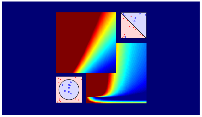
Machine Learning Techniques
(機器學習技巧)



Lecture 8: Adaptive Boosting

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)

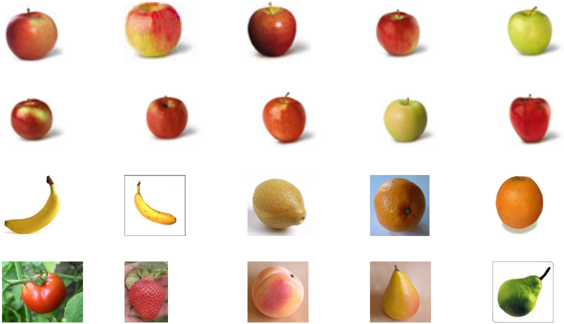
Agenda

Lecture 8: Adaptive Boosting

- Motivation of Boosting
- Diversify by Re-weighting
- Adaptive Boosting Algorithm
- Adaptive Boosting in Action

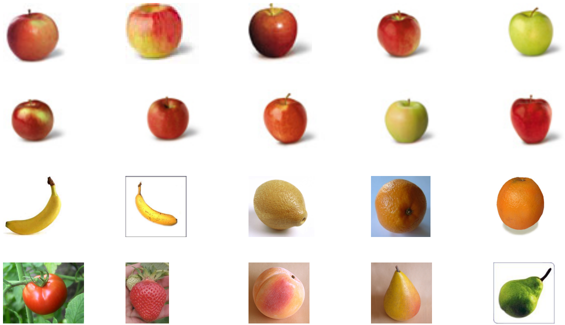
Apple Recognition Problem

- is this a picture of an apple?
- say, want to teach a class of **6 year olds**
- gather photos from NY Apple Asso. and Google Image



Our Fruit Class Begins

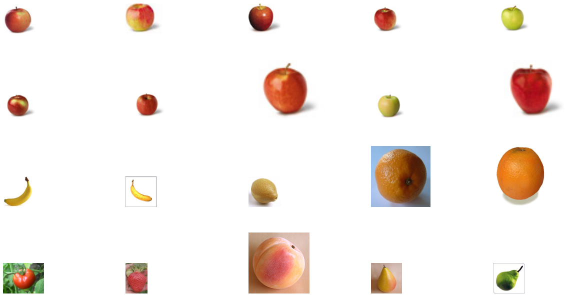
- Teacher: Please look at the pictures of apples and non-apples below. Based on those pictures, how would you describe an apple? Michael?
- Michael: I think apples are **circular**.



(Class): Apples are **circular**.

Our Fruit Class Continues

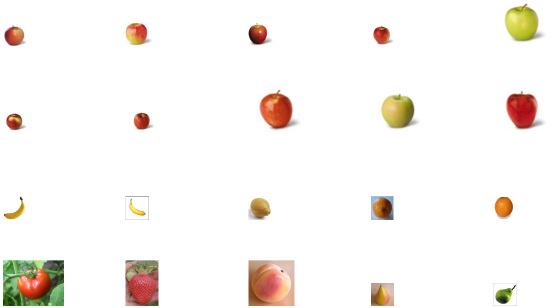
- Teacher: Being circular is a good feature for the apples. However, if you only say circular, you could make several mistakes. What else can we say for an apple? Tina?
- Tina: It looks like apples are **red**.



(Class): Apples are somewhat **circular** and somewhat **red**.

Our Fruit Class Continues

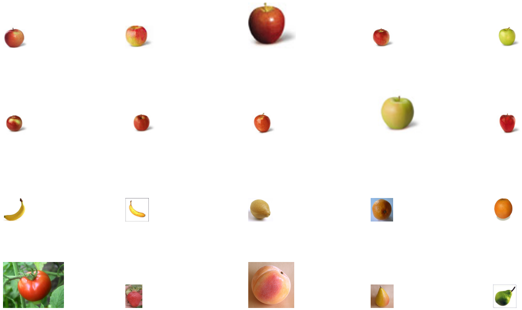
- Teacher: Yes. Many apples are red. However, you could still make mistakes based on circular and red. Do you have any other suggestions, Joey?
- Joey: Apples could also be **green**.



