CS 412 Intro. to Data Mining

Chapter 8. Classification: Basic Concepts

Jiawei Han, Computer Science, Univ. Illinois at Urbana-Champaign, 2017

Supervised vs. Unsupervised Learning (1)

การสร้างโมเดล ทขขร์ ฝึกสอน

สร้างโมเดลแล้วไปทดสอบ

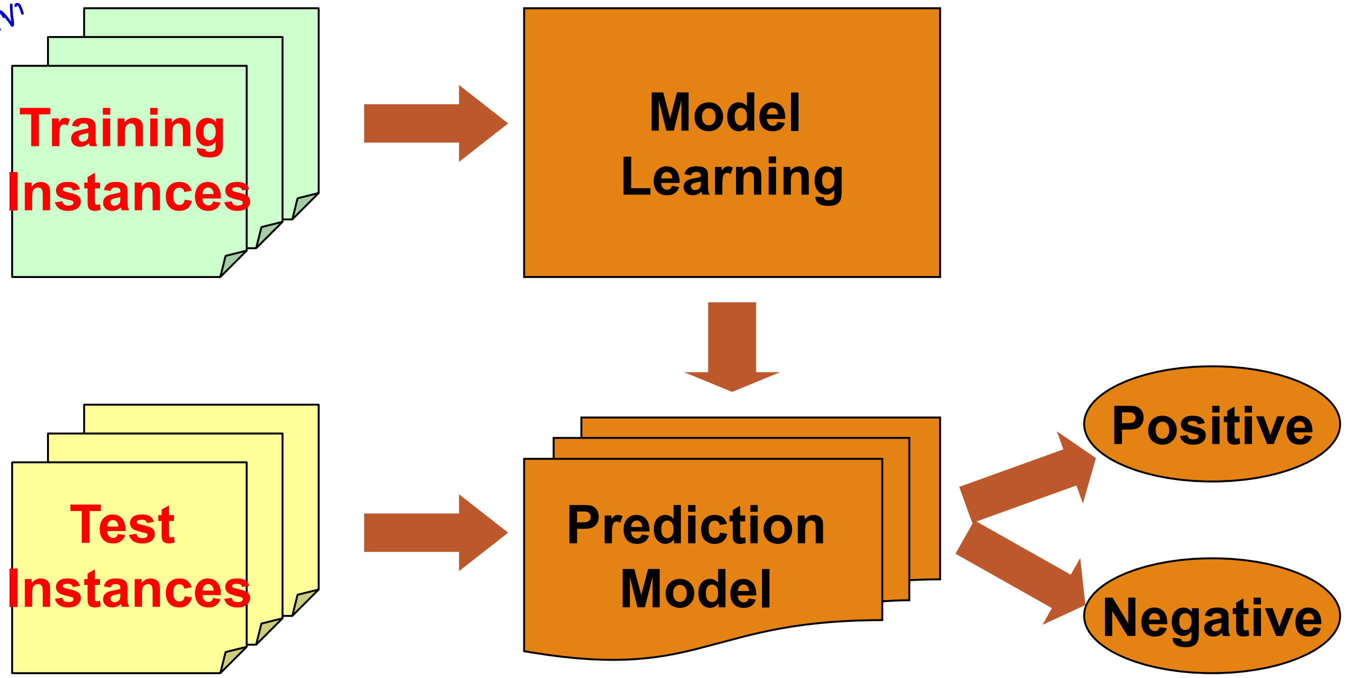
- Supervised learning (classification)
  - Supervision: The training data such as observations or measurements are accompanied by **labels** indicating the classes which they belong to
  - New data is classified based on the models built from the training set

Training Data with class label:

