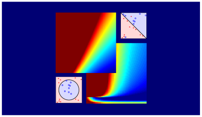
Machine Learning Foundations
(機器學習基石)



Lecture 15: Validation

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**?

Lecture 14: Regularization

minimizes **augmented error**, where the added **regularizer** effectively **limits model complexity**

Lecture 15: Validation

- Model Selection Problem
- Validation
- Leave-One-Out Cross Validation
- V-Fold Cross Validation

So Many Models Learned

Even Just for Binary Classification . . .

$\mathcal{A} \in \{$ PLA, pocket, linear regression, logistic regression$\}$

$\times$

$T \in \{ 100, 1000, 10000\}$

$\times$

$\eta \in \{ 1, 0.01, 0.0001\}$

$\times$

$\mathbf{\Phi} \in \{$ linear, quadratic, poly-10, Legendre-poly-10$\}$

$\times$

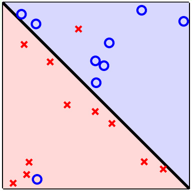
$\Omega(\mathbf{w}) \in \{$ L2 regularizer, L1 regularizer, symmetry regularizer$\}$

$\times$

$\lambda \in \{ 0, 0.01, 1 \}$

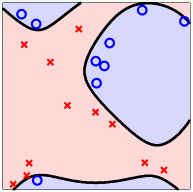
in addition to your **favorite** combination, may need to try other combinations to get a good g

Model Selection Problem



$\mathcal{H}_1$

which one do you prefer? :-)

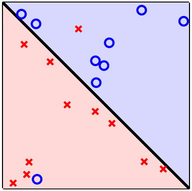


$\mathcal{H}_2$

- given: M models $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_M$, each with corresponding algorithm $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_M$
- goal: select $\mathcal{H}_{m^*}$ such that $g_{m^*} = \mathcal{A}_{m^*}(\mathcal{D})$ is of low $E_{\text{out}}(g_{m^*})$
- unknown E_{out} due to unknown $P(\mathbf{x})$ & $P(y|\mathbf{x})$, as always **:-)**
- arguably the **most important** practical problem of ML

how to select? **visually?**
—no, remember Lecture 12? :-)

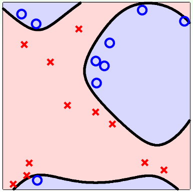
Model Selection by Best E_{in}



$\mathcal{H}_1$

select by best E_{in}?

$$m^* = \underset{1 \le m \le M}{\text{argmin}} (E_m = E_{\text{in}}(\mathcal{A}_m(\mathcal{D})))$$

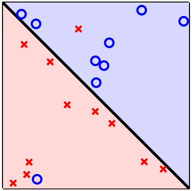


$\mathcal{H}_2$

- $\mathbf{\Phi}_{1126}$ always more preferred over $\mathbf{\Phi}_1$;
  $\lambda = 0$ always more preferred over $\lambda = 0.1$—**overfitting?**
- if $\mathcal{A}_1$ minimizes E_{in} over $\mathcal{H}_1$ and $\mathcal{A}_2$ minimizes E_{in} over $\mathcal{H}_2$,
  $\Longrightarrow g_{m^*}$ achieves minimal E_{in} over $\mathcal{H}_1 \cup \mathcal{H}_2$
  $\Longrightarrow$ 'model selection + learning' pays $d_{\text{VC}}(\mathcal{H}_1 \cup \mathcal{H}_2)$
  —**bad generalization?**

selecting by E_{in} is **dangerous**

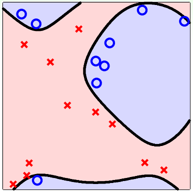
Model Selection by Best E_{test}



$\mathcal{H}_1$

select by best E_{test}, which is evaluated on a fresh $\mathcal{D}_{\text{test}}$?

$$m^* = \underset{1 \le m \le M}{\operatorname{argmin}} \left(E_m = E_{\text{test}}(\mathcal{A}_m(\mathcal{D})) \right)$$



$\mathcal{H}_2$

- generalization guarantee (finite-bin Hoeffding):

$$E_{\text{out}}(g_{m^*}) \le E_{\text{test}}(g_{m^*}) + O\left(\sqrt{\frac{\log M}{N_{\text{test}}}}\right)$$

