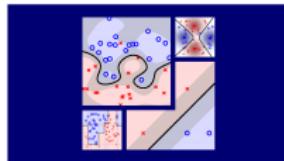


Machine Learning Techniques (機器學習技法)



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)



Roadmap

- ① Embedding Numerous Features: Kernel Models
- ② Combining Predictive Features: Aggregation Models

Lecture 7: Blending and Bagging

blending known diverse hypotheses **uniformly**, **linearly**, or even **non-linearly**; obtaining diverse hypotheses from **bootstrapped data**

Lecture 8: Adaptive Boosting

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

- ③ Distilling Implicit Features: Extraction Models

Apple Recognition Problem

- is this a picture of an apple?
- say, want to teach a class of **6 year olds**
- gather photos under CC-BY-2.0 license on Flickr
(thanks to the authors below!)

(APAL stands for Apple and Pear Australia Ltd)



Dan Foy

<https://flic.kr/p/jNQ55>



APAL

<https://flic.kr/p/jzP1VB>



adrianbartel

<https://flic.kr/p/bdy2hZ>



ANdrzej ch.

<https://flic.kr/p/51DKA8>



Stuart Webster

<https://flic.kr/p/9C3Ybd>



nachans

<https://flic.kr/p/9XD7Ag>



APAL

<https://flic.kr/p/jzRe4u>



Jo Jakeman

<https://flic.kr/p/7jwtGp>



APAL

<https://flic.kr/p/jzPYNr>



APAL

<https://flic.kr/p/jzScif>

Apple Recognition Problem

- is this a picture of an apple?
- say, want to teach a class of **6 year olds**
- gather photos under CC-BY-2.0 license on Flickr
(thanks to the authors below!)



Mr. Roboto.