ด.น่าจะ.เป็น

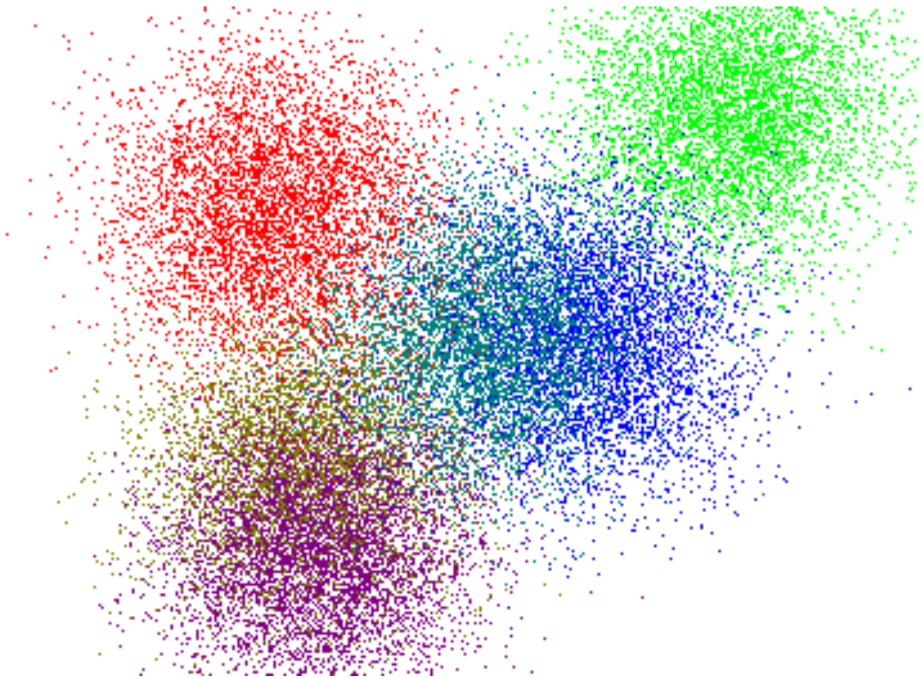
การตัดสินใจของเขา

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no



Supervised vs. Unsupervised Learning (2)

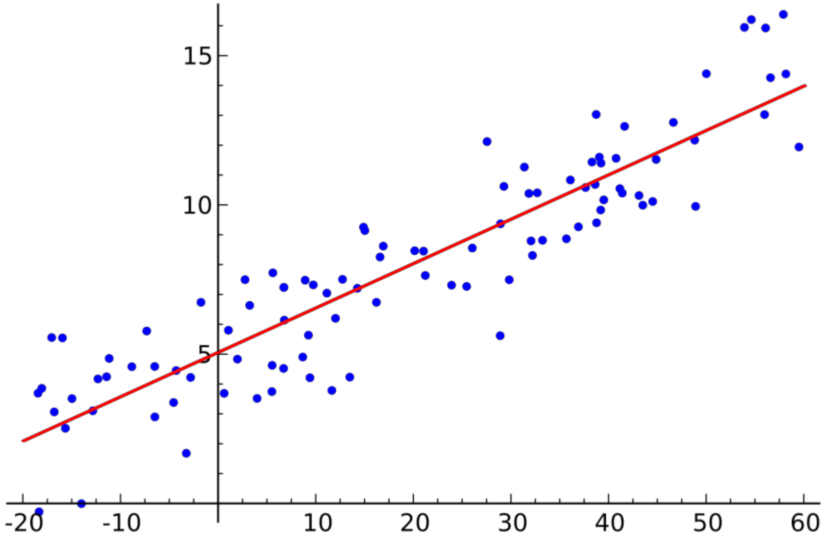
- Unsupervised learning (clustering) ไม่มีผู้สอนก็คือไม่มีจุดมุ่งหมายในการเรียน
  - The class labels of training data are unknown
  - Given a set of observations or measurements, establish the possible existence of classes or clusters in the data



Prediction Problems: Classification vs. Numeric Prediction

ถ้าทำนายออกมาเป็นตัวเลข
จะเรียกว่า Regretion

- **Classification**
  - Predict categorical class labels (discrete or nominal)
  - Construct a model based on the training set and the **class labels** (the values in a classifying attribute) and use it in classifying new data
- **Numeric prediction**
  - Model continuous-valued functions (i.e., predict unknown or missing values)
- Typical applications of classification
  - Credit/loan approval
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is



