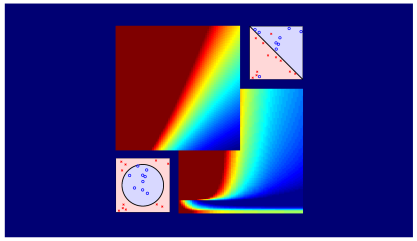
Machine Learning Techniques
(機器學習技巧)



Lecture 7: Blending and Bagging

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 7: Blending and Bagging

- Motivation of Aggregation
- Uniform Blending
- Linear and Any Blending
- Bagging

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 **[condition t true]** 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**

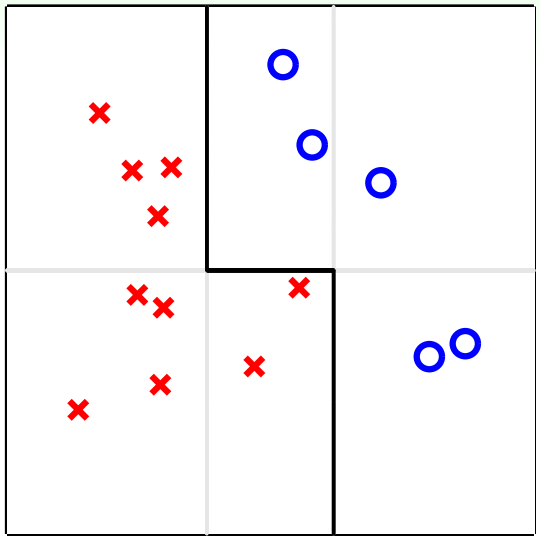
Recall: Selection by Validation

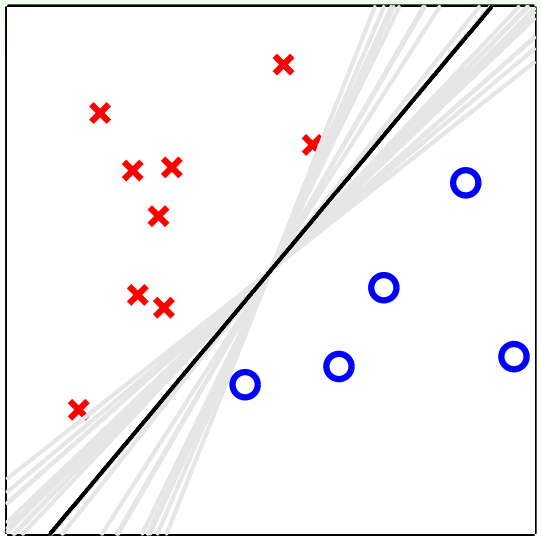
$$G(\mathbf{x}) = g_{t_*}(\mathbf{x}) \text{ with } t_* = \underset{t \in \{1,2,\cdots,T\}}{\text{argmin}} \; E_{\text{val}}(g_t)$$

- **simple** and popular
- can also use E_{in} instead of E_{val} (with **complexity price on** d_{VC})
- need **one strong** g_t to guarantee small E_{val} (and small E_{out})

selection:
rely on one strong hypothesis
aggregation:
can we do better with many (possibly weaker) hypotheses?

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**

Fun Time

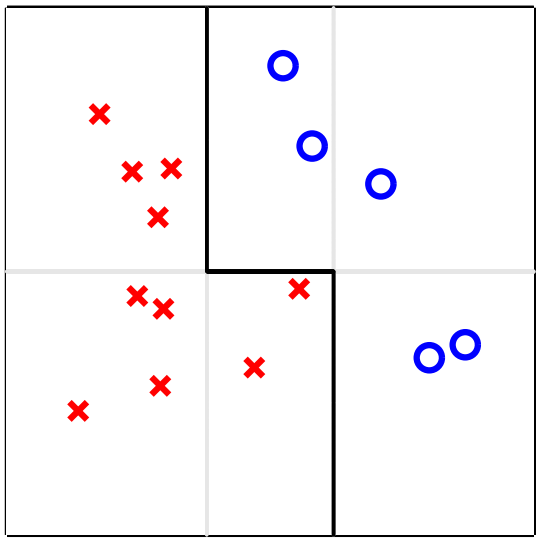
Uniform Blending (Voting) for Classification