<https://flic.kr/p/i5BN85>



Richard North

<https://flic.kr/p/bHhPkB>



Richard North

<https://flic.kr/p/d8tGou>



Emilian Robert Vicol

<https://flic.kr/p/bpmGXW>



Nathaniel Queen Mc-

<https://flic.kr/p/pZv1Mf>



Crystal

<https://flic.kr/p/kaPYp>



jf686

<https://flic.kr/p/6vjRFH>



skyseeker

<https://flic.kr/p/2MynV>



Janet Hudson

<https://flic.kr/p/7QDBbm>

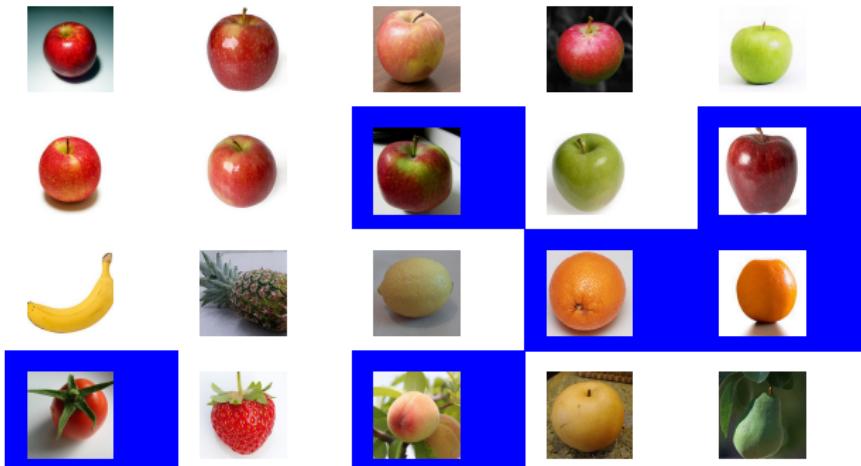


Rennett Stowe

<https://flic.kr/p/agmnrk>

Our Fruit Class Begins

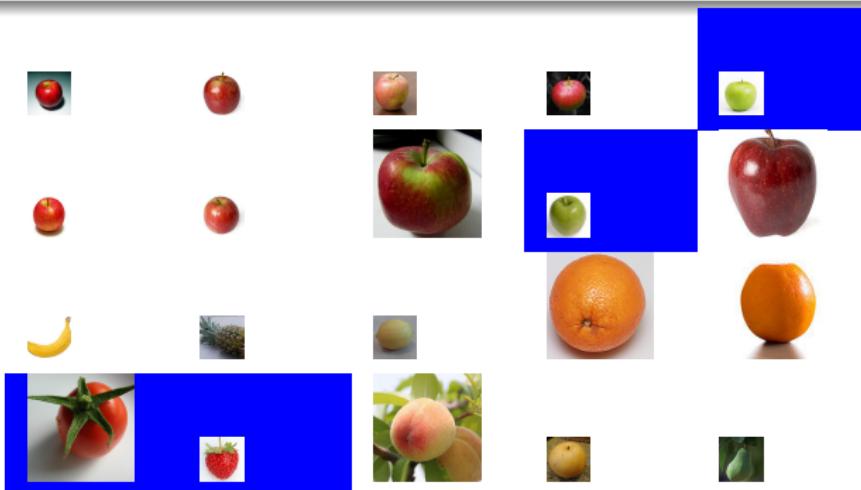
- Teacher: Please look at the pictures of apples and non-apples below. Based on those pictures, how would you describe an apple? Michael?
- Michael: I think apples are **circular**.



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

Our Fruit Class Continues

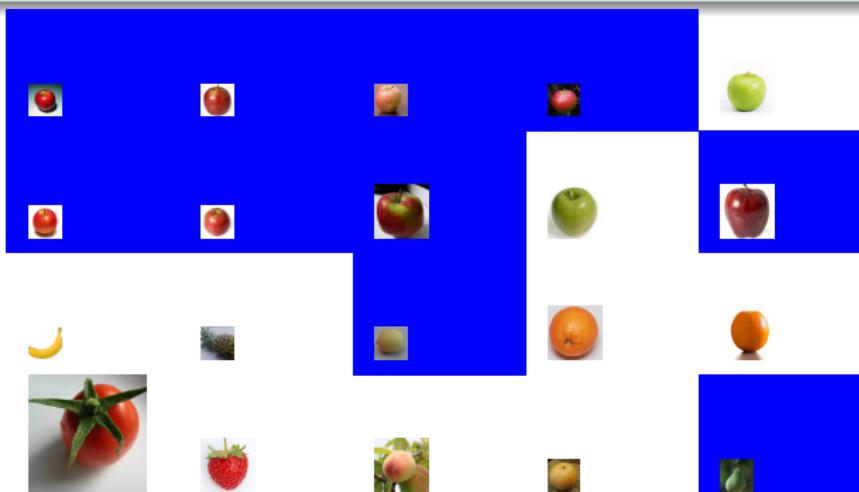
- Teacher: Being circular is a good feature for the apples. However, if you only say circular, you could make several mistakes. What else can we say for an apple? Tina?
- Tina: It looks like apples are **red**.



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

Our Fruit Class Continues More

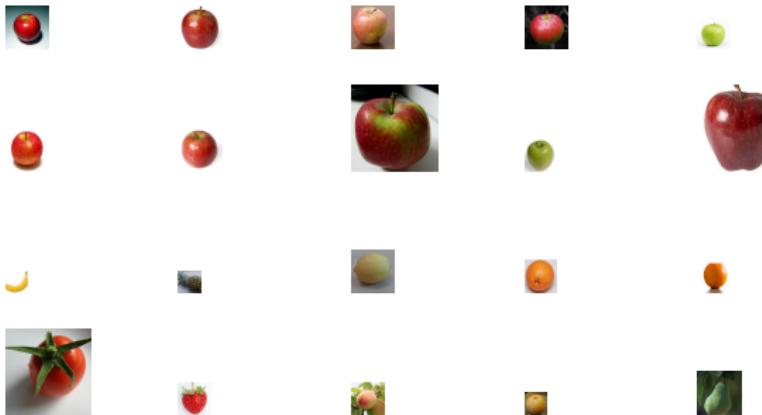
- Teacher: Yes. Many apples are red. However, you could still make mistakes based on circular and red. Do you have any other suggestions, Joey?
- Joey: Apples could also be **green**.