(Class): Apples are somewhat **circular** and somewhat **red** and possibly **green**.

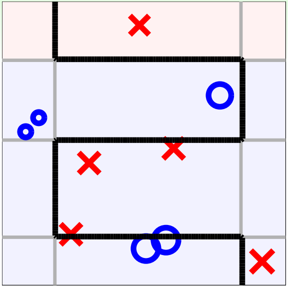
Our Fruit Class Continues

- Teacher: Yes. It seems that apples might be circular, red, green. But you may confuse them with tomatoes or peaches, right? Any more suggestions, Jessica?
- Jessica: Apples have **stems** at the top.



(Class): Apples are somewhat **circular**, somewhat **red**, possibly **green**, and may have **stems** at the top.

Motivation



- students: simple hypotheses g_t
- (Class): sophisticated hypothesis G
- Teacher: a tactic learning algorithm that **direct the students to focus on key examples**

next: the 'math' of such an algorithm

Fun Time

Bootstrapping as Re-weighting Process

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), (\mathbf{x}_4, y_4)\}$$

$$\xrightarrow{\text{bootstrap}} \quad \tilde{\mathcal{D}}_t = \{(\mathbf{x}_1, y_1), (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_4, y_4)\}$$

weighted E_{in} on $\mathcal{D}$

$$E_{\text{in}}^{\mathbf{u}}(h) = \frac{1}{4}\sum_{n=1}^{4} u_n^{(t)} \cdot [\![y_n \neq h(\mathbf{x}_n)]\!]$$

$(\mathbf{x}_1, y_1)$, $u_1 = 2$
$(\mathbf{x}_2, y_2)$, $u_2 = 1$
$(\mathbf{x}_3, y_3)$, $u_3 = 0$
$(\mathbf{x}_4, y_4)$, $u_4 = 1$

E_{in} on $\tilde{D}_t$

$$E_{\text{in}}^{0/1}(h) = \frac{1}{4}\sum_{(\mathbf{x},y)\in\tilde{D}_t} [\![y \neq h(\mathbf{x})]\!]$$

$(\mathbf{x}_1, y_1)$, $(\mathbf{x}_1, y_1)$
$(\mathbf{x}_2, y_2)$

$(\mathbf{x}_4, y_4)$

each diverse g_t in bagging:
by minimizing bootstrap-weighted error

Weighted Base Algorithm

minimize (regularized)

$$E_{\text{in}}^{\mathbf{u}}(h) = \frac{1}{N} \sum_{n=1}^{N} u_n \cdot \text{err}(y_n, h(\mathbf{x}_n))$$

SVM

$E_{\text{in}}^{\mathbf{u}} \propto C \sum_{n=1}^{N} u_n \widehat{\text{err}}_{\text{SVM}}$

by dual QP

$\Leftrightarrow$ loosen bound

$0 \leq \alpha_n \leq C u_n$

linear regression

$E_{\text{in}}^{\mathbf{u}} \propto \sum_{n=1}^{N} u_n (y_n - \mathbf{w}^T \mathbf{x}_n)^2$

by analytic solution

$\Leftrightarrow$ scale data

$(\mathbf{x}'_n, y'_n) = \sqrt{u_n}(\mathbf{x}_n, y_n)$

logistic regression

$E_{\text{in}}^{\mathbf{u}} \propto \sum_{n=1}^{N} u_n \text{err}_{\text{CE}}$

by SGD

$\Leftrightarrow$ sample $(\mathbf{x}_n, y_n)$ proportional to u_n

example-weighted learning: extension of **class-weighted** learning in Lecture 8

Re-weighting for More Diverse Hypothesis

'improving' bagging for binary classification:
how to re-weight for **more diverse hypotheses**?