Classification—Model Construction, Validation and Testing

- **Model construction** เอา data ที่สร้างฟีเจอร์ และเอาคำตอบมาแทนในโมเดลมาสร้างโมเดลโดยอาศัยฟีเจอร์นี้
  - Each sample is assumed to belong to a predefined class (shown by the **class label**)
  - The set of samples used for model construction is **training set**
  - Model: Represented as decision trees, rules, mathematical formulas, or other forms
- **Model Validation and Testing**: เอาโมเดลไปวัดผล
  - **Test:** Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - *Accuracy:* % of test set samples that are correctly classified by the model
    - Test set is independent of training set
  - **Validation**: If *the test set* is used to select or refine models, it is called **validation** (or development) **(test) set**
- **Model Deployment:** If the accuracy is acceptable, use the model to classify new data

Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Linear Classifier
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Additional Concepts on Classification
- Summary

Information Gain: An Attribute Selection Measure

- Select the attribute with the highest information gain (used in typical decision tree induction algorithm: ID3/C4.5)
- 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:

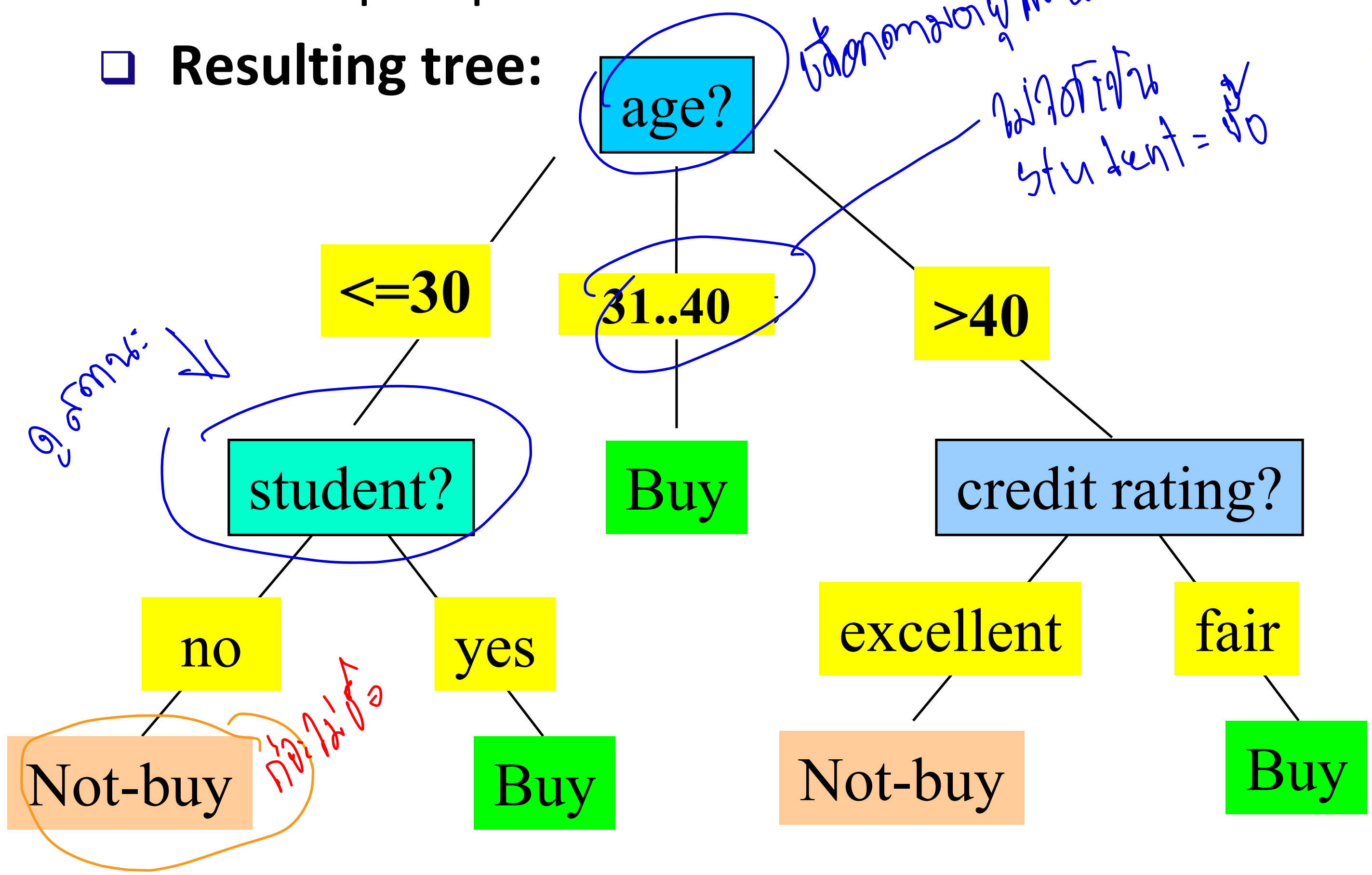
$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gained by branching on attribute A