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

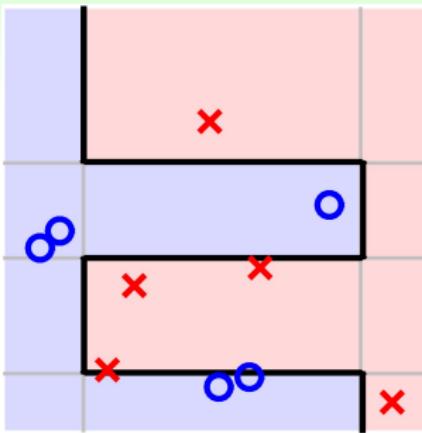
Our Fruit Class Ends

- 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 (like vertical/horizontal lines)
- (Class): sophisticated hypothesis G (like black curve)
- Teacher: a tactic learning algorithm that **directs the students to focus on key examples**

next: the '**math**' of such an algorithm

Fun Time

Which of the following can help recognize an apple?

- ① apples are often circular
- ② apples are often red or green
- ③ apples often have stems at the top
- ④ all of the above

Fun Time

Which of the following can help recognize an apple?

- ① apples are often circular
- ② apples are often red or green
- ③ apples often have stems at the top
- ④ all of the above

Reference Answer: ④

Congratulations! You have passed first
grade. :-)

Bootstrapping as Re-weighting Process

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

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

weighted E_{in} on \mathcal{D}

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

(\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 $\tilde{\mathcal{D}}_t$

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

(\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)

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

SVM

$E_{\text{in}}^{\mathbf{u}} \propto C \sum_{n=1}^N \mathbf{u}_n \widehat{\text{err}}_{\text{SVM}}$ by dual QP
 \Leftrightarrow adjusted upper bound

$$0 \leq \alpha_n \leq C \mathbf{u}_n$$

logistic regression

$E_{\text{in}}^{\mathbf{u}} \propto \sum_{n=1}^N \mathbf{u}_n \text{err}_{\text{CE}}$ by SGD
 \Leftrightarrow sample (\mathbf{x}_n, y_n) with probability proportional to \mathbf{u}_n

example-weighted learning:

extension of **class-weighted** learning in Lecture 8 of ML Foundations

Re-weighting for More Diverse Hypothesis

'improving' bagging for binary classification:

how to re-weight for **more diverse hypotheses**?

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

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

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

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

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

'Optimal' Re-weighting

want: $\frac{\sum_{n=1}^N u_n^{(t+1)} \llbracket y_n \neq g_t(\mathbf{x}_n) \rrbracket}{\sum_{n=1}^N u_n^{(t+1)}} = \frac{\blacksquare_{t+1}}{\blacksquare_{t+1} + \bullet_{t+1}} = \frac{1}{2}$, where

$$\blacksquare_{t+1} = \sum_{n=1}^N u_n^{(t+1)} \llbracket y_n \neq g_t(\mathbf{x}_n) \rrbracket, \bullet_{t+1} = \sum_{n=1}^N u_n^{(t+1)} \llbracket y_n = g_t(\mathbf{x}_n) \rrbracket$$

- need: $\underbrace{\text{(total } u_n^{(t+1)} \text{ of incorrect)}}_{\blacksquare_{t+1}} = \underbrace{\text{(total } u_n^{(t+1)} \text{ of correct)}}_{\bullet_{t+1}}$
- one possibility by **re-scaling (multiplying) weights**, if

$$\text{(total } u_n^{(t)} \text{ of incorrect)} = 1126 ;$$

$$\text{(weighted incorrect rate)} = \frac{1126}{7337}$$

$$\text{incorrect: } u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot 6211$$

$$\text{(total } u_n^{(t)} \text{ of correct)} = 6211 ;$$

$$\text{(weighted correct rate)} = \frac{6211}{7337}$$

$$\text{correct: } u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot 1126$$

'optimal' re-weighting under **weighted incorrect rate ϵ_t** :

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

Fun Time

For four examples with $u_n^{(1)} = \frac{1}{4}$ for all examples. If g_1 predicts the first example wrongly but all the other three examples correctly. After the ‘optimal’ re-weighting, what is $u_1^{(2)}/u_2^{(2)}$?

