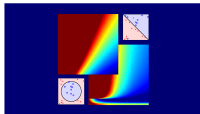# Machine Learning Techniques
(機器學習技巧)

Lecture 9: Decision Tree

Hsuan-Tien Lin (林軒田)

htlin@csie.ntu.edu.tw

Department of Computer Science
& Information Engineering

National Taiwan University
(國立台灣大學資訊工程系)

Agenda

Lecture 9: Decision Tree

- Decision Tree Hypothesis
- Decision Tree Algorithm
- Decision Tree in Practice
- Decision Tree in Action

What We Have Done

blending: aggregate after getting g_t;
learning: aggregate as well as getting g_t

| aggregation type	blending	learning
uniform	voting/averaging	Bagging
non-uniform	linear	AdaBoost
conditional	stacking	**Decision Tree**

decision tree: a traditional learning model that
realizes **conditional aggregation**

Decision Tree for Playing Golf

$$G(\mathbf{x}) = \sum_{t=1}^{T} q_t(\mathbf{x}) \cdot g_t(\mathbf{x})$$

- **base hypothesis** $g_t(\mathbf{x})$: leaf at end of path t, a **constant** here
- **condition** $q_t(\mathbf{x})$: [[is $\mathbf{x}$ on path t?]]
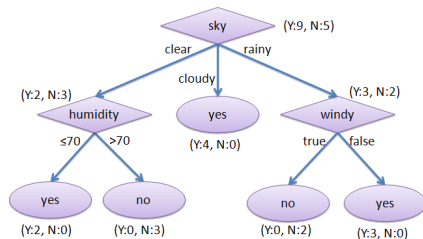- usually with **simple internal nodes**
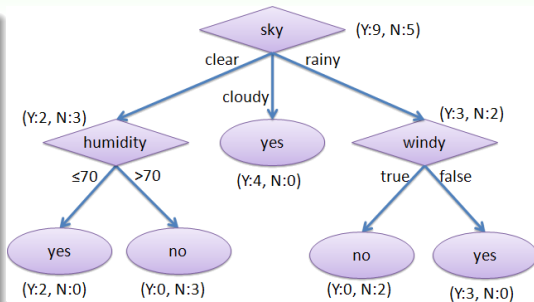
decision tree: arguably one of the most
human-mimicking models

Recursive View of Decision Tree

Path View: $G(\mathbf{x}) = \sum_{t=1}^{T} [\![\mathbf{x} \text{ on path } t]\!] \cdot \text{leaf}_t(\mathbf{x})$

Recursive View

$$G(\mathbf{x}) = \sum_{c=1}^{C} [\![b(\mathbf{x}) = c]\!] \cdot G_c(\mathbf{x})$$

- $G(\mathbf{x})$: full-tree hypothesis
- $b(\mathbf{x})$: branching criteria
- $G_c(\mathbf{x})$: sub-tree hypothesis at the c-th branch

tree = (root, subtrees), just like what
your data structure instructor would say :-)

Disclaimers about Decision Tree

Usefulness

- human-explainable: **widely used** in business/medical data analysis
- simple: **even freshmen can implement one :-)**
- efficient in prediction and **training**

However......

- heuristic: mostly **little theoretical** explanations
- heuristic**s**: 'heuristic**s** selection' confusing to beginners
- arguably no single **representative algorithm**

decision tree: mostly **heuristic but useful** on its own

Fun Time

A Basic Decision Tree Algorithm

$$G(\mathbf{x}) = \sum_{c=1}^{C} [\![b(\mathbf{x}) = c]\!] \, G_c(\mathbf{x})$$

function DecisionTree$\big(\text{data } \mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}\big)$

if termination criteria met

      return base hypothesis $g_t(\mathbf{x})$

else

    ① learn branching criteria $b(\mathbf{x})$

    ② split $\mathcal{D}$ to C parts $\mathcal{D}_c = \{(\mathbf{x}_n, y_n) \colon b(\mathbf{x}_n) = c\}$