—**yes! strong guarantee :-)**

- but where is $\mathcal{D}_{\text{test}}$?—**your boss's safe, maybe? :-(**

selecting by E_{test} is **infeasible** and **cheating**

Comparison between E_{in} and E_{test}

in-sample error E_{in}

- calculated from $\mathcal{D}$
- feasible on hand
- 'contaminated' as $\mathcal{D}$ also used by $\mathcal{A}_m$ to 'select' g_m

test error E_{test}

- calculated from $\mathcal{D}_{\text{test}}$
- infeasible in boss's safe
- 'clean' as $\mathcal{D}_{\text{test}}$ never used for selection before

something in between: E_{val}

- calculated from $\mathcal{D}_{\text{val}} \subset \mathcal{D}$
- **feasible** on hand
- 'clean' **if** $\mathcal{D}_{\text{val}}$ never used by $\mathcal{A}_m$ before

selecting by E_{val}: **legal cheating :-)**

Fun Time

For $\mathcal{X} = \mathbb{R}^d$, consider two hypothesis sets, $\mathcal{H}_+$ and $\mathcal{H}_-$. The first hypothesis set contains all perceptrons with $w_1 \geq 0$, and the second hypothesis set contains all perceptrons with $w_1 \leq 0$. Denote g_+ and g_- as the minimum-E_{in} hypothesis in each hypothesis set, respectively. Which statement below is true?

1. If $E_{\text{in}}(g_+) < E_{\text{in}}(g_-)$, then g_+ is the minimum-E_{in} hypothesis of all perceptrons in $\mathbb{R}^d$.
2. If $E_{\text{test}}(g_+) < E_{\text{test}}(g_-)$, then g_+ is the minimum-E_{test} hypothesis of all perceptrons in $\mathbb{R}^d$.
3. The two hypothesis sets are disjoint.
4. None of the above

Fun Time

For $\mathcal{X} = \mathbb{R}^d$, consider two hypothesis sets, $\mathcal{H}_+$ and $\mathcal{H}_-$. The first hypothesis set contains all perceptrons with $w_1 \geq 0$, and the second hypothesis set contains all perceptrons with $w_1 \leq 0$. Denote g_+ and g_- as the minimum-E_{in} hypothesis in each hypothesis set, respectively. Which statement below is true?

1. If $E_{\text{in}}(g_+) < E_{\text{in}}(g_-)$, then g_+ is the minimum-E_{in} hypothesis of all perceptrons in $\mathbb{R}^d$.
2. If $E_{\text{test}}(g_+) < E_{\text{test}}(g_-)$, then g_+ is the minimum-E_{test} hypothesis of all perceptrons in $\mathbb{R}^d$.
3. The two hypothesis sets are disjoint.
4. None of the above

Reference Answer: 1

Note that the two hypothesis sets are not disjoint (sharing '$w_1 = 0$' perceptrons) but their union is all perceptrons.

Validation Set $\mathcal{D}_{\text{val}}$

$$
\begin{array}{ccccc}
E_{\text{in}}(h) & & & & E_{\text{val}}(h) \\
\uparrow & & & & \uparrow \\
\underbrace{\mathcal{D}}_{\text{size } N} & \rightarrow & \underbrace{\mathcal{D}_{\text{train}}}_{\text{size } N-K} & \cup & \underbrace{\mathcal{D}_{\text{val}}}_{\text{size } K} \\
\downarrow & & \downarrow & & \\
g_m = \mathcal{A}_m(\mathcal{D}) & & g_m^- = \mathcal{A}_m(\mathcal{D}_{\text{train}}) & &
\end{array}
$$

- $\mathcal{D}_{\text{val}} \subset \mathcal{D}$: called **validation set**—'on-hand' simulation of test set
- to connect E_{val} with E_{out}:
  $\mathcal{D}_{\text{val}} \overset{iid}{\sim} P(\mathbf{x}, y) \Longleftarrow$ select K examples from $\mathcal{D}$ at random
- to make sure $\mathcal{D}_{\text{val}}$ 'clean':
  feed only $\mathcal{D}_{\text{train}}$ to $\mathcal{A}_m$ for model selection

$$E_{\text{out}}(g_m^-) \le E_{\text{val}}(g_m^-) + O\left(\sqrt{\frac{\log M}{K}}\right)$$

Model Selection by Best E_{val}

$$m^* = \operatorname*{argmin}_{1 \le m \le M} \left(E_m = \textcolor{blue}{E_{\text{val}}}(\mathcal{A}_m(\textcolor{red}{\mathcal{D}_{\text{train}}})) \right)$$