- 1 4
- 2 3
- 3 1/3
- 4 1/4

Fun Time

For four examples with $u_n^{(1)} = \frac{1}{4}$ for all examples. If g_1 predicts the first example wrongly but all the other three examples correctly. After the ‘optimal’ re-weighting, what is $u_1^{(2)}/u_2^{(2)}$?

- ① 4
- ② 3
- ③ 1/3
- ④ 1/4

Reference Answer: ②

By ‘optimal’ re-weighting, u_1 is scaled proportional to $\frac{3}{4}$ and every other u_n is scaled proportional to $\frac{1}{4}$. So example 1 is now three times more important than any other example.

Scaling Factor

'optimal' re-weighting: let $\epsilon_t = \frac{\sum_{n=1}^N u_n^{(t)} \llbracket y_n \neq g_t(\mathbf{x}_n) \rrbracket}{\sum_{n=1}^N u_n^{(t)}} ,$

multiply **incorrect** $\propto (1 - \epsilon_t)$; multiply **correct** $\propto \epsilon_t$

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

$$\begin{array}{lcl} \text{incorrect} & \leftarrow & \text{incorrect} \cdot \diamond_t \\ \text{correct} & \leftarrow & \text{correct} / \diamond_t \end{array}$$

- **equivalent** to optimal re-weighting
- $\diamond_t \geq 1$ 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 $t = 1, 2, \dots, T$

- 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 $\Delta_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$,
where ϵ_t = weighted error (incorrect) rate of g_t

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

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

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

Linear Aggregation on the Fly

$$\mathbf{u}^{(1)} = [\frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}]$$

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

① obtain \mathbf{g}_t by $\mathcal{A}(\mathcal{D}, \mathbf{u}^{(t)})$, where ...

② update $\mathbf{u}^{(t)}$ to $\mathbf{u}^{(t+1)}$ by $\diamond_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$, where ...

③ compute $\alpha_t = \ln(\diamond_t)$

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

- wish: large α_t for ‘good’ $\mathbf{g}_t \iff \alpha_t = \text{monotonic}(\diamond_t)$
- will take $\alpha_t = \ln(\diamond_t)$
 - $\epsilon_t = \frac{1}{2} \implies \diamond_t = 1 \implies \alpha_t = 0$ (bad \mathbf{g}_t zero weight)
 - $\epsilon_t = 0 \implies \diamond_t = \infty \implies \alpha_t = \infty$ (super \mathbf{g}_t superior weight)

Adaptive Boosting = weak base learning algorithm \mathcal{A} (Student)
 + optimal re-weighting factor \diamond_t (Teacher)
 + ‘magic’ linear aggregation α_t (Class)

Adaptive Boosting (AdaBoost) Algorithm

$\mathbf{u}^{(1)} = [\frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}]$
for $t = 1, 2, \dots, T$

- 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

$$\begin{aligned} \llbracket y_n \neq g_t(\mathbf{x}_n) \rrbracket \text{ (incorrect examples): } u_n^{(t+1)} &\leftarrow u_n^{(t)} \cdot \diamond_t \\ \llbracket y_n = g_t(\mathbf{x}_n) \rrbracket \text{ (correct examples): } u_n^{(t+1)} &\leftarrow u_n^{(t)} / \diamond_t \end{aligned}$$

$$\text{where } \diamond_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}} \text{ and } \epsilon_t = \frac{\sum_{n=1}^N u_n^{(t)} \llbracket y_n \neq g_t(\mathbf{x}_n) \rrbracket}{\sum_{n=1}^N u_n^{(t)}}$$