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

Decision Tree Induction: An Example

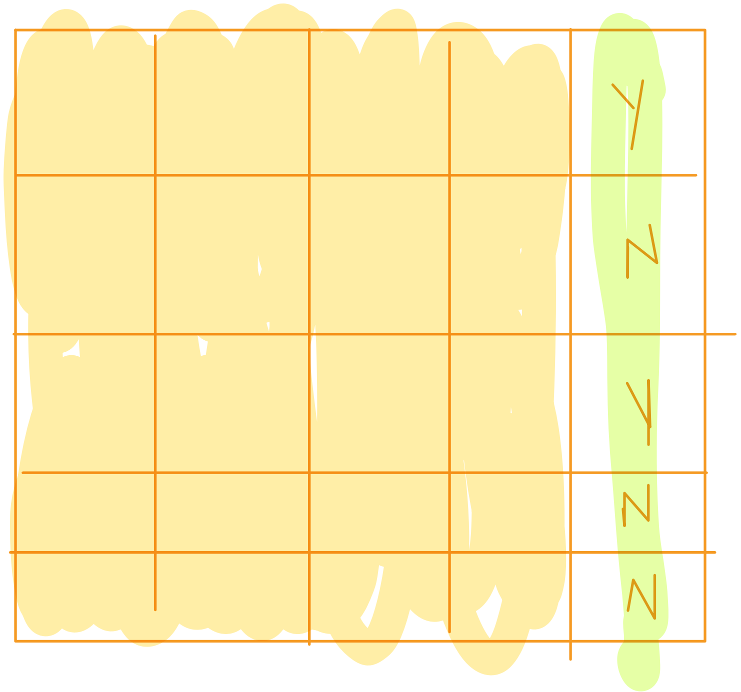
- **Decision tree construction**:
  - A top-down, recursive, divide-and-conquer process
- **Resulting tree:**