    ③ build sub-tree $G_c \leftarrow$ DecisionTree$(\mathcal{D}_c)$

    ④ return $G(\mathbf{x}) = \sum_{c=1}^{C} [\![b(\mathbf{x}) = c]\!] \, G_c(\mathbf{x})$

four choices: number of branches, branching criteria, termination criteria, & base hypothesis

Classification and Regression Tree (C&RT)

function DecisionTree(data $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$)
if termination criteria met
      return base hypothesis $g_t(\mathbf{x})$
else ...
      2 split $\mathcal{D}$ to C parts $\mathcal{D}_c = \{(\mathbf{x}_n, y_n): b(\mathbf{x}_n) = c\}$

two simple choices

- $C = 2$ (binary tree)
- $g_t(\mathbf{x}) = E_{\text{in}}$-optimal constant
    - binary/multiclass classification (0/1 error): majority of $\{y_n\}$
    - regression (squared error): average of $\{y_n\}$

disclaimer:
C&RT here is based on **selected components**
of **CART^{TM} of California Statistical Software**

Branching in C&RT: Purifying

function DecisionTree(data $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$)
if termination criteria met
      return base hypothesis $g_t(\mathbf{x}) = E_{\text{in}}$-optimal constant
else ...
      ① learn branching criteria $b(\mathbf{x})$
      ② split $\mathcal{D}$ to 2 parts $\mathcal{D}_c = \{(\mathbf{x}_n, y_n)\colon b(\mathbf{x}_n) = c\}$

more simple choices

- simple internal node for $C = 2$: $\{1, 2\}$-**output decision stump**
- 'easier' subtree: branch by **purifying**

$$b(\mathbf{x}) = \underset{\text{decision stumps } h(\mathbf{x})}{\operatorname{argmin}} \sum_{c=1}^{2} |\mathcal{D}_c \text{ with } h| \cdot \text{impurity}(\mathcal{D}_c \text{ with } h)$$

C&RT: **bi**-**branching** by **purifying**

Impurity Functions

by E_{in} of optimal constant

- regression error:

$$\text{impurity}(\mathcal{D}) = \frac{1}{N} \sum_{n=1}^{N} (y_n - \bar{y})^2$$

with $\bar{y}$ = average of $\{y_n\}$

- classification error:

$$\text{impurity}(\mathcal{D}) = \frac{1}{N} \sum_{n=1}^{N} [\![y_n = y^*]\!]$$

with y^* = majority of $\{y_n\}$

for classification

- Gini index:

$$1 - \frac{1}{K} \sum_{k=1}^{K} \left(\frac{\sum_{n=1}^{N} [\![y_n = k]\!]}{N} \right)^2$$

—all k considered together

- classification error:

$$1 - \max_{1 \leq k \leq K} \frac{\sum_{n=1}^{N} [\![y_n = k]\!]}{N}$$

—optimal $k = y^*$ only

popular choices: **Gini** for classification,
regression error for regression

Termination in C&RT

function DecisionTree(data $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$)
if termination criteria met
    return base hypothesis $g_t(\mathbf{x}) = E_{\text{in}}$-optimal constant
else ...

    ① learn branching criteria

$$b(\mathbf{x}) = \operatorname*{argmin}_{\text{decision stumps } h(\mathbf{x})} \sum_{c=1}^{2} |\mathcal{D}_c \text{ with } h| \cdot \text{impurity}(\mathcal{D}_c \text{ with } h)$$

'forced' to terminate when

- all y_n the same: impurity = 0 $\Longrightarrow g_t(\mathbf{x}) = y_n$
- all $\mathbf{x}_n$ the same: **no decision stumps**

> C&RT: **fully-grown tree** with constant leaves
> that come from **bi-branching** by **purifying**