- ③ compute $\alpha_t = \ln(\diamond_t)$

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

AdaBoost: provable **boosting property**

Theoretical Guarantee of AdaBoost

- From VC bound

$$E_{\text{out}}(G) \leq E_{\text{in}}(G) + O\left(\sqrt{\underbrace{O(d_{\text{VC}}(\mathcal{H}) \cdot T \log T)}_{d_{\text{VC}} \text{ of all possible } G} \cdot \frac{\log N}{N}}\right)$$

- first term can be small:**
 $E_{\text{in}}(G) = 0$ after $T = O(\log N)$ iterations if $\epsilon_t \leq \epsilon < \frac{1}{2}$ always
- second term can be small:**
 overall d_{VC} grows “slowly” with T

boosting view of AdaBoost:

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

Fun Time

According to $\alpha_t = \ln(\diamond_t)$, and $\diamond_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$, when would $\alpha_t > 0$?

- ① $\epsilon_t < \frac{1}{2}$
- ② $\epsilon_t > \frac{1}{2}$
- ③ $\epsilon_t \neq 1$
- ④ $\epsilon_t \neq 0$

Fun Time

According to $\alpha_t = \ln(\diamond_t)$, and $\diamond_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$, when would $\alpha_t > 0$?

- ① $\epsilon_t < \frac{1}{2}$
- ② $\epsilon_t > \frac{1}{2}$
- ③ $\epsilon_t \neq 1$
- ④ $\epsilon_t \neq 0$

Reference Answer: ①

The math part should be easy for you, and it is interesting to think about the physical meaning:
 $\alpha_t > 0$ (g_t is useful for G) if and only if the weighted error rate of g_t is better than random!

Decision Stump

want: a ‘**weak**’ base learning algorithm \mathcal{A}

that minimizes $E_{\text{in}}^{\text{u}}(h) = \frac{1}{N} \sum_{n=1}^N u_n \cdot [y_n \neq h(\mathbf{x}_n)]$ a little bit

a popular choice: decision stump

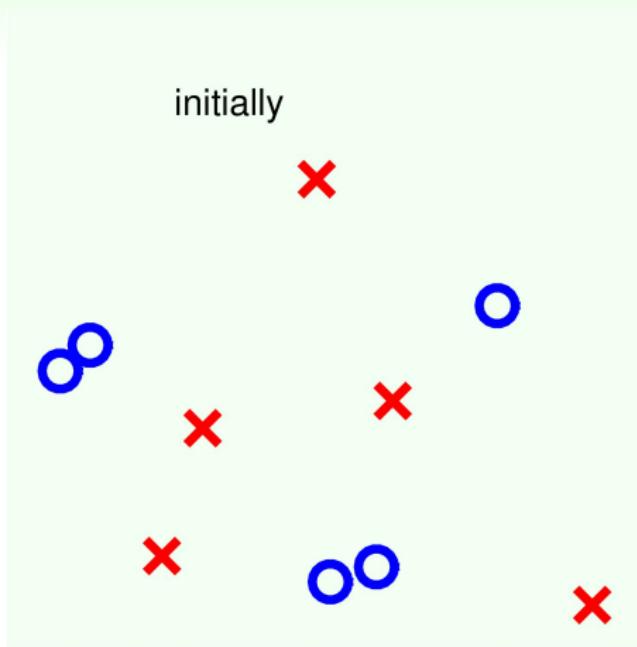
- in ML Foundations Homework 2, remember? :-)