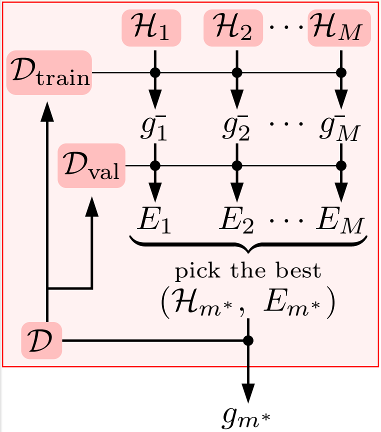
- generalization guarantee for all m:

$$E_{\text{out}}(\textcolor{red}{g_m^-}) \le \textcolor{blue}{E_{\text{val}}}(\textcolor{red}{g_m^-}) + O\left(\sqrt{\frac{\log M}{K}}\right)$$

- heuristic gain from $N - K$ to N:

$$E_{\text{out}}\left(\underbrace{g_{m^*}}_{\mathcal{A}_{m^*}(\mathcal{D})}\right) \le E_{\text{out}}\left(\underbrace{g_{m^*}^-}_{\mathcal{A}_{m^*}(\mathcal{D}_{\text{train}})}\right)$$

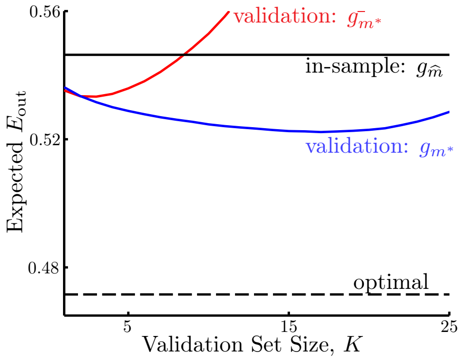
—**learning curve, remember? :-)**



$$E_{\text{out}}(g_{m^*}) \le E_{\text{out}}(\textcolor{red}{g_{m^*}^-}) \le \textcolor{blue}{E_{\text{val}}}(\textcolor{red}{g_{m^*}^-}) + O\left(\sqrt{\frac{\log M}{K}}\right)$$

Validation in Practice

use validation to select between $\mathcal{H}_{\mathbf{\Phi}_5}$ and $\mathcal{H}_{\mathbf{\Phi}_{10}}$



- in-sample: selection with E_{in}
- optimal: cheating-selection with E_{test}
- sub-g: selection with E_{val} and report $g^-_{m^*}$
- full-g: selection with E_{val} and report g_{m^*} —$E_{\text{out}}(g_{m^*}) \leq E_{\text{out}}(g^-_{m^*})$ indeed

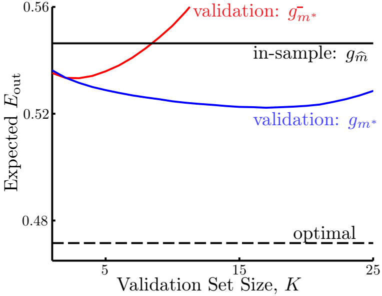
why is sub-g worse than in-sample some time?

The Dilemma about K

reasoning of validation:

$$E_{\text{out}}(g) \underset{\textbf{(small } K\textbf{)}}{\approx} E_{\text{out}}(g^-) \underset{\textbf{(large } K\textbf{)}}{\approx} E_{\text{val}}(g^-)$$

- large K: **every** $E_{\text{val}} \approx E_{\text{out}}$, but all g_m^- much worse than g_m
- small K: every $g_m^- \approx g_m$, but E_{val} far from E_{out}



practical rule of thumb: $K = \frac{N}{5}$

Fun Time

For a learning model that takes N^2 seconds of training when using N examples, what is the total amount of seconds needed when running the whole validation procedure with $K = \frac{N}{5}$ on 25 such models with different parameters to get the final g_{m^*}?

1. $6N^2$
2. $17N^2$
3. $25N^2$
4. $26N^2$

Fun Time

For a learning model that takes N^2 seconds of training when using N examples, what is the total amount of seconds needed when running the whole validation procedure with $K = \frac{N}{5}$ on 25 such models with different parameters to get the final g_{m^*}?

1. $6N^2$
2. $17N^2$
3. $25N^2$
4. $26N^2$

Reference Answer: 2

To get all the g_m^-, we need $\frac{16}{25}N^2 \cdot 25$ seconds.
Then to get g_{m^*}, we need another N^2 seconds.
So in total we need $17N^2$ seconds.