Fun Time

Basic C&RT Algorithm

function DecisionTree(data $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$)

if cannot branch anymore

return $g_t(\mathbf{x}) = E_{\text{in}}$-optimal constant

else

① learn branching criteria

$$b(\mathbf{x}) = \underset{\text{decision stumps } h(\mathbf{x})}{\text{argmin}} \sum_{c=1}^{2} |\mathcal{D}_c \text{ with } h| \cdot \text{impurity}(\mathcal{D}_c \text{ with } h)$$

② split $\mathcal{D}$ to 2 parts $\mathcal{D}_c = \{(\mathbf{x}_n, y_n) \colon b(\mathbf{x}_n) = c\}$

③ build sub-tree $G_c \leftarrow$ DecisionTree($\mathcal{D}_c$)

④ return $G(\mathbf{x}) = \sum_{c=1}^{2} [\![b(\mathbf{x}) = c]\!] G_c(\mathbf{x})$

> easily handle binary classification,
> regression, & **multi-class classification**

Regularization by Pruning

fully-grown tree: $E_{in}(G) = 0$ if all $\mathbf{x}_n$ different
but **overfit** (large E_{out}) because **low-level trees built with small** $\mathcal{D}_c$

- need a **regularizer**, say, $\Omega(G) = \text{NumberOfLeaves}(G)$
- want **regularized** decision tree:

$$\underset{\text{all possible } G}{\text{argmin}} \quad E_{in}(G) + \lambda\Omega(G)$$

  —called **pruned** decision tree

- cannot enumerate all possible G computationally:
  —often consider only
  - $G^{(0)}$ = fully-grown tree
  - $G^{(i)} = \text{argmin}_G E_{in}(G)$ such that G is **one-leaf removed** from $G^{(i-1)}$

systematic choice of λ? **validation**

Branching on Categorical Features

numerical features

blood pressure:
130, 98, 115, 147, 120

categorical features

major symptom:
fever, pain, tired, sweaty

branching for numerical

decision stump

$$b(\mathbf{x}) = [\![x_i \leq \theta]\!] + 1$$

with $\theta \in \mathbb{R}$

branching for categorical

decision subset

$$b(\mathbf{x}) = [\![x_i \in S]\!] + 1$$

with $S \subset \{1, 2, \ldots, K\}$

C&RT (& general decision trees):
handles **categorical features easily**

Missing Features by Surrogate Branch

possible $b(\mathbf{x}) = [\![\text{weight} \le 50\text{kg}]\!]$

if weight missing during prediction:

- what would human do?
  - go get weight
  - or, use threshold on height instead, because
    threshold on height $\approx$ threshold on weight
- surrogate branch:
  - maintain surrogate branch $b_1(\mathbf{x})$, $b_2(\mathbf{x})$, ... $\approx$ best branch $b(\mathbf{x})$
    during training
  - allows missing feature for $b(\mathbf{x})$ during prediction by using surrogate
    instead