$$h_{s,i,\theta}(\mathbf{x}) = s \cdot \text{sign}(x_i - \theta)$$

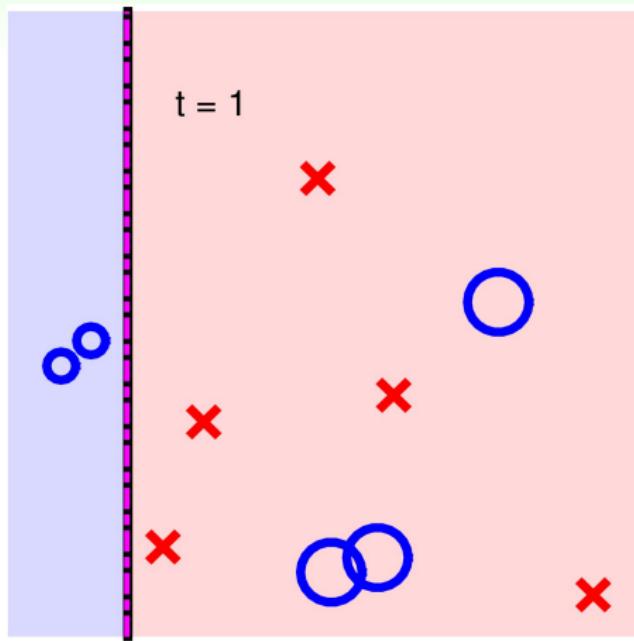
- positive and negative rays on some feature: three parameters (feature i , threshold θ , direction s)
- physical meaning: vertical/horizontal lines in 2D
- efficient to optimize: $O(d \cdot N \log N)$ time

decision stump model:
allows efficient minimization of E_{in}^{u}
but perhaps **too weak to work by itself**

