Machine Learning
(機器學習)

Lecture 12: Bagging and 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. When Can Machines Learn?
2. Why Can Machines Learn?
3. How Can Machines Learn?
4. How Can Machines Learn Better?
5. Embedding Numerous Features: Kernel Models
6. Combining Predictive Features: Aggregation Models

Lecture 12: Bagging and Boosting

- Uniform Blending
- Linear and Any Blending
- Bagging (Bootstrap Aggregation)
- Motivation of Boosting
- Diversity by Re-weighting
- Adaptive Boosting Algorithm
- Adaptive Boosting in Action

An Aggregation Story

Your T friends $g_1, \cdots, g_T$ predicts whether stock will go up as $g_t(\mathbf{x})$.

You can . . .

- **select** the most trust-worthy friend from their **usual performance**
  —**validation!**
- **mix** the predictions from all your friends **uniformly**
  —let them **vote!**
- **mix** the predictions from all your friends **non-uniformly**
  —let them vote, but **give some more ballots**
- **combine** the predictions **conditionally**
  —if **[t satisfies some condition]** give some ballots to friend t
- ...

aggregation models: **mix** or **combine**
hypotheses (for better performance)

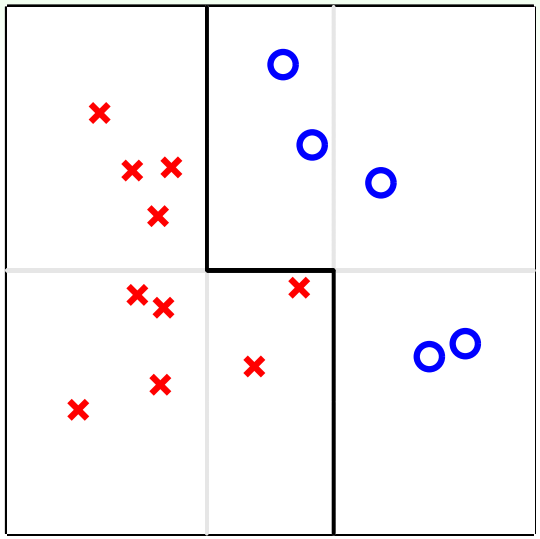
Aggregation with Math Notations

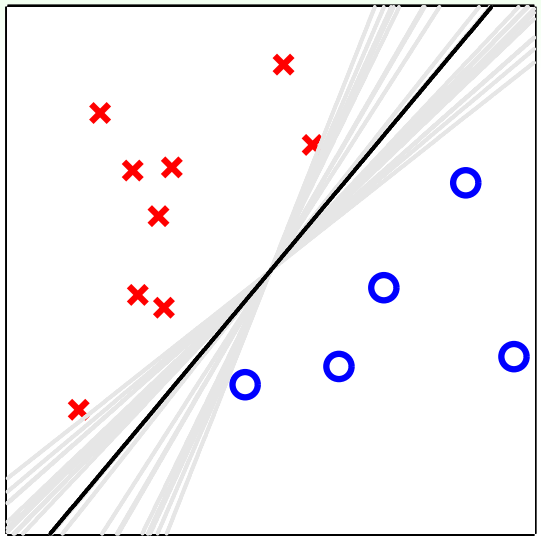
Your T friends $g_1, \cdots, g_T$ predicts whether stock will go up as $g_t(\mathbf{x})$.

- **select** the most trust-worthy friend from their **usual performance**
$$G(\mathbf{x}) = g_{t_*}(\mathbf{x}) \text{ with } t_* = \text{argmin}_{t \in \{1,2,\cdots,T\}} E_{\text{val}}(g_t^-)$$
- **mix** the predictions from all your friends **uniformly**
$$G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} 1 \cdot g_t(\mathbf{x})\right)$$
- **mix** the predictions from all your friends **non-uniformly**
$$G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t \cdot g_t(\mathbf{x})\right) \text{ with } \alpha_t \geq 0$$
  - include **select**: $\alpha_t = [\![E_{\text{val}}(g_t^-) \text{ smallest}]\!]$
  - include **uniformly**: $\alpha_t = 1$
- **combine** the predictions **conditionally**
$$G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} q_t(\mathbf{x}) \cdot g_t(\mathbf{x})\right) \text{ with } q_t(\mathbf{x}) \geq 0$$
  - include **non-uniformly**: $q_t(\mathbf{x}) = \alpha_t$

aggregation models: a **rich family**

Why Might Aggregation Work?





- mix **different weak hypotheses** uniformly —$G(\mathbf{x})$ 'strong'
- aggregation $\Longrightarrow$ **feature transform (?)**

- mix **different random-PLA hypotheses** uniformly —$G(\mathbf{x})$ 'moderate'
- aggregation $\Longrightarrow$ **regularization (?)**

proper aggregation $\Longrightarrow$ **better performance**

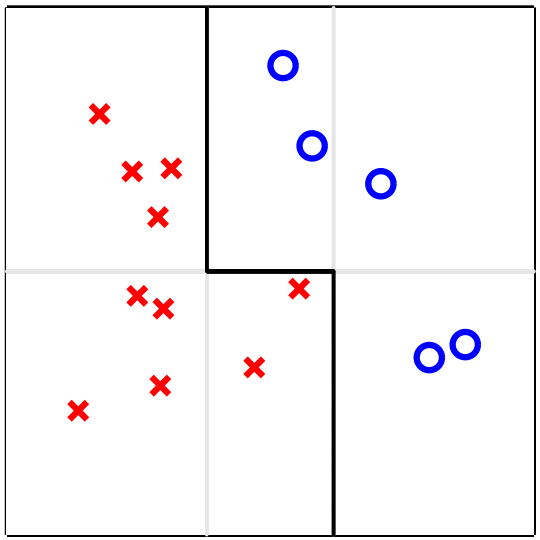
Uniform Blending (Voting) for Classification

uniform blending: known g_t, each with 1 ballot

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

- same g_t (autocracy): as good as one single g_t
- very different g_t (**diversity** + **democracy**): majority can **correct** minority
- similar results with uniform voting for multiclass

$$G(\mathbf{x}) = \underset{1 \le k \le K}{\text{argmax}} \sum_{t=1}^{T} [\![g_t(\mathbf{x}) = k]\!]$$



how about **regression**?