C&RT: handles **missing features easily**

A Simple Data Set
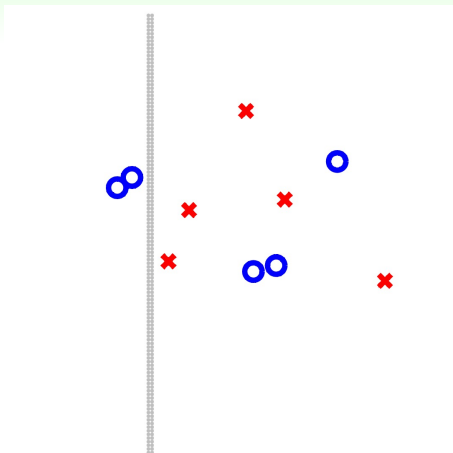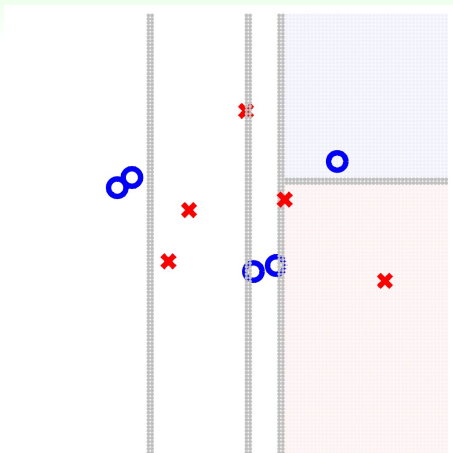
A Simple Data Set
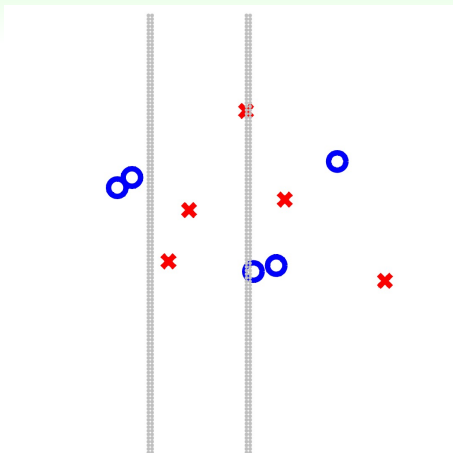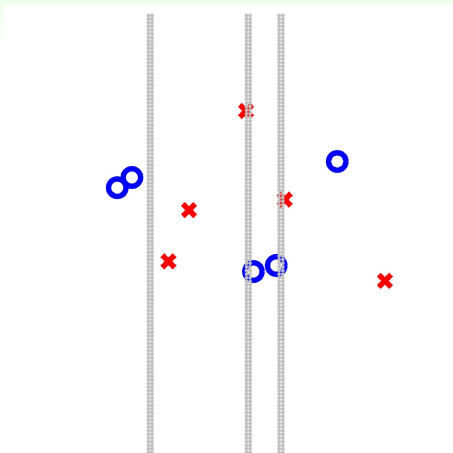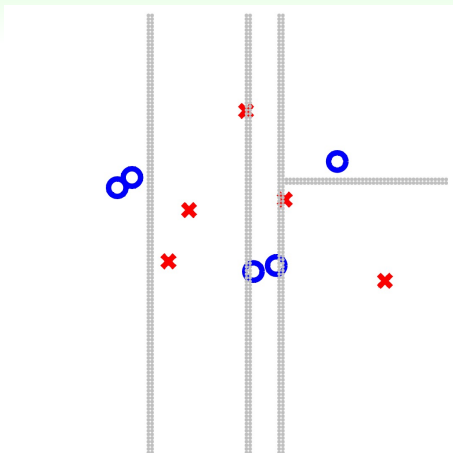
A Simple Data Set