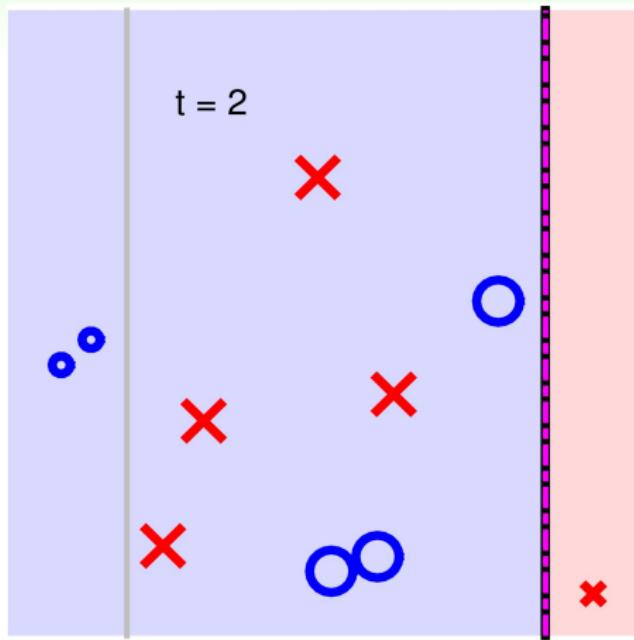
A Simple Data Set



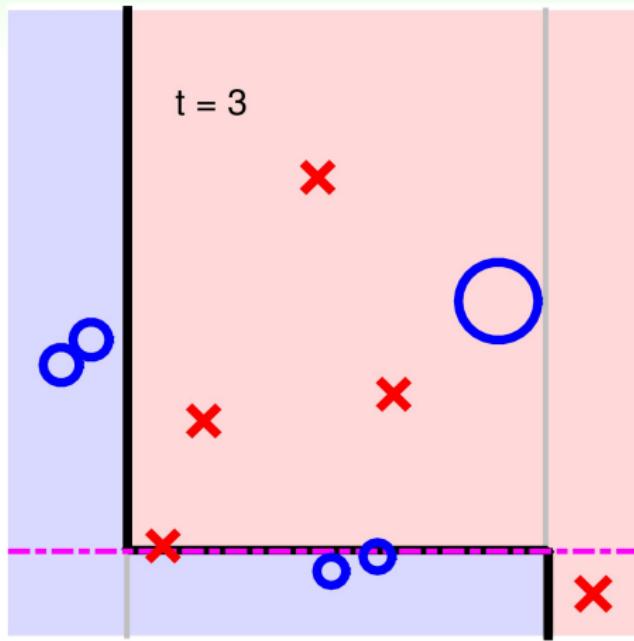
A Simple Data Set



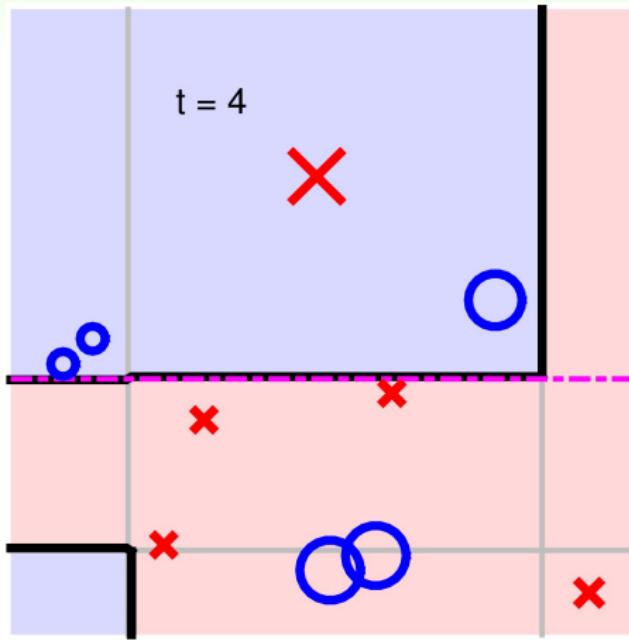
A Simple Data Set



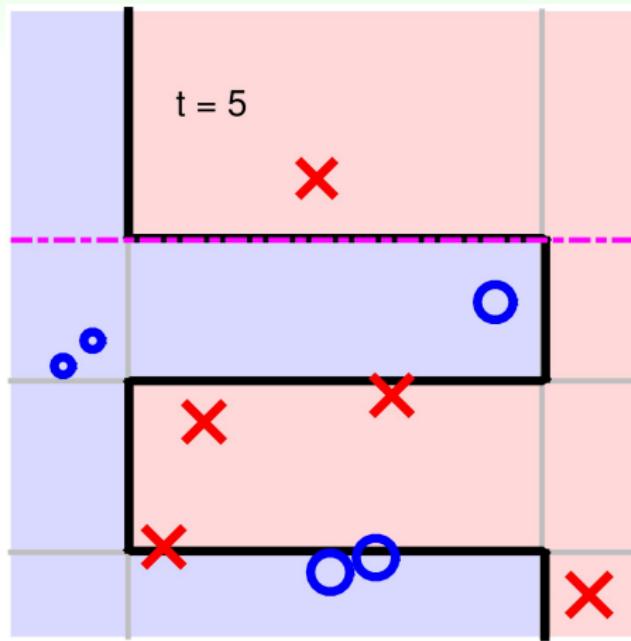
A Simple Data Set



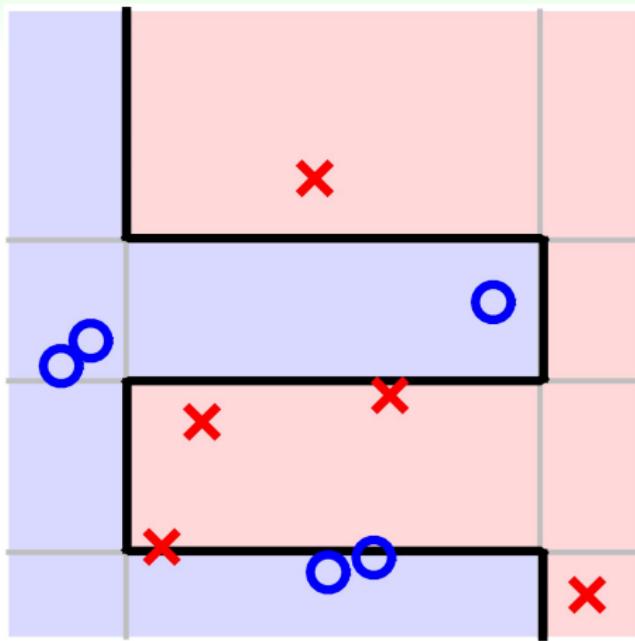
A Simple Data Set



A Simple Data Set

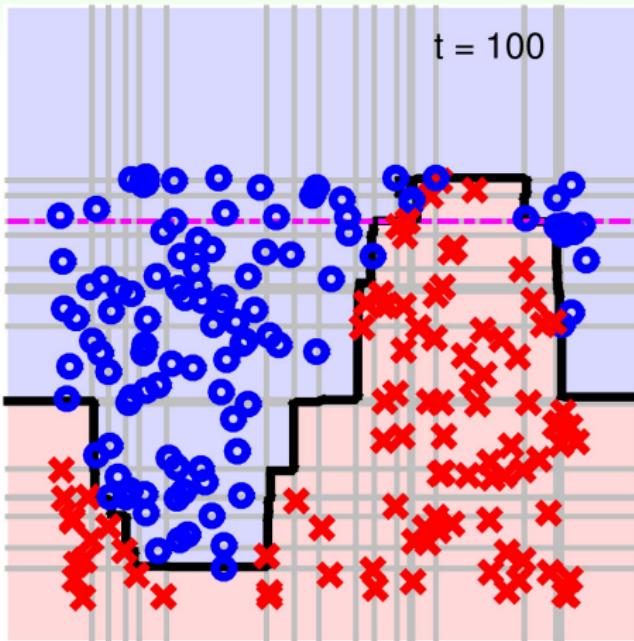


A Simple Data Set



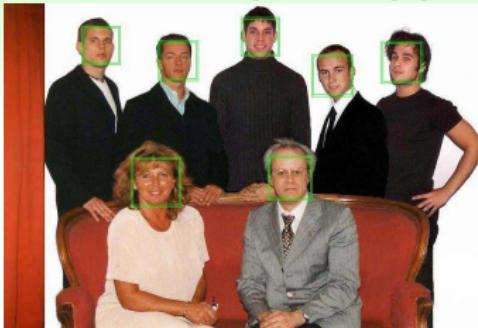
'Teacher'-like algorithm works!

A Complicated Data Set



AdaBoost-Stump: **non-linear yet efficient**

AdaBoost-Stump in Application



original picture by F.U.S.I.A. assistant and derivative work by Sylenius via Wikimedia Commons

The World's First 'Real-Time' Face Detection Program

- AdaBoost-Stump as core model: linear aggregation of key patches selected out of 162,336 possibilities in 24x24 images
 - feature selection achieved through AdaBoost-Stump
- modified linear aggregation G to rule out non-face earlier
 - efficiency achieved through modified linear aggregation

AdaBoost-Stump:
efficient feature selection and aggregation

Fun Time

For a data set of size 9876 that contains $\mathbf{x}_n \in \mathbb{R}^{5566}$, after running AdaBoost-Stump for 1126 iterations, what is the number of distinct features within \mathbf{x} that are effectively used by G ?