Uniform Blending for Regression

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

- same g_t (autocracy):
  as good as one single g_t
- very different g_t (**diversity** + **democracy**):
  some $g_t(\mathbf{x}) > f(\mathbf{x})$, some $g_t(\mathbf{x}) < f(\mathbf{x})$
  $\Longrightarrow$ average **could be** more accurate than individual

diverse hypotheses:

even simple uniform blending
can be better than any single hypothesis

Theoretical Analysis of Uniform Blending

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

$$\begin{aligned}
\text{avg}\left((g_t(\mathbf{x}) - f(\mathbf{x}))^2\right) &= \text{avg}\left(g_t^2 - 2g_t f + f^2\right) \\
&= \text{avg}\left(g_t^2\right) - 2Gf + f^2 \\
&= \text{avg}\left(g_t^2\right) - G^2 + (G - f)^2 \\
&= \text{avg}\left(g_t^2\right) - 2G^2 + G^2 + (G - f)^2 \\
&= \text{avg}\left(g_t^2 - 2g_t G + G^2\right) + (G - f)^2 \\
&= \text{avg}\left((g_t - G)^2\right) + (G - f)^2
\end{aligned}$$

$$\begin{aligned}
\text{avg}\left(E_{\text{out}}(g_t)\right) &= \text{avg}\left(\mathcal{E}(g_t - G)^2\right) + E_{\text{out}}(G) \\
&\geq \qquad\qquad\qquad + E_{\text{out}}(G)
\end{aligned}$$

Questions?

Linear Blending

linear blending: known g_t, each to be given α_t ballot

$$G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t \cdot g_t(\mathbf{x})\right) \text{ with } \alpha_t \geq 0$$

computing 'good' α_t : $\min_{\alpha_t \geq 0} E_{\text{in}}(\boldsymbol{\alpha})$

linear blending for regression

$$\min_{\alpha_t \geq 0} \frac{1}{N} \sum_{n=1}^{N} \left(y_n - \sum_{t=1}^{T} \alpha_t g_t(\mathbf{x}_n)\right)^2$$

LinReg + transformation

$$\min_{w_i} \frac{1}{N} \sum_{n=1}^{N} \left(y_n - \sum_{i=1}^{\tilde{d}} w_i \phi_i(\mathbf{x}_n)\right)^2$$

linear blending = LinModel + hypotheses as transform + constraints

Constraint on α_t

linear blending = LinModel + hypotheses as transform + constraints:

$$\min_{\alpha_t \geq 0} \quad \frac{1}{N} \sum_{n=1}^{N} \text{err}\left(y_n, \sum_{t=1}^{T} \alpha_t g_t(\mathbf{x}_n)\right)$$

linear blending for binary classification

$$\text{if } \alpha_t < 0 \implies \alpha_t g_t(\mathbf{x}) = |\alpha_t| \left(-g_t(\mathbf{x})\right)$$

- negative α_t for $g_t \equiv$ positive $|\alpha_t|$ for $-g_t$
- **if you have a stock up/down classifier with 99% error, tell me! :-)**

in practice, often
linear blending = LinModel + hypotheses as transform ~~+ constraints~~

Linear Blending versus Selection

in practice, often

$$g_1 \in \mathcal{H}_1, g_2 \in \mathcal{H}_2, \ldots, g_T \in \mathcal{H}_T$$

by minimum E_{in}

- recall: **selection by minimum** E_{in}
  —best of best, paying $d_{\text{VC}}\left(\bigcup_{t=1}^{T} \mathcal{H}_t\right)$
- recall: linear blending includes selection as special case
  —by setting $\alpha_t = [\![E_{\text{val}}(g_t^-) \text{ smallest}]\!]$
- complexity price of linear blending with E_{in} (aggregation of best):
  $\geq d_{\text{VC}}\left(\bigcup_{t=1}^{T} \mathcal{H}_t\right)$

like selection, blending practically done with
(E_{val} instead of E_{in}) + (g_t^- from minimum E_{train})

Any Blending

Given $g_1^-, g_2^-, \ldots, g_T^-$ from $\mathcal{D}_{\text{train}}$, transform $(\mathbf{x}_n, y_n)$ in $\mathcal{D}_{\text{val}}$ to $(\mathbf{z}_n = \Phi^-(\mathbf{x}_n), y_n)$, where $\Phi^-(\mathbf{x}) = (g_1^-(\mathbf{x}), \ldots, g_T^-(\mathbf{x}))$

Linear Blending

1. compute α
   $= \text{LinearModel}\Big(\{(\mathbf{z}_n, y_n)\}\Big)$
2. return $G_{\text{LINB}}(\mathbf{x}) =$
   $\text{LinearHypothesis}_\alpha\,(\Phi(\mathbf{x}))$,

Any Blending (Stacking)

1. compute $\tilde{g}$
   $= \textbf{AnyModel}\Big(\{(\mathbf{z}_n, y_n)\}\Big)$
2. return $G_{\text{ANYB}}(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x}))$,

where $\Phi(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_T(\mathbf{x}))$

any blending:

- **powerful**, achieves conditional blending
- but **danger of overfitting**, as always **:-(**

Blending in Practice

(Chen et al., A linear ensemble of individual and blended models for music rating prediction, 2012)

KDDCup 2011 Track 1: World Champion Solution by NTU

- validation set blending: a special any blending model

  E_{test} (squared): 519.45 $\Longrightarrow$ 456.24

  —helped **secure the lead** in last two weeks

- test set blending: linear blending using $\tilde{E}_{\text{test}}$

  E_{test} (squared): 456.24 $\Longrightarrow$ 442.06

  —helped **turn the tables** in last hour

blending 'useful' in practice,
despite the computational burden

Questions?

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	?
non-uniform	linear	?
conditional	stacking	?

learning g_t for uniform aggregation: diversity important

- diversity by different models: $g_1 \in \mathcal{H}_1, g_2 \in \mathcal{H}_2, \ldots, g_T \in \mathcal{H}_T$
- diversity by different parameters: GD with $\eta = 0.001, 0.01, \ldots, 10$
- diversity by algorithmic randomness:
  random PLA with different random seeds
- diversity by data randomness:
  within-cross-validation hypotheses g_v^-

next: diversity by data randomness without g^-

Bootstrap Aggregation

bootstrapping

bootstrap sample $\tilde{\mathcal{D}}_t$: re-sample N examples from $\mathcal{D}$ **uniformly with replacement**—can also use arbitrary N' instead of original N

- $\tilde{\mathcal{D}}_t \neq \mathcal{D}$ in general (diverse)
- $\tilde{\mathcal{D}}_t$ 'acts like' $\mathcal{D}$ statistically (g_t 'acts like' good g)
- each g_t similarly good/bad (uniform)

bootstrap aggregation

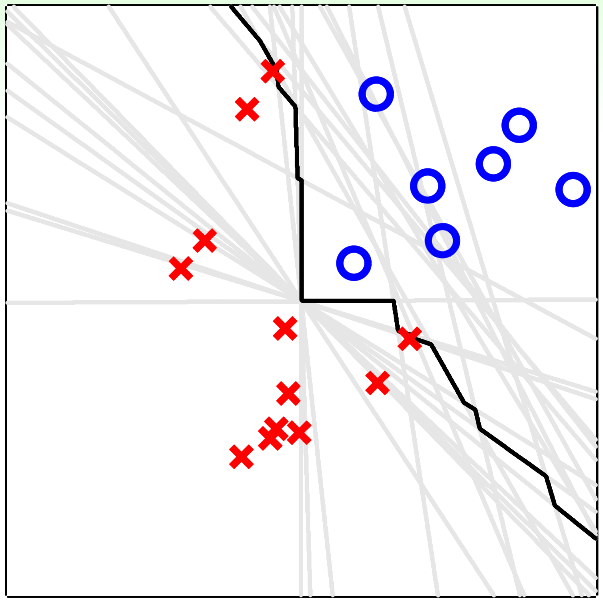
iterative process that for $t = 1, 2, \ldots, T$

1. request size-N' data $\tilde{\mathcal{D}}_t$ from bootstrapping
2. obtain g_t by $\mathcal{A}(\tilde{\mathcal{D}}_t)$

$$G = \text{Uniform}(\{g_t\})$$

bootstrap aggregation (BAGging):
a simple **meta algorithm**

Bagging on Top of (Modified) PLA in Action



$T_{\text{BAG}} = 25$

- very diverse g_t from bagging
- proper **non-linear** boundary after aggregating binary classifiers

bagging works reasonably well **if base algorithm sensitive to data randomness**

Questions?