$$g_t \leftarrow \operatorname*{argmin}_{h \in \mathcal{H}} \left(\sum_{n=1}^{N} u_n^{(t)} [\![y_n \neq h(\mathbf{x}_n)]\!] \right)$$

$$g_{t+1} \leftarrow \operatorname*{argmin}_{h \in \mathcal{H}} \left(\sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq h(\mathbf{x}_n)]\!] \right)$$

if g_t '**not good**' for $\mathbf{u}^{(t+1)}$ $\Longrightarrow$ g_t-like hypotheses not returned as g_{t+1}
$\Longrightarrow$ g_{t+1} diverse from g_t

idea: **construct** $\mathbf{u}^{(t+1)}$ to make g_t **random-like**

$$\frac{\sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]}{\sum_{n=1}^{N} u_n^{(t+1)}} = \frac{1}{2}$$

'Optimal' Re-weighting

want: $$\frac{\sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]}{\sum_{n=1}^{N} u_n^{(t+1)}} = \frac{1}{2}$$

- re-write: $\frac{\square_{t+1}}{\square_{t+1} + \bigcirc_{t+1}} = \frac{1}{2}$, with

$$\square_{t+1} = \sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq g_t(\mathbf{x}_n)]\!], \quad \bigcirc_{t+1} = \sum_{n=1}^{N} u_n^{(t+1)} [\![y_n = g_t(\mathbf{x}_n)]\!]$$

- need: (total $u_n^{(t+1)}$ of incorrect) = (total $u_n^{(t+1)}$ of incorrect)
- how? with $\epsilon_t = \frac{\square_t}{\square_t + \bigcirc_t}$

$\square_{t+1} \leftarrow \square_t \cdot \bigcirc_t$	$\bigcirc_{t+1} \leftarrow \bigcirc_t \cdot \square_t$
incorrect: $u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot (1 - \epsilon_t)$	correct: $u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot \epsilon_t$

'optimal' re-weighting for diverse hypotheses:

scale incorrect $\propto (1 - \epsilon_t)$; scale correct $\propto \epsilon_t$

Fun Time

Scaling Factor

'optimal' re-weighting:

$$\text{scale incorrect} \propto (1 - \epsilon_t); \text{ scale correct} \propto \epsilon_t$$

define scaling factor $\diamondsuit_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$

$$\text{incorrect} \leftarrow \text{incorrect} \cdot \diamondsuit_t; \text{ correct} \leftarrow \text{correct} / \diamondsuit_t;$$

- equivalent to optimal re-weighting
- $\diamondsuit_t \geq 1$ iff $\epsilon_t \leq \frac{1}{2}$
  —physical meaning: scale up incorrect; scale down correct
  —like what Teacher does

scaling-up incorrect examples
leads to **diverse hypotheses**

A Preliminary Algorithm

$\mathbf{u}^{(1)} = ?$

for $t = 1, 2, \ldots, T$

1. obtain g_t by $\mathcal{A}(\mathcal{D}, \mathbf{u}^{(t)})$, where $\mathcal{A}$ tries to minimize $\mathbf{u}^{(t)}$-weighted 0/1 error
2. update $\mathbf{u}^{(t)}$ to $\mathbf{u}^{(t+1)}$ by $\diamondsuit_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$, where ϵ_t = weighted error rate of g_t

return $G(\mathbf{x}) = ?$

- want g_1 'best' for E_{in}: $u_n^{(1)} = \frac{1}{N}$
- $G(\mathbf{x})$:
  - uniform? but g_2 very bad for E_{in} (**why?** :-))
  - linear, non-linear? **as you wish**

next: a special algorithm to aggregate **on the fly** with theoretical guarantee

Adaptive Boosting

$\mathbf{u}^{(1)} = \frac{1}{N}$

for $t = 1, 2, \ldots, T$

1. obtain g_t by $\mathcal{A}(\mathcal{D}, \mathbf{u}^{(t)})$, where ...
2. update $\mathbf{u}^{(t)}$ to $\mathbf{u}^{(t+1)}$ by $\lozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$, where ...
3. compute α_t

return $G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t g_t(\mathbf{x})\right)$