- 1 $0 \leq \text{number} \leq 1126$
- 2 $1126 < \text{number} \leq 5566$
- 3 $5566 < \text{number} \leq 9876$
- 4 $9876 < \text{number}$

Fun Time

For a data set of size 9876 that contains $\mathbf{x}_n \in \mathbb{R}^{5566}$, after running AdaBoost-Stump for 1126 iterations, what is the number of distinct features within \mathbf{x} that are effectively used by G ?

- ① $0 \leq \text{number} \leq 1126$
- ② $1126 < \text{number} \leq 5566$
- ③ $5566 < \text{number} \leq 9876$
- ④ $9876 < \text{number}$

Reference Answer: ①

Each decision stump takes only one feature.
So 1126 decision stumps need at most 1126
distinct features.

Summary

- ① Embedding Numerous Features: Kernel Models
- ② Combining Predictive Features: Aggregation Models

Lecture 8: Adaptive Boosting

- Motivation of Boosting
aggregate weak hypotheses for strength
- Diversity by Re-weighting
scale up incorrect, scale down correct
- Adaptive Boosting Algorithm
two heads are better than one, theoretically
- Adaptive Boosting in Action
AdaBoost-Stump useful and efficient

- next: learning conditional aggregation instead of linear one

- ③ Distilling Implicit Features: Extraction Models