Extreme Case: $K = 1$

reasoning of validation:

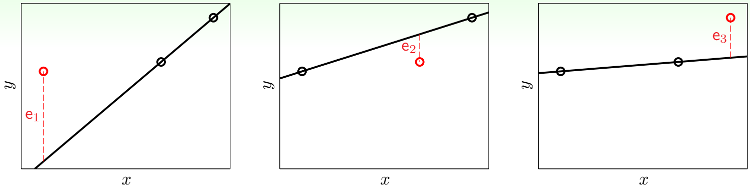
$$E_{\text{out}}(g) \underset{\textbf{(small } K\textbf{)}}{\approx} E_{\text{out}}(g^-) \underset{\textbf{(large } K\textbf{)}}{\approx} E_{\text{val}}(g^-)$$

- take $K = 1$? $\mathcal{D}_{\text{val}}^{(n)} = \{(\mathbf{x}_n, y_n)\}$ and $E_{\text{val}}^{(n)}(g_n^-) = \text{err}(g_n^-(\mathbf{x}_n), y_n) = e_n$
- make e_n closer to $E_{\text{out}}(g)$?—average over possible $E_{\text{val}}^{(n)}$
- leave-one-out cross validation estimate:

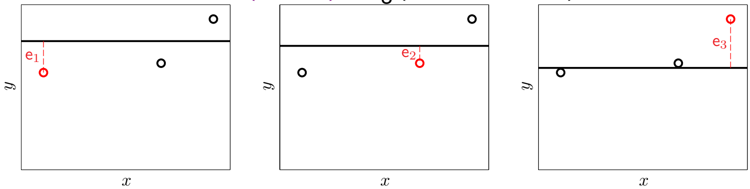
$$E_{\text{loocv}}(\mathcal{H}, \mathcal{A}) = \frac{1}{N}\sum_{n=1}^{N} e_n = \frac{1}{N}\sum_{n=1}^{N} \text{err}(g_n^-(\mathbf{x}_n), y_n)$$

hope: $E_{\text{loocv}}(\mathcal{H}, \mathcal{A}) \approx E_{\text{out}}(g)$

Illustration of Leave-One-Out



$$E_{\text{loocv}}(\text{linear}) = \frac{1}{3}(e_1 + e_2 + e_3)$$



$$E_{\text{loocv}}(\text{constant}) = \frac{1}{3}(e_1 + e_2 + e_3)$$

which one would you choose?

$$m^* = \underset{1 \le m \le M}{\text{argmin}}(E_m = E_{\text{loocv}}(\mathcal{H}_m, \mathcal{A}_m))$$

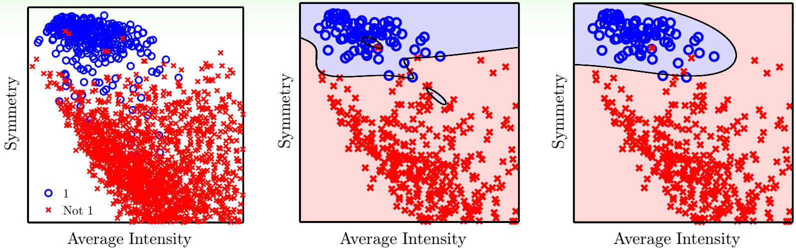
Theoretical Guarantee of Leave-One-Out Estimate

does $E_{\text{loocv}}(\mathcal{H}, \mathcal{A})$ say something about $E_{\text{out}}(g)$?
yes, for average E_{out} on size-$(N-1)$ data

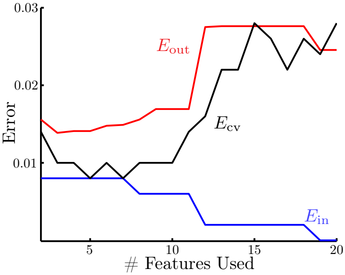
$$
\begin{aligned}
\mathop{\mathcal{E}}_{\mathcal{D}} E_{\text{loocv}}(\mathcal{H}, \mathcal{A}) = \mathop{\mathcal{E}}_{\mathcal{D}} \frac{1}{N}\sum_{n=1}^{N} e_n \quad &= \quad \frac{1}{N}\sum_{n=1}^{N} \mathop{\mathcal{E}}_{\mathcal{D}} e_n \\
&= \quad \frac{1}{N}\sum_{n=1}^{N} \mathop{\mathcal{E}}_{\mathcal{D}_n} \mathop{\mathcal{E}}_{(\mathbf{x}_n, y_n)} \mathrm{err}(g_n^-(\mathbf{x}_n), y_n) \\
&= \quad \frac{1}{N}\sum_{n=1}^{N} \mathop{\mathcal{E}}_{\mathcal{D}_n} E_{\text{out}}(g_n^-) \\
&= \quad \frac{1}{N}\sum_{n=1}^{N} \overline{E_{\text{out}}}(N-1) = \overline{E_{\text{out}}}(N-1)
\end{aligned}
$$

expected $E_{\text{loocv}}(\mathcal{H}, \mathcal{A})$ says something about expected $E_{\text{out}}(g^-)$
—often called 'almost unbiased estimate of $E_{\text{out}}(g)$'