- wish: large α_t for 'good' $g_t \Longleftarrow \alpha_t = \text{monotonic}(\lozenge_t)$
- will take $\alpha_t = \ln(\lozenge_t)$
  - $\epsilon_t = \frac{1}{2} \Longrightarrow \lozenge_t = 1 \Longrightarrow \alpha_t = 0$ (bad g_t zero weight)
  - $\epsilon_t = 0 \Longrightarrow \lozenge_t = \infty \Longrightarrow \alpha_t = \infty$ (super g_t superior weight)

Adaptive Boosting (AdaBoost): with such α_t, provable **boosting property**

Theoretical Guarantee of AdaBoost

- From VC bound

$$E_{\text{out}}(G) \le E_{\text{in}}(G) + O\left(\sqrt{\frac{d_{\text{VC}}(\text{lin}(\mathcal{H}))}{N} \log N}\right)$$

- **first term can be small** (to be proved in homework): $E_{\text{in}}(G) = 0$ after $T = O(\log N)$ iterations if $\epsilon_t \le \epsilon < \frac{1}{2}$ always
- **second term can be small**: $d_{\text{VC}}(\text{lin}(\mathcal{H})) = O(d_{\text{VC}}(\mathcal{H}) \cdot T \log T)$ grows "slowly" with T

boosting view of AdaBoost:

if $\mathcal{A}$ is weak but always **slightly better than random** ($\epsilon_t \le \epsilon < \frac{1}{2}$)
(AdaBoost+$\mathcal{A}$) can be strong ($E_{\text{in}} = 0$ and E_{out} small).

Fun Time

Decision Stump

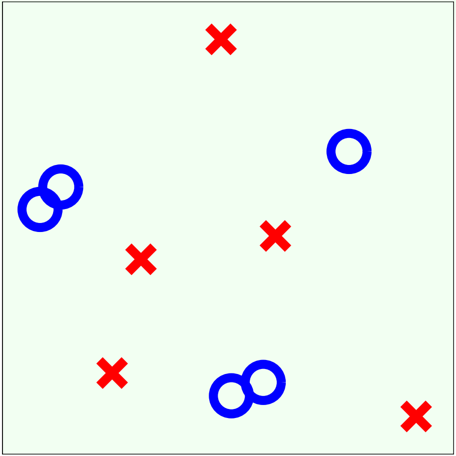
want: a '**weak**' base learning algorithm $\mathcal{A}$
that minimizes $E_{\text{in}}^{\mathbf{u}}(h) = \frac{1}{N} \sum_{n=1}^{N} u_n \cdot [\![y_n \neq h(\mathbf{x}_n)]\!]$ **a little bit**

a popular choice: decision stump

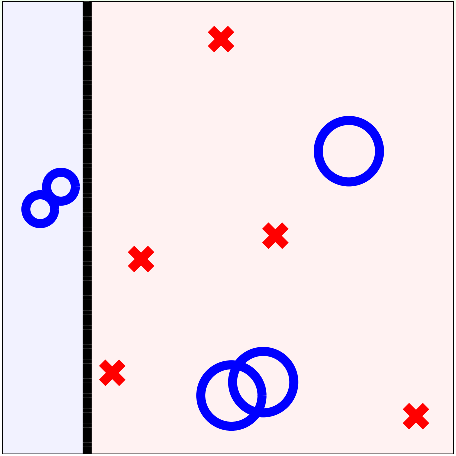
- positive and negative rays on some feature: three parameters (feature, threshold, direction)
- physical meaning: vertical/horizontal lines in 2D, or hyperplanes $\perp$ natural axes
- efficient to optimize: $O(d \cdot N \log N)$ time

decision stump model:
allows efficient minimization of $E_{\text{in}}^{\mathbf{u}}$
but perhaps **too weak to use by itself**

