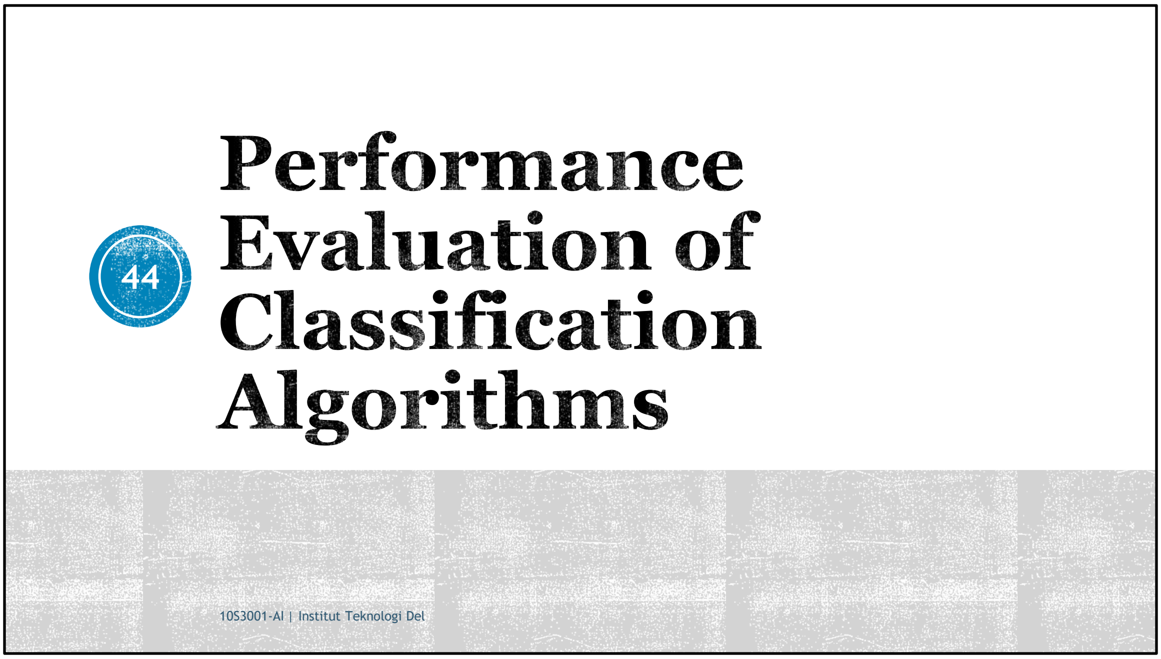
- Two approaches to avoid overfitting
  1. Prepruning: *Halt tree construction early* – do not split a node if this would result in the goodness measure falling below a threshold
     - Difficult to choose an appropriate threshold
  2. Postpruning: *Remove branches from a “fully grown” tree* -get a sequence of progressively pruned trees
     - Use a set of data different from the training data to decide which is the “best pruned tree”