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 Flicker (**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 Flicker (**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 McQueen

https://flic.kr/p/pZv1Mf

Crystal

https://flic.kr/p/kaPYp

jfh686

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

skyseeker

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

Janet Hudson

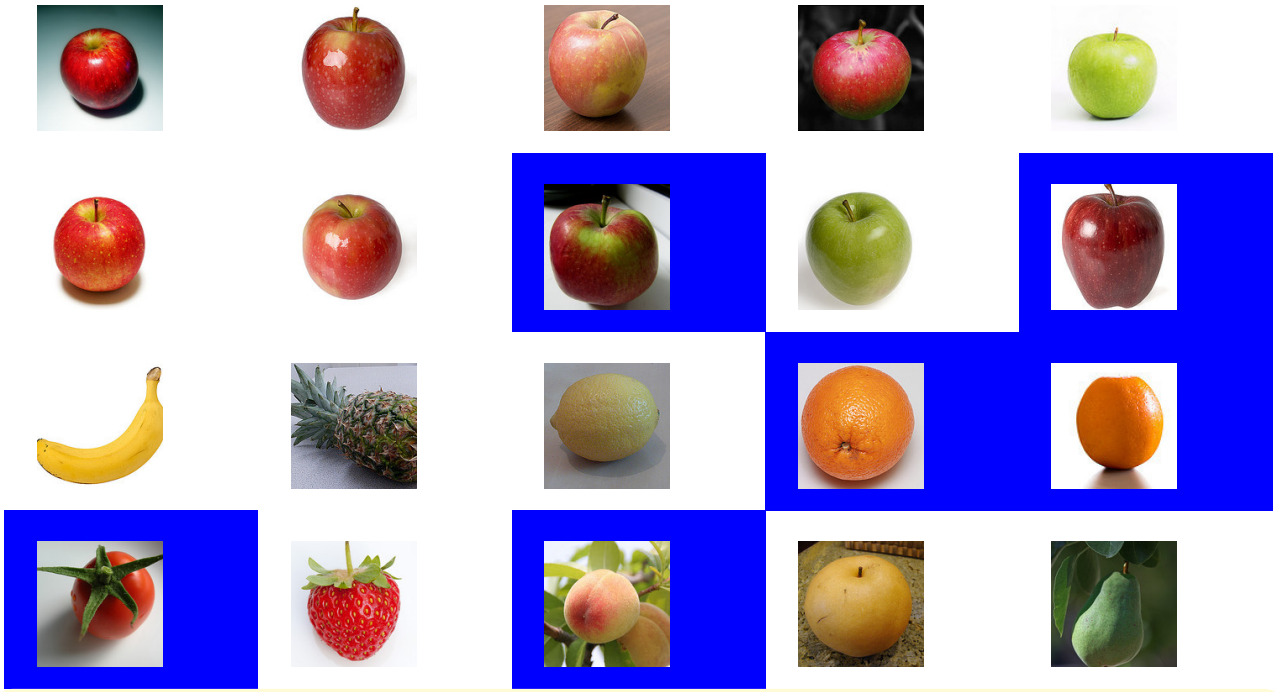
https://flic.kr/p/7QDBbm

Rennett Stowe

https://flic.kr/p/agmnrk

Our Fruit Class Begins

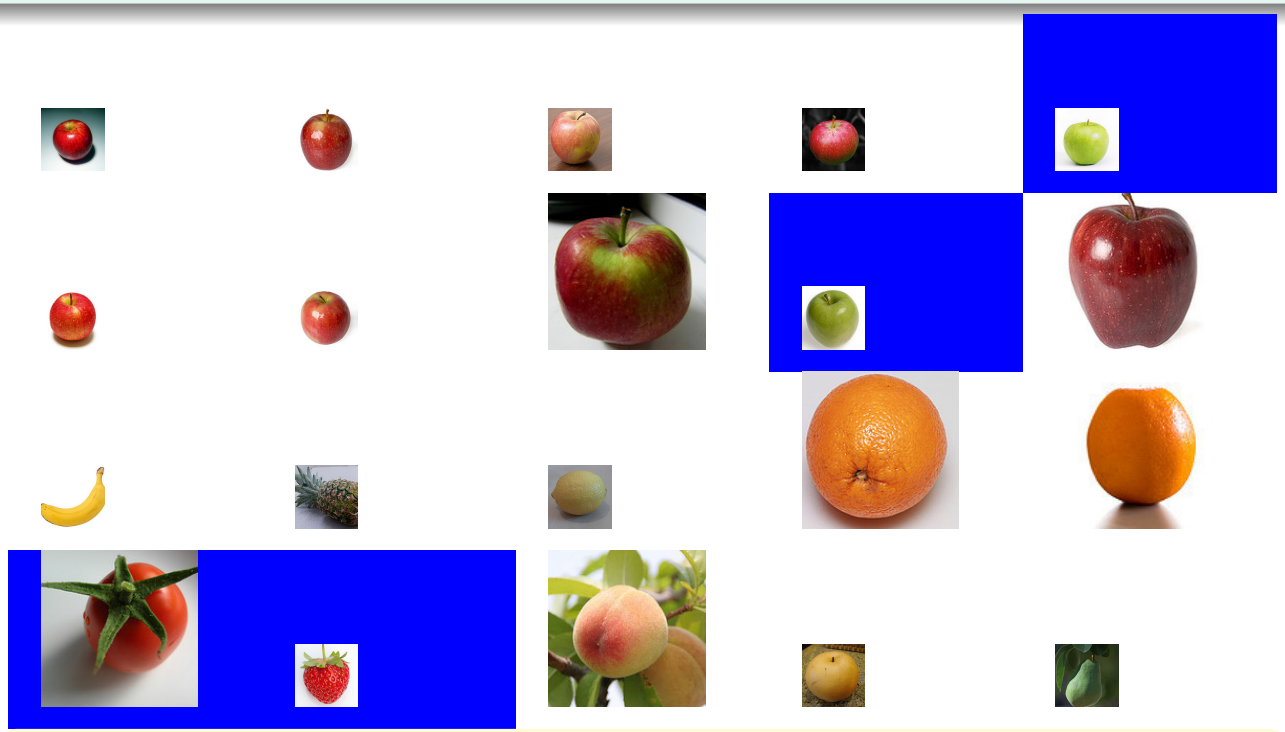
- Teacher: Please look at the pictures of apples and non-apples below. Based on those pictures, how would you describe an apple? Michael?
- Michael: I think apples are **circular**.