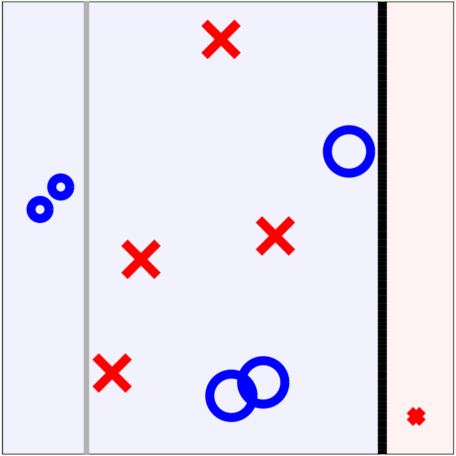
A Simple Data Set



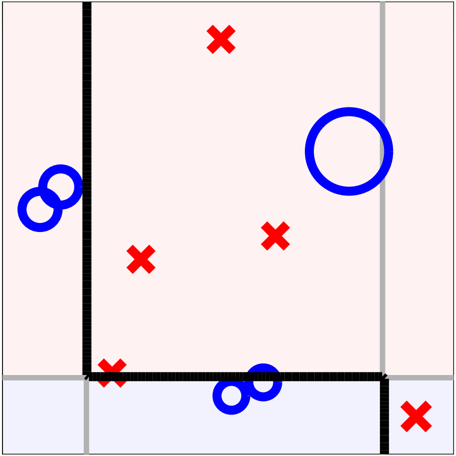
A Simple Data Set



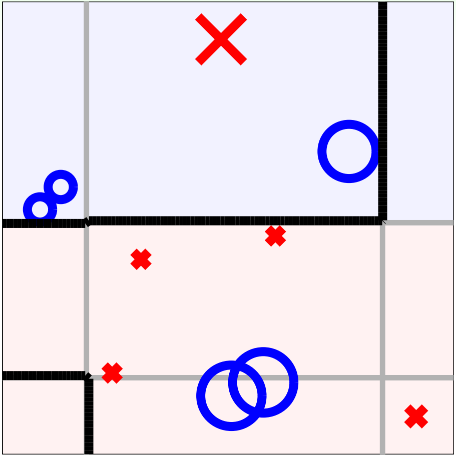
A Simple Data Set



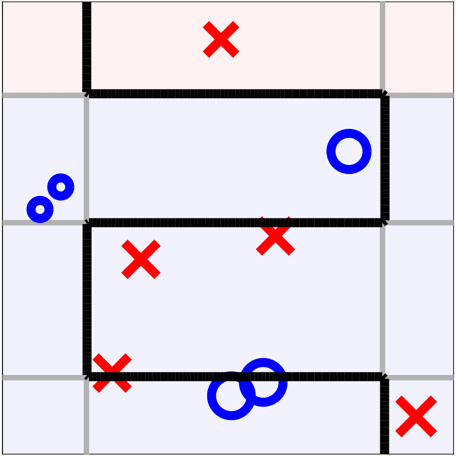
A Simple Data Set



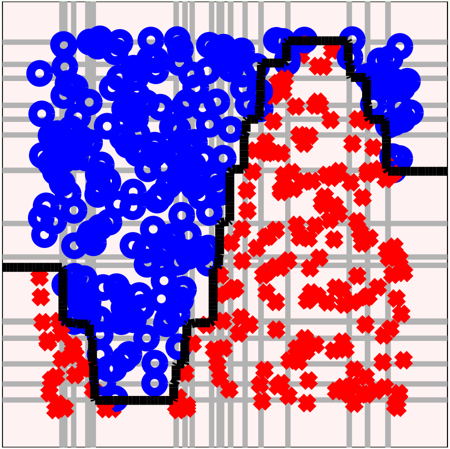
A Simple Data Set



A Simple Data Set



A Complicated Data Set



Fun Time

Summary

Lecture 8: Adaptive Boosting

- Motivation of Boosting
- Diversify by Re-weighting
- Adaptive Boosting Algorithm
- Adaptive Boosting in Action