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 one

Theoretical Analysis of Uniform Blending

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

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

$$\text{avg}\left(E_{\text{out}}(g_t)\right) = \text{avg}\left(\mathcal{E}(g_t - G)^2\right) + E_{\text{out}}(G)$$

Some Special g_t

consider a **virtual** iterative process that for $t = 1, 2, \ldots, T$

1. request size-N data $\mathcal{D}_t$ from P^N (i.i.d.)
2. obtain g_t by $\mathcal{A}(\mathcal{D}_t)$

$$\bar{g} = \lim_{T\to\infty} G = \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} g_t = \mathop{\mathcal{E}}_{\mathcal{D}} \mathcal{A}(\mathcal{D})$$

$$\text{avg}\left(E_{\text{out}}(g_t)\right) = \text{avg}\left(\mathcal{E}(g_t - \bar{g})^2\right) + E_{\text{out}}(\bar{g})$$

expected performance of $\mathcal{A}$ = expected deviation to consensus + performance of consensus

- performance of consensus: called **bias**
- expected deviation to consensus: called **variance**

uniform blending:
reduces **variance** for stabler performance

Fun Time

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):
  $\gg d_{\text{VC}}\left(\bigcup_{t=1}^{T} \mathcal{H}_t\right)$

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

Any Blending

Linear Blending

Given $g_1, g_2, \ldots, g_T$

1. transform $(\mathbf{x}_n, y_n)$ in $\mathcal{D}_{\text{val}}$ to $(\mathbf{z}_n = \mathbf{\Phi}(\mathbf{x}_n), y_n)$, where $\mathbf{\Phi}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_T(\mathbf{x}))$
2. compute $\alpha = \text{Lin}\Big(\{(\mathbf{z}_n, y_n)\}\Big)$

return $G_{\text{LINB}}(\mathbf{x}) = \text{LinH}(\alpha^T \mathbf{\Phi}(\mathbf{x}))$

Any Blending (Stacking)

Given $g_1, g_2, \ldots, g_T$

1. transform $(\mathbf{x}_n, y_n)$ in $\mathcal{D}_{\text{val}}$ to $(\mathbf{z}_n = \mathbf{\Phi}(\mathbf{x}_n), y_n)$, where $\mathbf{\Phi}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_T(\mathbf{x}))$
2. compute $\tilde{g} = \textbf{Any}\Big(\{(\mathbf{z}_n, y_n)\}\Big)$

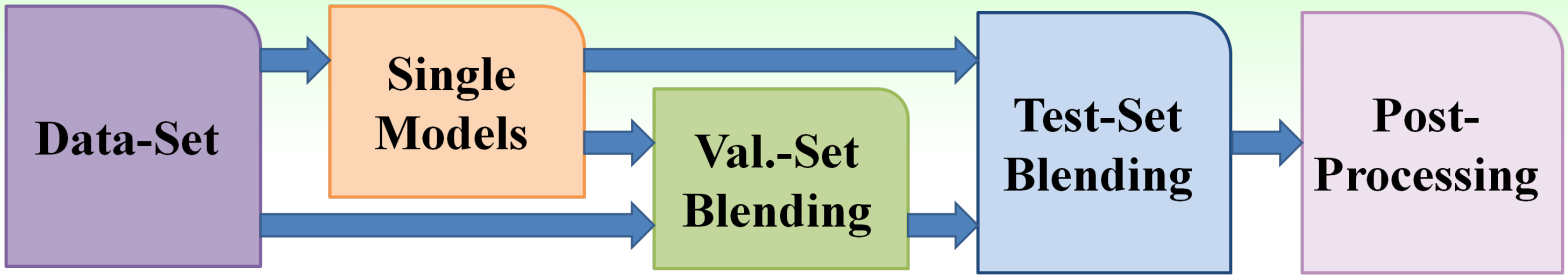
return $G_{\text{ANYB}}(\mathbf{x}) = \tilde{g}(\mathbf{\Phi}(\mathbf{x}))$