Overfitting and Tree Pruning



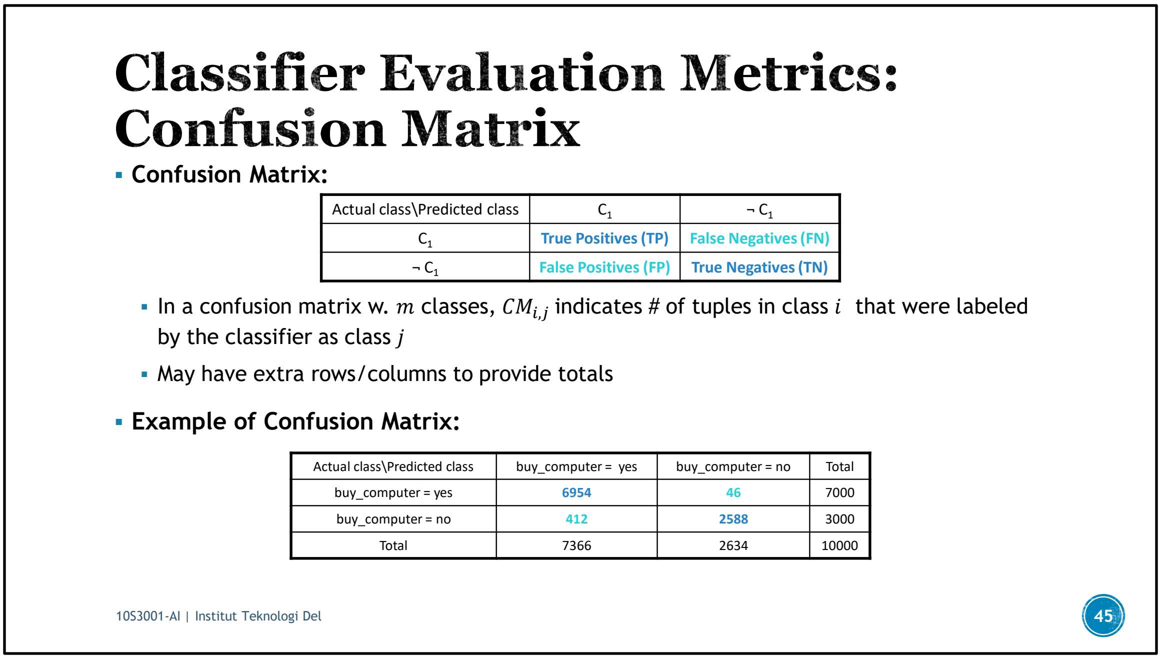
Row No.	Play	Outlook	Temperature	Humidity	Windy
1	no	sunny	hot	high	false
2	no	sunny	hot	high	true
3	yes	cloudy	hot	high	false
4	yes	rainy	mild	high	false
5	yes	rainy	cool	normal	false
6	yes	rainy	cool	normal	true
7	yes	cloudy	cool	normal	true
8	no	sunny	mild	high	false
9	yes	sunny	cool	normal	false
10	yes	rainy	mild	normal	false
11	yes	sunny	mild	normal	true
12	yes	cloudy	mild	high	true
13	yes	cloudy	hot	normal	false
14	no	rainy	mild	high	true

Pruning



44

Performance Evaluation of Classification Algorithms



Classifier Evaluation Metrics: Confusion Matrix

- **Confusion Matrix:**

Actual class\Predicted class	C_1	¬ C_1
C_1	**True Positives (TP)**	**False Negatives (FN)**
¬ C_1	**False Positives (FP)**	**True Negatives (TN)**

  - In a confusion matrix w. m classes, $CM_{i,j}$ indicates # of tuples in class i that were labeled by the classifier as class j
  - May have extra rows/columns to provide totals

- **Example of Confusion Matrix:**

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

A confusion matrix is a specific table layout that visualize the performance summary of a classification algorithm.

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P’	**N’**	**All**

- Classifier Accuracy or recognition rate: percentage of test set tuples that are correctly classified
  Accuracy = (TP + TN)/All

- Error rate: 1 – accuracy, or
  Error rate = (FP + FN)/All

Class Imbalance Problem:

- **One class may be rare**, e.g. fraud, or HIV-positive
- Significant **majority of the negative class** and minority of the positive class

- Sensitivity: True Positive recognition rate
  - **Sensitivity** = TP/P
- Specificity: True Negative recognition rate
  - **Specificity** = TN/N

Classifier Evaluation Metrics: Precision and Recall, and F-measures

- **Precision**: exactness – what % of tuples that the classifier labeled as positive are actually positive?

$$\mathrm{P} = \mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP}\ + \mathrm{FP}}$$

- **Recall**: completeness – what % of positive tuples did the classifier label as positive?

$$\mathrm{R} = \mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP}\ + \mathrm{FN}}$$

- Perfect score is 1.0
- Inverse relationship between precision & recall
- **F-measure (F-score)**: harmonic mean of precision and recall
  - In general, it is the weighted measure of precision & recall

$$\mathrm{F}_{\beta} = \frac{1}{\alpha \cdot \frac{1}{\mathrm{P}} + (1 - \alpha) \cdot \frac{1}{\mathrm{R}}} = \frac{(\beta^2 + 1)\mathrm{PR}}{\beta^2\mathrm{P} + \mathrm{R}}$$

Assigning β times as much weight to recall as to precision)