A Simple Data Set
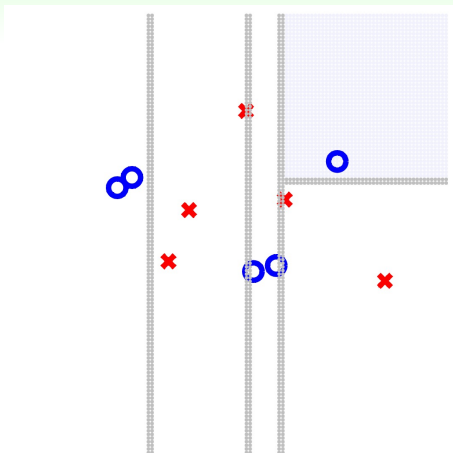
A Simple Data Set

A Simple Data Set

A Simple Data Set

A Simple Data Set

A Simple Data Set

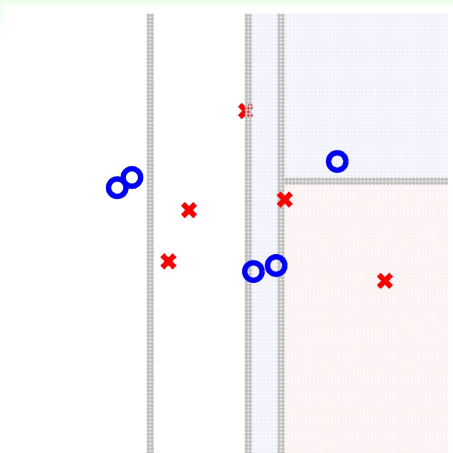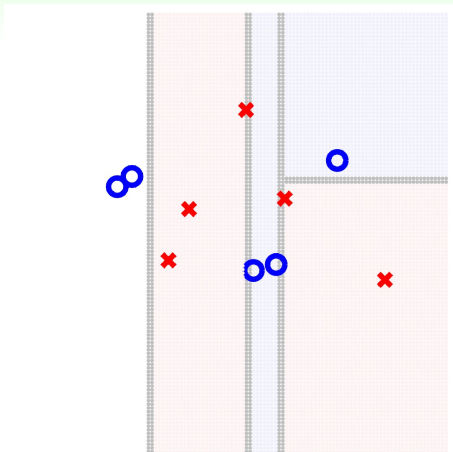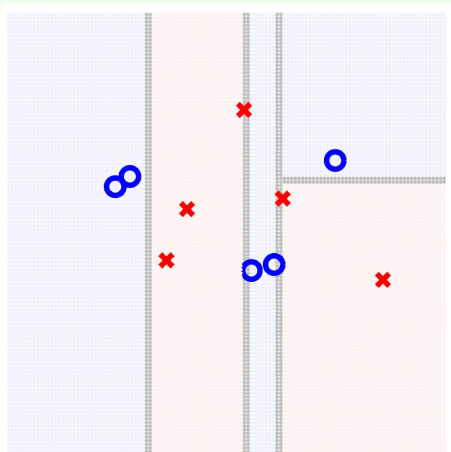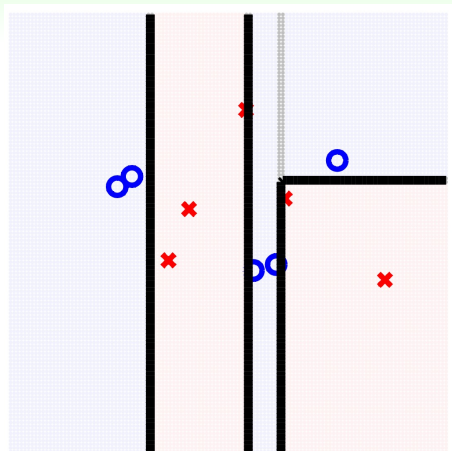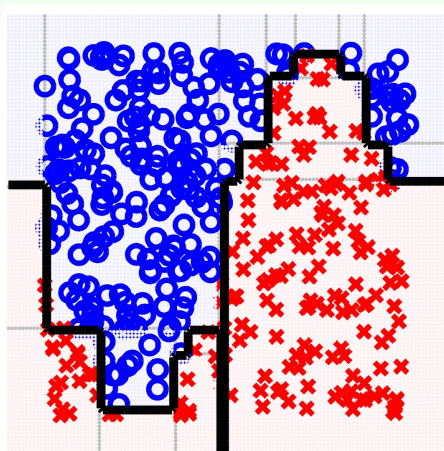# A Simple Data Set

A Complicated Data Set

Practical Specialties of C&RT

- human-explainable
- multiclass easily
- categorical features easily
- missing features easily

—almost no other learning model share all such specialties, except for **other decision trees**

> **another** popular decision tree algorithm:
> **C4.5**, with different **choices of heuristics**

Fun Time

Summary

Lecture 9: Decision Tree

- Decision Tree Hypothesis

- Decision Tree Algorithm

- Decision Tree in Practice

- Decision Tree in Action