if **Any**Model = quadratic polynomial:

$$G_{\text{ANYB}}(\mathbf{x}) = \sum_{t=1}^{T} \underbrace{\left(\alpha_t + \sum_{\tau=1}^{T} \alpha_{\tau,t} g_\tau(\mathbf{x})\right)}_{q(\mathbf{x})} \cdot g_t(\mathbf{x}) \text{ —conditional aggregation}$$

danger: overfitting with **any** blending!

Blending in Practice



KDDCup 2012 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**

Fun Time

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^-

Revisit of Bias-Variance

expected performance of $\mathcal{A}$ $=$ expected deviation to consensus
$+$performance of consensus

consensus $\bar{g}$ $=$ expected g_t from $\mathcal{D}_t \sim P^N$

- consensus more stable than direct $\mathcal{A}(\mathcal{D})$,
  but comes from many more $\mathcal{D}_t$ than the $\mathcal{D}$ on hand
- want: approximate $\bar{g}$ by
  - finite (large) T
  - approximate $g_t = \mathcal{A}(\mathcal{D}_t)$ from $\mathcal{D}_t \sim P^N$ using only $\mathcal{D}$

bootstrapping: a statistical tool that
re-samples from $\mathcal{D}$ to 'simulate' $\mathcal{D}_t$

Bootstrap Aggregation

bootstrapping

bootstrap sample $\tilde{D}_t$: re-sample N examples from $\mathcal{D}$ **with replacement**

virtual aggregation

consider a **virtual** iterative process that for $t = 1, 2, \ldots, T$

1. request size-N data $\mathcal{D}_t$ from P^N (i.i.d.)
2. obtain g_t by $\mathcal{A}(\mathcal{D}_t)$

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

bootstrap aggregation

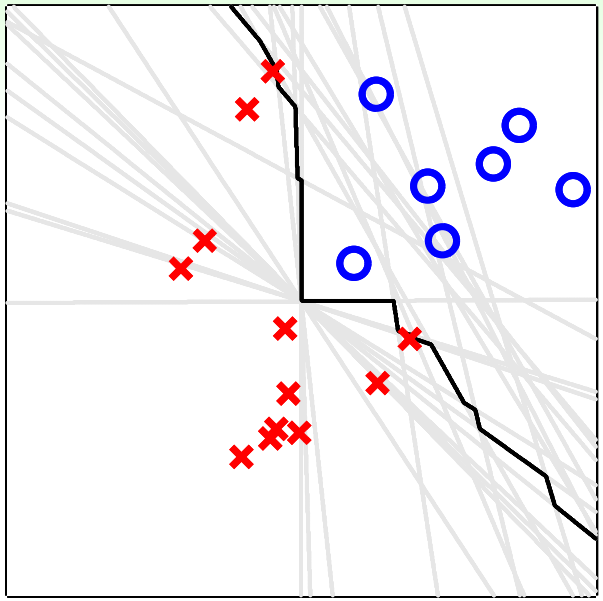
consider a **physical** iterative process that for $t = 1, 2, \ldots, T$

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**
on top of **base algorithm** $\mathcal{A}$

Bagging Pocket in Action



$T_{\text{POCKET}} = 1000$; $T_{\text{BAG}} = 25$

- 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**

Fun Time

Summary

Lecture 7: Blending and Bagging

- Motivation of Aggregation
  strong and/or moderate
- Uniform Blending
  one hypothesis, one vote, one value
- Linear and Any Blending
  learning with hypotheses as transform
- Bagging
  bootstrapping for diverse g_t