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

Our Fruit Class Continues

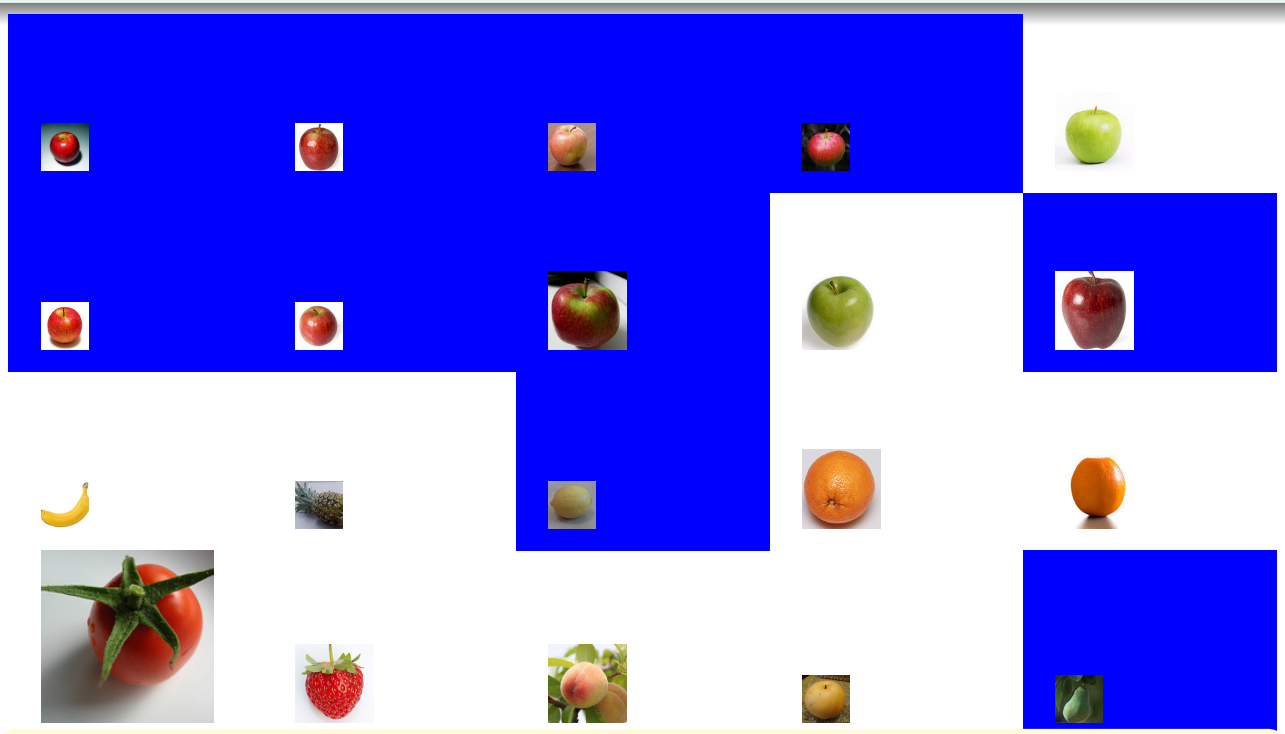
- Teacher: Being circular is a good feature for the apples. However, if you only say circular, you could make several mistakes. What else can we say for an apple? Tina?
- Tina: It looks like apples are **red**.



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

Our Fruit Class Continues More

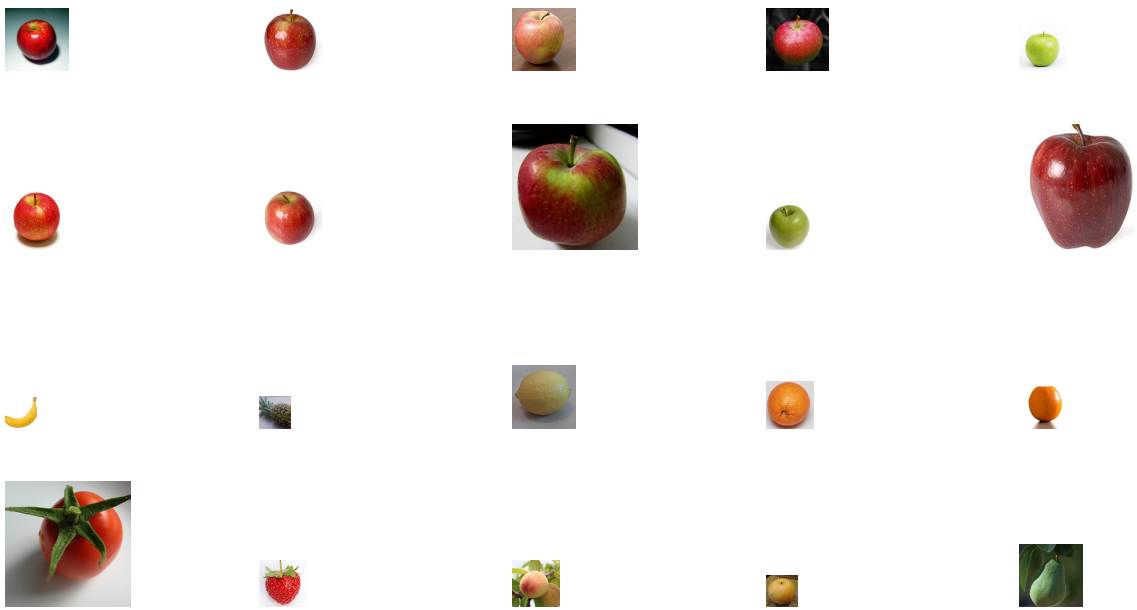
- Teacher: Yes. Many apples are red. However, you could still make mistakes based on circular and red. Do you have any other suggestions, Joey?
- Joey: Apples could also be **green**.



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

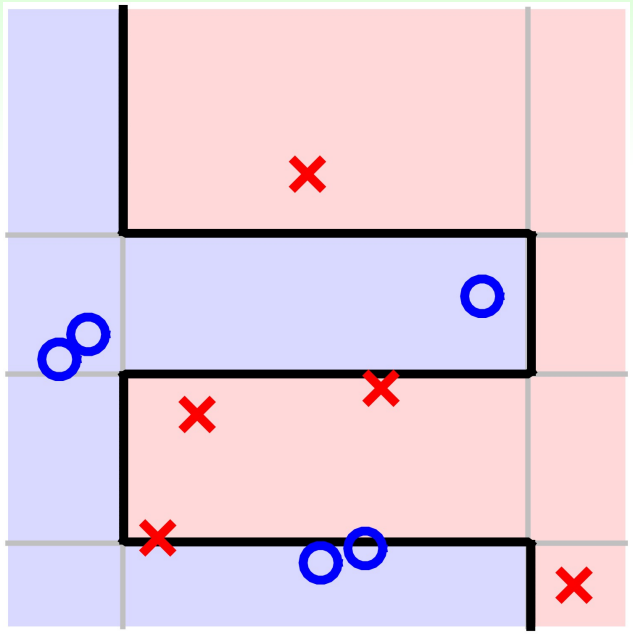
Our Fruit Class Ends

- 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

Questions?