  - ***F1-measure (balanced F-measure)***
    - That is, when β = 1,

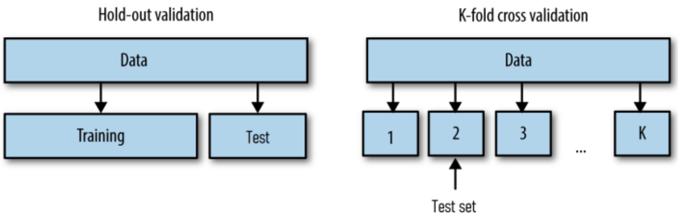
$$\mathrm{F}_1 = \frac{2\mathrm{PR}}{\mathrm{P} + \mathrm{R}}$$

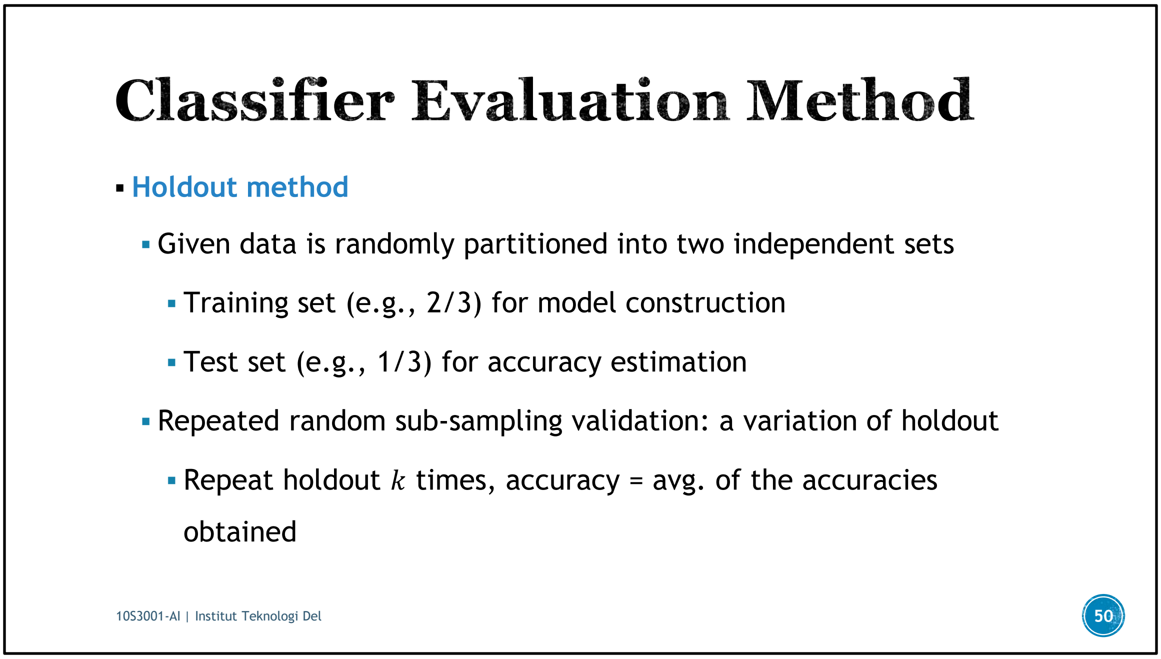
Evaluation: Multi-class

- Micro-average of Precision= $\frac{\sum_{i=1}^{N} \mathrm{TP}_i}{\sum_{i=1}^{N} \mathrm{TP}_i+\sum_{i=1}^{N} \mathrm{FP}_i}$
- Micro-average of Recall= $\frac{\sum_{i=1}^{N} \mathrm{TP}_i}{\sum_{i=1}^{N} \mathrm{TP}_i+\sum_{i=1}^{N} \mathrm{FN}_i}$
- Macro-average of Precision= $\frac{\sum_{i=1}^{N} \mathrm{P}_i}{N}$
- Macro-average of Recall= $\frac{\sum_{i=1}^{N} \mathrm{R}_i}{N}$

Classifier Evaluation Method

- Two of the most popular strategies to perform the evaluation step are the **hold-out** method and the **k-fold cross-validation** method.