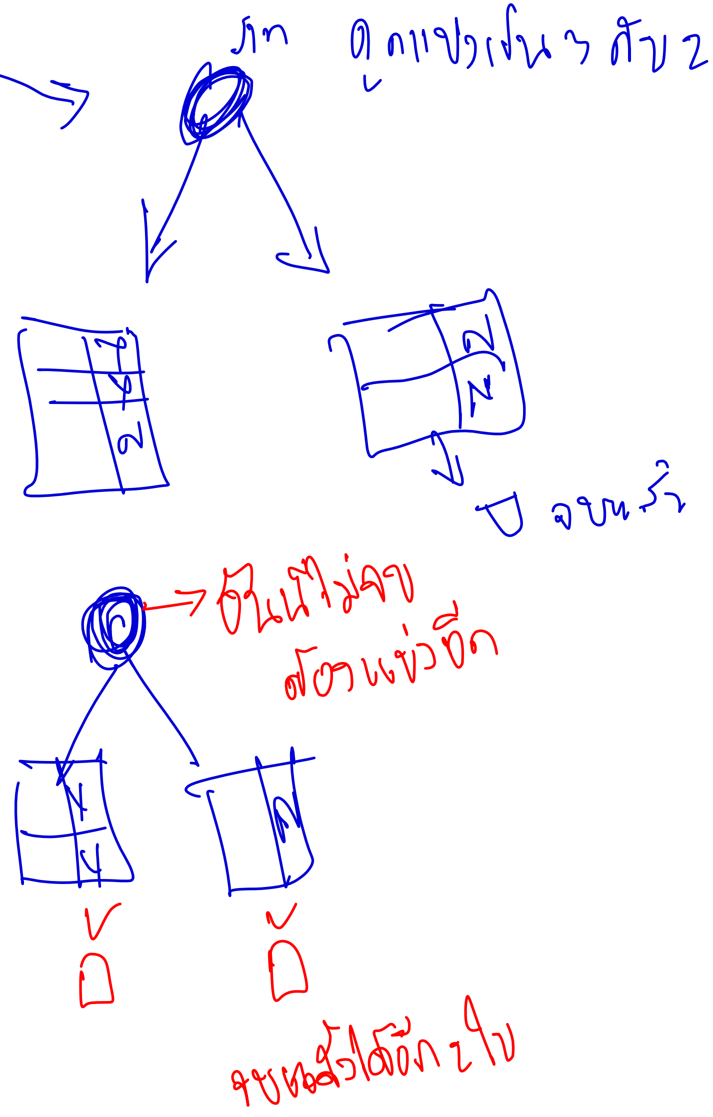


Training data set: Who buys computer?

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	~~no~~
<=30	high	no	excellent	~~no~~
31…40	high	no	fair	~~yes~~
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Note: The data set is adapted from "Playing Tennis" example of R. Quinlan

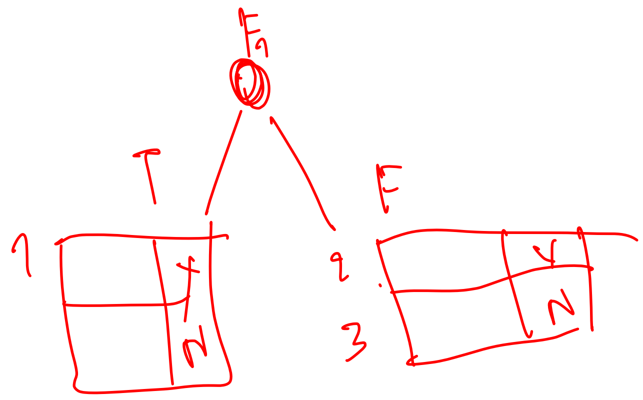


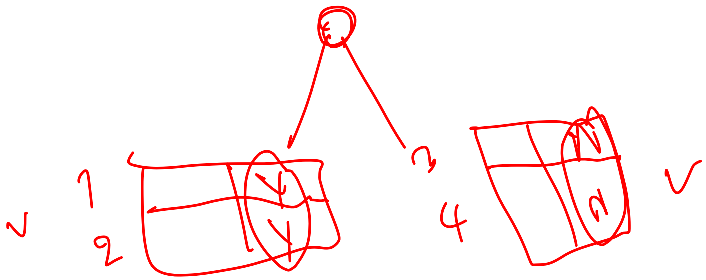


คำตอบสุดท้าย

ดูข้อมูลส่วนนี้ในการตอบคำถาม

F_1	F_2	F_3	Y
T	T	F	N
F	T	F	Y
F	F	F	N
T	F	F	N





ตอบ True ก็ yes

ตอบ fals ก็ no

Example: Attribute Selection with Information Gain

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

14 age

$$Info_{age}(D) = \overset{\le 30 \;\; Y\; N}{\frac{5}{14} I(2,3)} + \overset{31-40 \;\; Y\; N}{\frac{4}{14} I(4,0)} + \underset{>40}{\frac{5}{14} I(3,2)} = 0.694$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

$\frac{5}{14} I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's.

Hence

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Similarly, we can get

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

Decision Tree Induction: Algorithm

- Basic algorithm
  - แบ่ง data ให้เล็กลงเรื่อยๆ
  - Tree is constructed in a **top-down, recursive, divide-and-conquer manner**
  - At start, all the training examples are at the root
  - data แบ่งเหมือนเดิมไปเรื่อยๆ
  - Examples are partitioned recursively based on selected attributes
  - On each node, attributes are selected based on the training examples on that node, and a heuristic or statistical measure (e.g., **information gain**)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning
  - There are no samples left
- Prediction
  - **Majority voting** is employed for classifying the leaf

How to Handle Continuous-Valued Attributes?