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 $\mathcal{D}$

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

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

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)

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

e.g. logistic regression

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

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

example-weighted learning:
modified error function, optimized by **sampling**/re-derivation/...

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 $\mathbf{u}^{(t+1)} \implies g_t$-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**

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

'Optimal' Re-weighting

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

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

- 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

(total $u_n^{(t)}$ of incorrect) = 1126 ; (weighted incorrect rate) = $\frac{1126}{7337}$	(total $u_n^{(t)}$ of correct) = 6211 ; (weighted correct rate) = $\frac{6211}{7337}$
incorrect: $u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot 6211$	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 $\propto \epsilon_t$

Questions?

Scaling Factor

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

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

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

$$\begin{array}{rcrcl} \text{incorrect} & \leftarrow & \text{incorrect} & \cdot & \blacklozenge_t \\ \text{correct} & \leftarrow & \text{correct} & / & \blacklozenge_t \end{array}$$

- **equivalent** to optimal re-weighting
- $\blacklozenge_t \geq 1$ 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 $\blacklozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$,
   where ϵ_t = weighted error (incorrect) rate of g_t

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

- want g_1 'best' for E_{in}: $u_n^{(1)} = \frac{1}{N}$
- $G(\mathbf{x})$:
  - 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

$\mathbf{u}^{(1)} = [\frac{1}{N}, \frac{1}{N}, \cdots, \frac{1}{N}]$
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 $\blacklozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$, where ...
3. compute $\alpha_t = \ln(\blacklozenge_t)$

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

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

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

Adaptive Boosting (AdaBoost) Algorithm

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

$$[\![y_n \neq g_t(\mathbf{x}_n)]\!] \text{ (incorrect examples)}: \quad u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot \blacklozenge_t$$
$$[\![y_n = g_t(\mathbf{x}_n)]\!] \text{ (correct examples)}: \quad u_n^{(t+1)} \leftarrow u_n^{(t)} / \blacklozenge_t$$

   where $\blacklozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$ and $\epsilon_t = \frac{\sum_{n=1}^{N} u_n^{(t)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]}{\sum_{n=1}^{N} u_n^{(t)}}$

3. compute $\alpha_t = \ln(\blacklozenge_t)$

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)

Questions?

Decision Stump

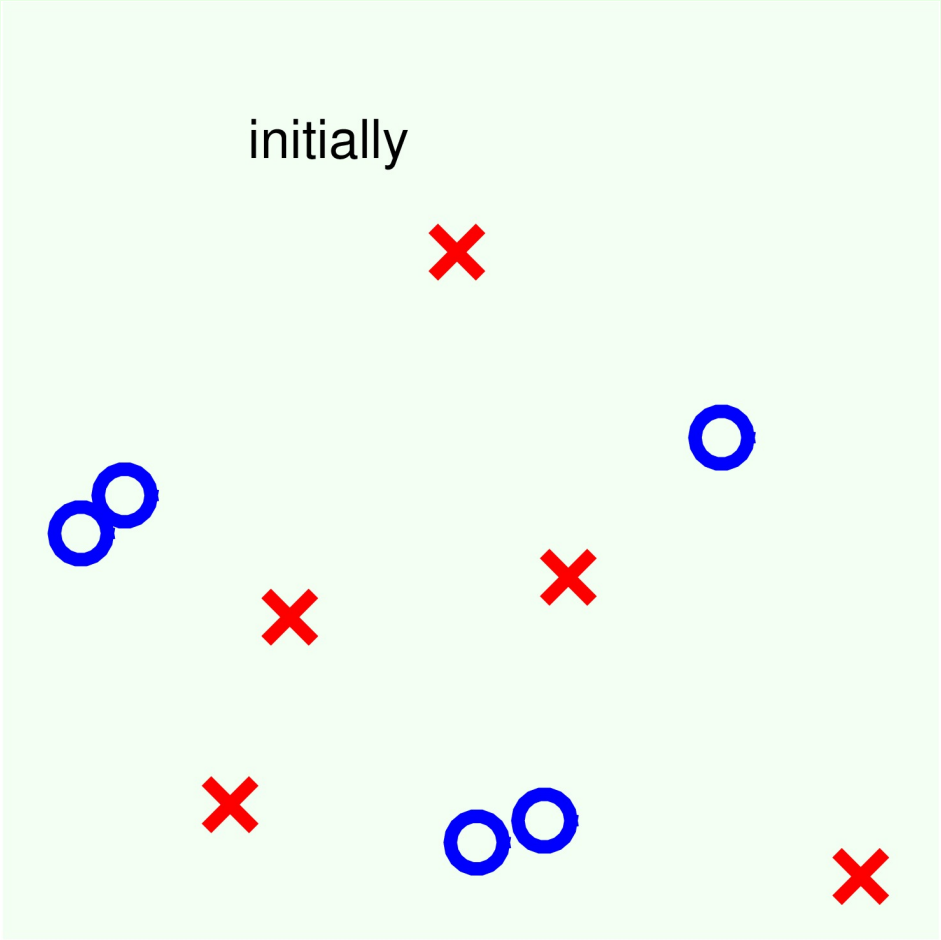
want: a '**weak**' base learning algorithm $\mathcal{A}$
that minimizes $E_{\text{in}}^{\mathbf{u}}(h) = \frac{1}{N}\sum_{n=1}^{N} u_n \cdot [\![y_n \neq h(\mathbf{x}_n)]\!]$ **a little bit**