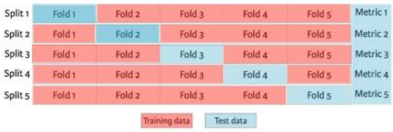


Classifier Evaluation Method

- **Holdout method**
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Repeated random sub-sampling validation: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained

Classifier Evaluation Method

- K-Fold Cross-Validation (k-fold, where $k = 10$ is most popular)
  - Randomly partition the data into k *mutually exclusive* subsets, each approximately equal size
  - At i-th iteration, use D_i as test set and others as training set
  - Leave-one-out: k folds where k = # of tuples, for small sized data



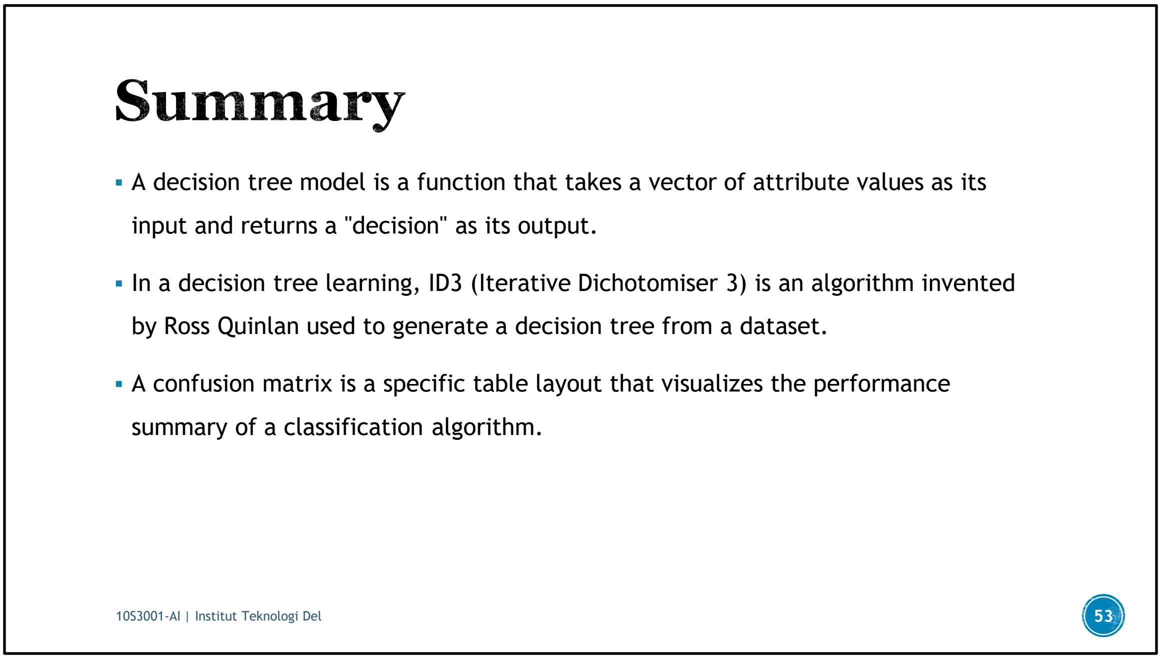
Split 1	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Metric 1
Split 2	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Metric 2
Split 3	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Metric 3
Split 4	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Metric 4
Split 5	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Metric 5

Source: www.datacamp.com

  - ***Stratified cross-validation***: folds are stratified so that class distribution, in each fold is approximately the same as that in the initial data

Summary

- Supervised learning is a learning model built to make a prediction, given an unforeseen input instance.
- Supervised learning uses classification and regression techniques to develop predictive models.
- Classification techniques predict discrete responses.
- There are three types of classification, which are binary, single-label multi-class (SLMC), and multi-label multi-class (MLMC).
- Classification can be divided into a two-step process, which is model construction and model usage.



Summary

- A decision tree model is a function that takes a vector of attribute values as its input and returns a "decision" as its output.
- In a decision tree learning, ID3 (Iterative Dichotomiser 3) is an algorithm invented by Ross Quinlan used to generate a decision tree from a dataset.
- A confusion matrix is a specific table layout that visualizes the performance summary of a classification algorithm.

Summary

- Based on the confusion matrix, we can calculate accuracy, error rate, sensitivity, specificity, precision, recall, and f-measure.
- For multi-class classification performance evaluation, we can use micro-averaging or macro-averaging of a specific evaluation metric.
- Two of the most popular strategies to perform the evaluation step are the hold-out method and the k-fold cross-validation method.

References

- S. J. Russell and P. Borvig, *Artificial Intelligence: A Modern Approach (4th Edition)*, Prentice Hall International, 2020.
  - Chapter 19. Learning from Examples
- J. Han and M. Kamber, “*Data Mining: Concepts and Techniques (3rd Edition)*,” Elsevier, 2012.

eof