วิธีที่ 1 แบ่งกลุ่ม

- Method 1: Discretize continuous values and treat them as categorical values
  - E.g., age: < 20, 20..30, 30..40, 40..50, > 50
- Method 2: Determine the ***best split point*** for continuous-valued attribute A
  - Sort the value A in increasing order:, e.g. 15, 18, 21, 22, 24, 25, 29, 31, …
  - *Possible split point:* the midpoint between *each pair of adjacent values*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
    - e.g., (15+18/2 = 16.5, 19.5, 21.5, 23, 24.5, 27, 30, …
  - The point with the *maximum information gain* for A is selected as the **split-point** for A
- Split: Based on split point P
  - The set of tuples in D satisfying A ≤ P vs. those with A > P

Math 1 Categorize

15, 18, 21, 22, 24, 25, 29, 31, ...

< 18, 18 - 22, 22 - 30, > 31

Math 2 Best spitpoint

เรียง data 15, 18, 21, 22, 24, 25, 29, 31, ...

Y Y N Y Y N N N N

≤ 21.5 age > 21.5

O O O
Y Y N

O O O O O
Y N N N N

≤ 24.5 > 24.5

O O O O O O O O

$\frac{5}{8} I(3,2) + \frac{3}{8} I(0,3)$

age 21.5 $= \frac{3}{8} I(2,1) + \frac{5}{8} I(1,4)$

Gain Ratio: A Refined Measure for Attribute Selection

- Information gain measure is biased towards attributes with a large number of values
- Gain ratio: Overcomes the problem (as a normalization to information gain)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

  - GainRatio(A) = Gain(A)/SplitInfo(A)
  - The attribute with the maximum gain ratio is selected as the splitting attribute
  - Gain ratio is used in a popular algorithm C4.5 (a successor of ID3) by R. Quinlan
- Example
  - $\mathrm{SplitInfo}_{\mathrm{income}}(\mathrm{D}) = -\frac{4}{14}\log_2\frac{4}{14} - \frac{6}{14}\log_2\frac{6}{14} - \frac{4}{14}\log_2\frac{4}{14} = 1.557$
  - GainRatio(income) = 0.029/1.557 = 0.019

Another Measure: Gini Index

- Gini index: Used in CART, and also in IBM IntelligentMiner
- If a data set D contains examples from n classes, gini index, $gini(D)$ is defined as
  - $gini(D) = 1 - \sum_{j=1}^{n} p_j^2$ $\quad \Sigma p \log p \rightarrow (-p \log p) + (-p \log p)$
    - p_j is the relative frequency of class j in D
- If a data set D is split on A into two subsets D_1 and D_2, the $gini$ index $gini(D)$ is defined as

  $9(4,5)$ data [illegible]

  $8(3,5)$

  - $gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$
- Reduction in Impurity:
  - $\Delta gini(A) = gini(D) - gini_A(D)$

  อันเดิม $-\frac{3}{8}\log\frac{3}{8} - \frac{5}{8}\log\frac{5}{8}$

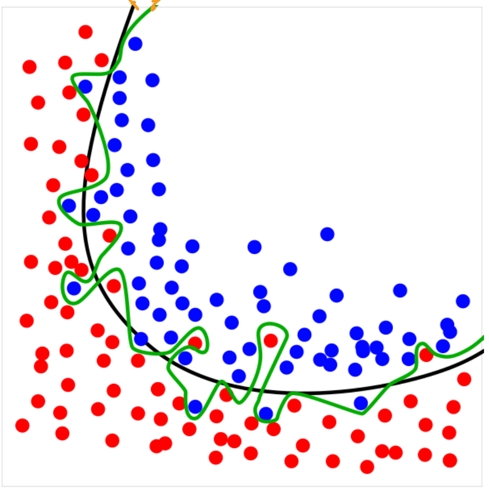
  อันใหม่ $1-\{(\frac{3}{8})^2+(\frac{5}{8})^2\}$