a popular choice: decision stump

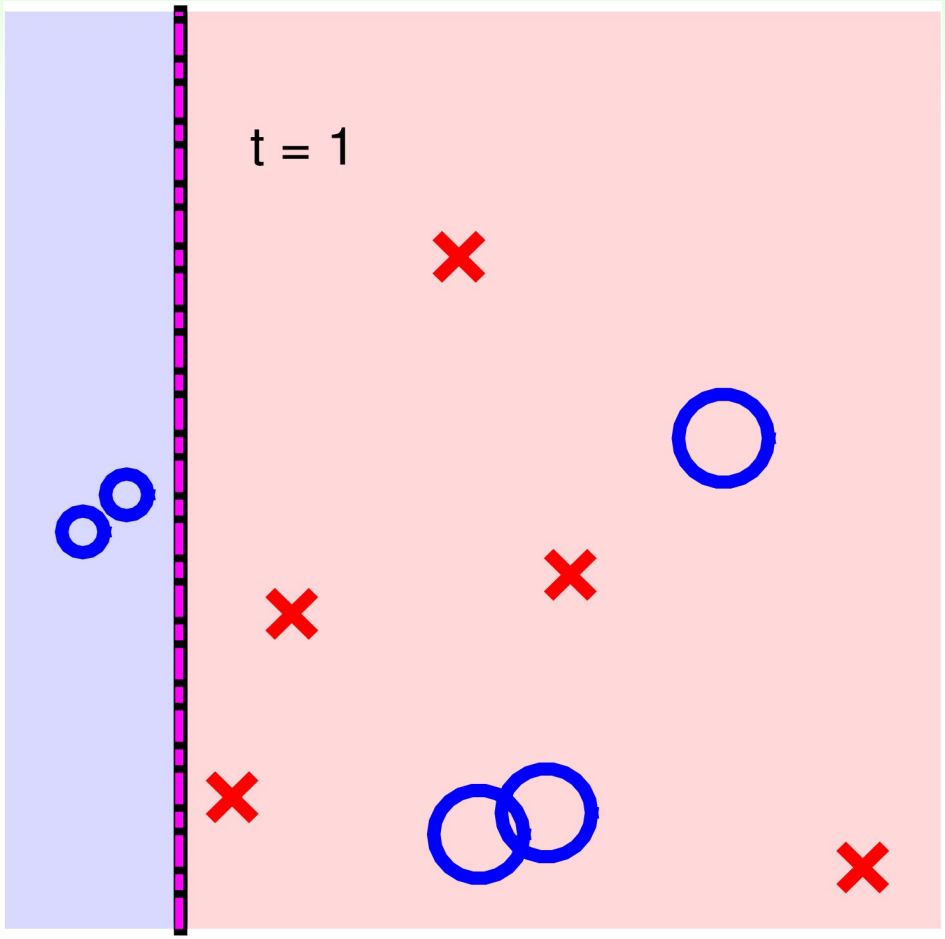
- $$h_{s,i,\theta}(\mathbf{x}) = s \cdot \text{sign}(x_i - \theta)$$
- positive and negative rays on some feature: three parameters (feature i, threshold θ, direction s)
- physical meaning: vertical/horizontal lines in 2D
- efficient to optimize: $O(d \cdot N \log N)$ time

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

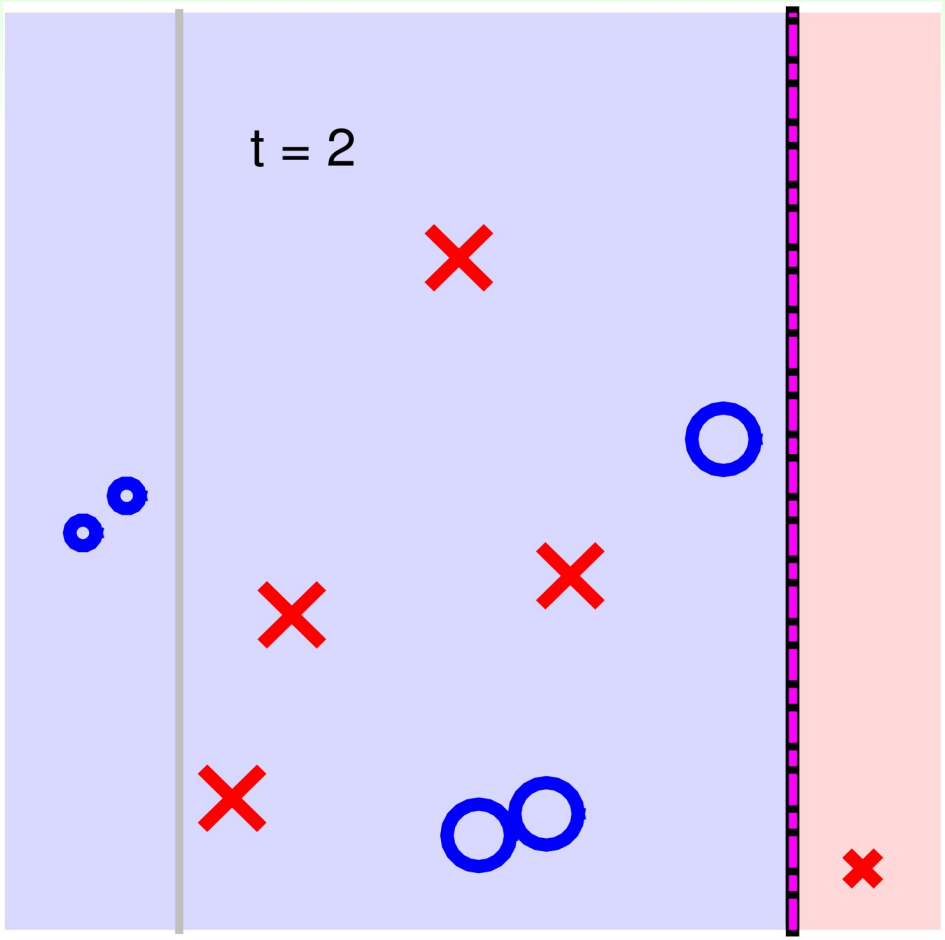
A Simple Data Set



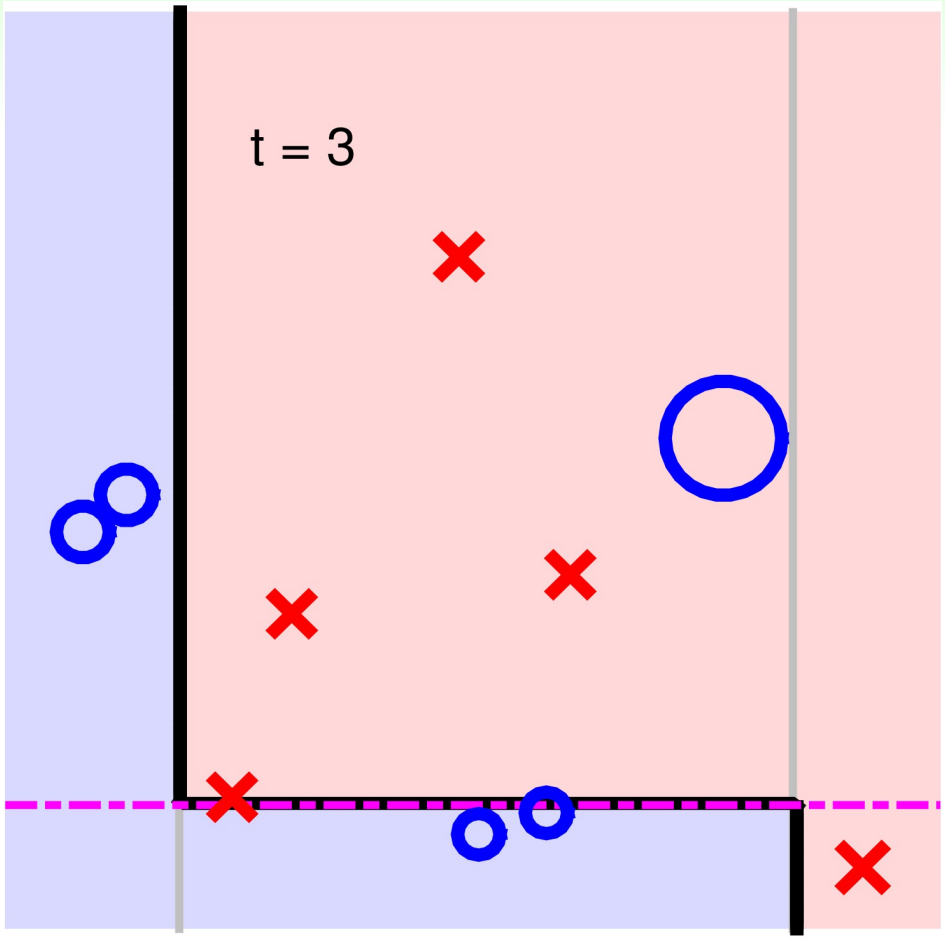
A Simple Data Set



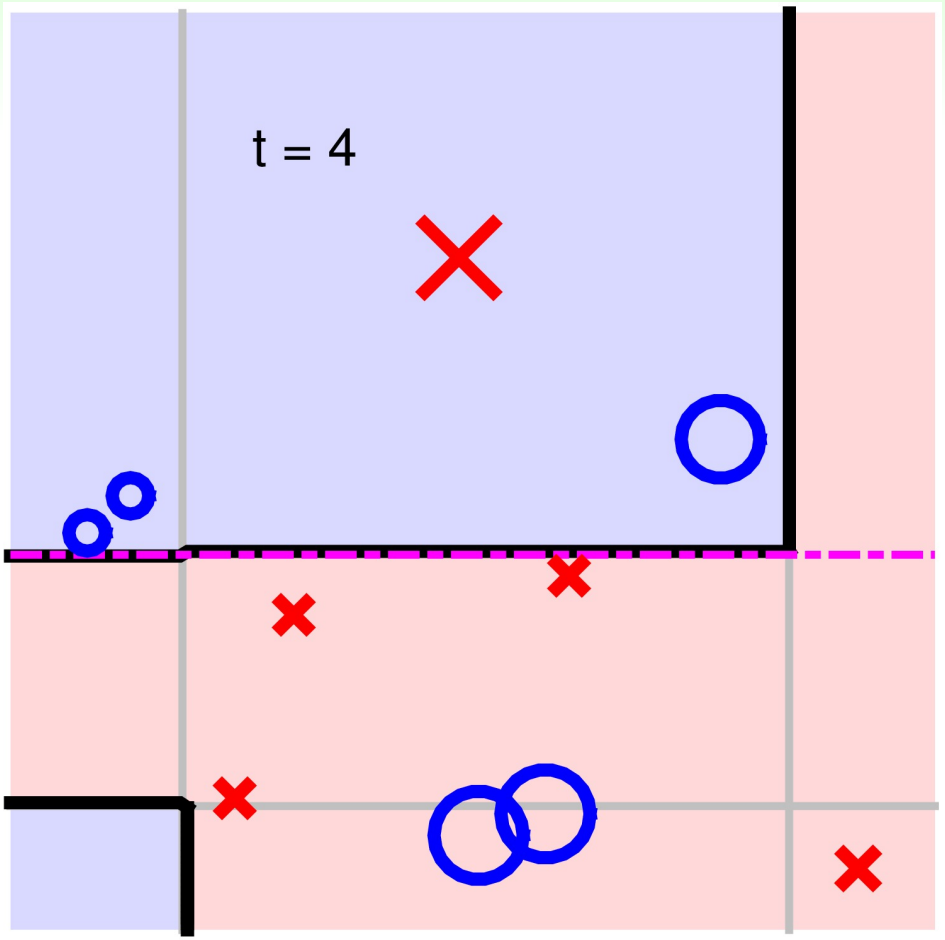
A Simple Data Set



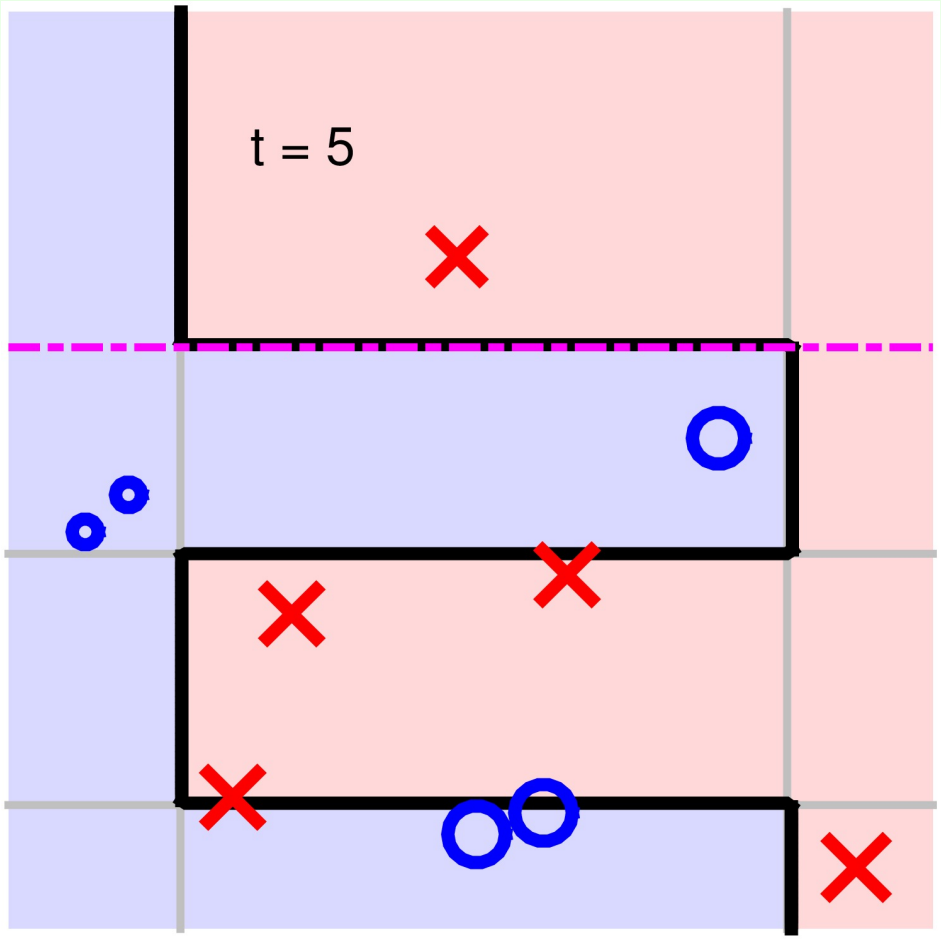
A Simple Data Set



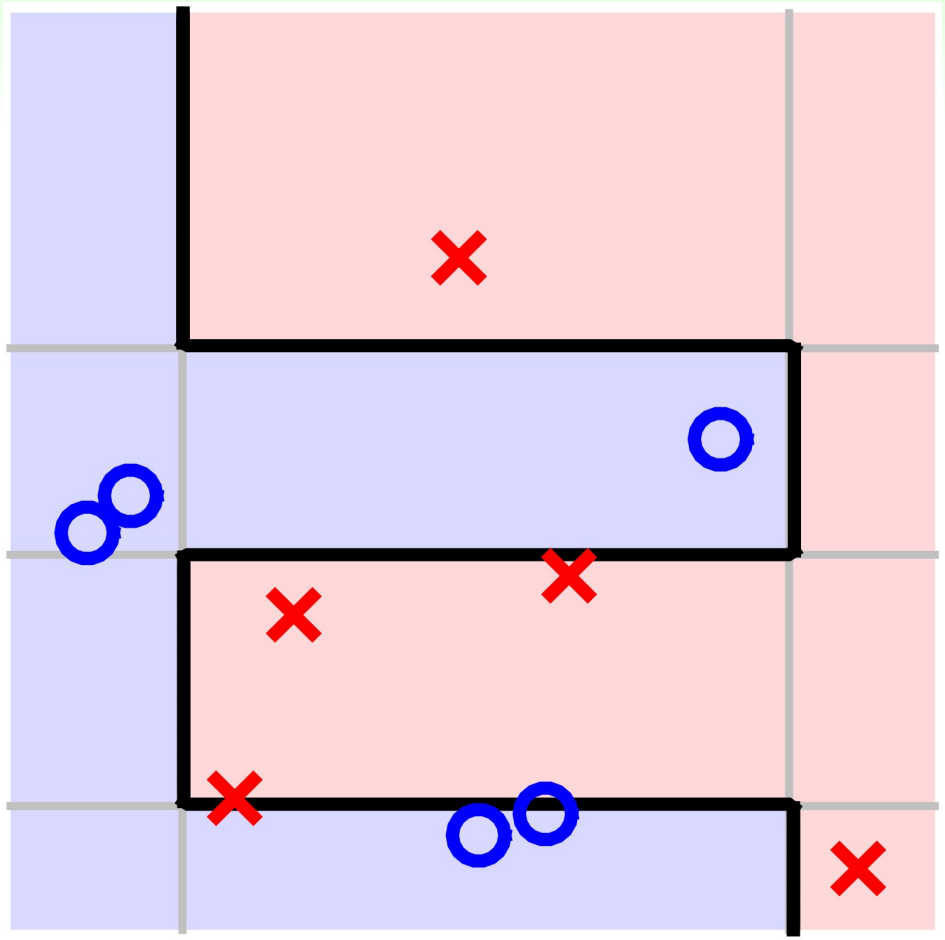
A Simple Data Set



A Simple Data Set



A Simple Data Set



'Teacher'-like algorithm works!

Putting Everything Together

Gradient Boosted Decision Tree (GBDT)

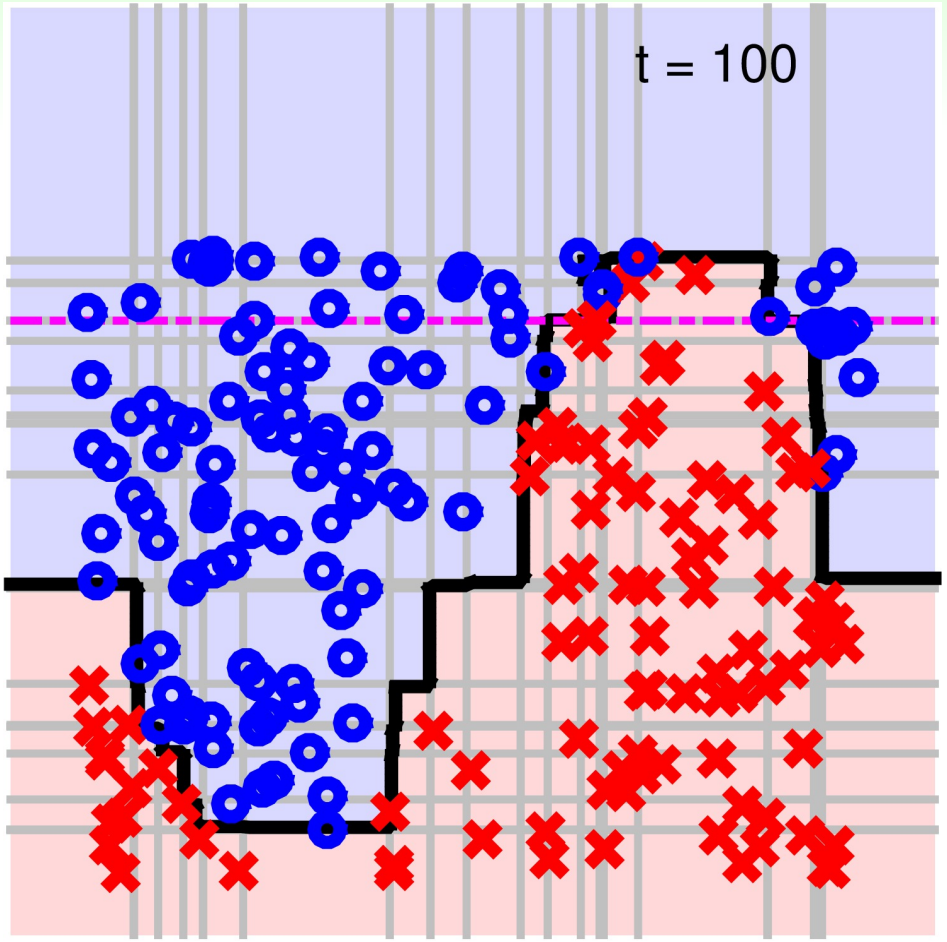
$s_1 = s_2 = \ldots = s_N = 0$
for $t = 1, 2, \ldots, T$

1. obtain g_t by $\mathcal{A}(\{(\mathbf{x}_n, y_n - s_n)\})$ where $\mathcal{A}$ is a (squared-error) regression algorithm
   —**how about sampled and pruned C&RT?**
2. compute $\alpha_t = \text{OneVarLinearRegression}(\{(g_t(\mathbf{x}_n), y_n - s_n)\})$
3. update $s_n \leftarrow s_n + \alpha_t g_t(\mathbf{x}_n)$

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

GBDT: 'regression sibling' of AdaBoost-DTree
—**popular in practice**

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

Questions?

Summary

1. Combining Predictive Features: Aggregation Models

Lecture 12: Bagging and Boosting

- Uniform Blending
  diverse hypotheses, 'one vote, one value'
- Linear and Any Blending

two-level learning with hypotheses as transform

- Bagging (Bootstrap Aggregation)
  bootstrapping for diverse hypotheses
- 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