- The attribute provides the smallest $gini_{split}(D)$ (or the largest reduction in impurity) is chosen to split the node (*need to enumerate all the possible splitting points for each attribute*)

การตัดแต่งกิ่ง

Overfitting and Tree Pruning

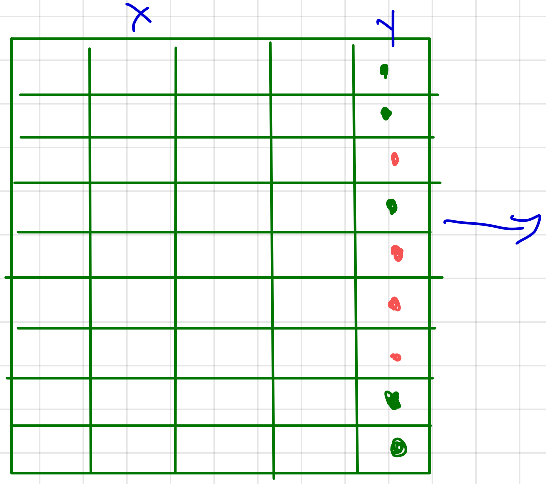
#2 #6

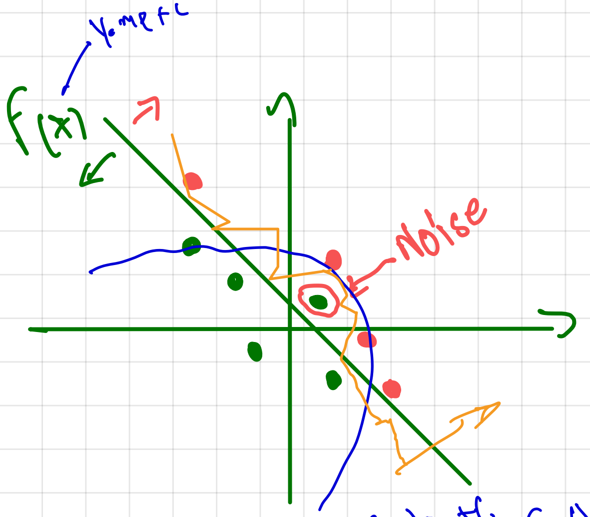


- **Overfitting**: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Prepruning: *Halt tree construction early*-do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - 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"

หยุดการสร้างต้นไม้ก่อน

ตัดกิ่งออกหลังจากสร้างต้นไม้เสร็จแล้ว





$$ax^2 + bx + c$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Occam's Razor

Classifier Evaluation Metrics: Confusion Matrix

- **Confusion Matrix:**

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

Recall

- 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:** Test

Positives Positives Negatives

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

neg

	P	N
P	TP 0	FN 2
N	FP 0	TN 98

$\frac{0}{0+0}$

$\frac{0}{0+2} = 0$

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

    ความแม่นยำ

    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
  - Measures handle the class imbalance problem
    - **Sensitivity** (recall): True positive recognition rate

      positive ทั้งหมด

      - **Sensitivity = TP/P**
    - **Specificity**: True negative recognition rate
      - **Specificity = TN/N** [illegible] ทั้งหมด

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

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

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

→ ตัวที่ model ทายว่าเป็น Pos ถูกต้องมากน้อยแค่ไหน

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

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

โมเดลเราหา ⇒ ตัวที่เป็น Pos จริงๆ เจอมากน้อยแค่ไหน

  - Range: [0, 1]
- The "inverse" relationship between precision & recall
- ***F* measure (or *F*-score):** harmonic mean of precision and recall
  - In general, it is the weighted measure of precision & recall

$$\boxed{F_\beta = \frac{1}{\alpha \cdot \frac{1}{P} + (1 - \alpha) \cdot \frac{1}{R}} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}}$$

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

  - ***F1-measure (balanced F-measure)***
    - That is, when β = 1, $F_1 = \frac{2PR}{P + R}$

F สูงดี