Leave-One-Out in Practice



select by E_{in} select by E_{loocv}



E_{loocv} much better than E_{in}

Fun Time

Consider three examples $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3)$ with $y_1 = 1$, $y_2 = 5$, $y_3 = 7$. If we use E_{loocv} to estimate the performance of a learning algorithm that predicts with the average y value of the data set—the optimal constant prediction with respect to the squared error. What is E_{loocv} (squared error) of the algorithm?

1. 0
2. $\frac{56}{9}$
3. $\frac{60}{9}$
4. 14

Fun Time

Consider three examples $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3)$ with $y_1 = 1$, $y_2 = 5$, $y_3 = 7$. If we use E_{loocv} to estimate the performance of a learning algorithm that predicts with the average y value of the data set—the optimal constant prediction with respect to the squared error. What is E_{loocv} (squared error) of the algorithm?

1. 0
2. $\frac{56}{9}$
3. $\frac{60}{9}$
4. 14

Reference Answer: 4

This is based on a simple calculation of
$e_1 = (1-6)^2, e_2 = (5-4)^2, e_3 = (7-3)^2$.

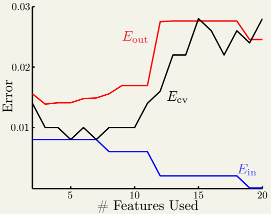
Disadvantages of Leave-One-Out Estimate

Computation

$$E_{\text{loocv}}(\mathcal{H}, \mathcal{A}) = \frac{1}{N}\sum_{n=1}^{N} e_n = \frac{1}{N}\sum_{n=1}^{N} \text{err}(g_n^-(\mathbf{x}_n), y_n)$$

- N 'additional' training per model, not always feasible in practice
- except 'special case' like analytic solution for linear regression

Stability—due to variance of single-point estimates

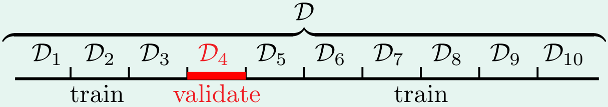


E_{loocv}: not often used practically

V-fold Cross Validation

how to **decrease computation need** for cross validation?

- essence of leave-one-out cross validation: partition $\mathcal{D}$ to N parts, taking $N-1$ for training and 1 for validation orderly
- V-fold cross-validation: random-partition of $\mathcal{D}$ **to V equal parts**,



take $V-1$ for training and 1 for validation orderly

$$E_{\text{cv}}(\mathcal{H}, \mathcal{A}) = \frac{1}{V} \sum_{v=1}^{V} E_{\text{val}}^{(v)}(g_v^-)$$

- selection by E_{cv}: $m^* = \underset{1 \le m \le M}{\text{argmin}}(E_m = E_{\text{cv}}(\mathcal{H}_m, \mathcal{A}_m))$

practical rule of thumb: $V = 10$

Final Words on Validation

'Selecting' Validation Tool

- ***V*-Fold** generally preferred over single validation if computation allows
- **5-Fold or 10-Fold** generally works well: not necessary to trade V-Fold with Leave-One-Out

Nature of Validation

- all training models: select among hypotheses
- all validation schemes: select among finalists
- all testing methods: just evaluate

validation still more optimistic than testing

do not fool yourself and others **:-)**,
report test result, not best validation result

Fun Time

For a learning model that takes N^2 seconds of training when using N examples, what is the total amount of seconds needed when running 10-fold cross validation on 25 such models with different parameters to get the final g_{m^*}?

1. $\frac{47}{2}N^2$
2. $47N^2$
3. $\frac{407}{2}N^2$
4. $407N^2$

Fun Time

For a learning model that takes N^2 seconds of training when using N examples, what is the total amount of seconds needed when running 10-fold cross validation on 25 such models with different parameters to get the final g_{m^*}?

1. $\frac{47}{2}N^2$
2. $47N^2$
3. $\frac{407}{2}N^2$
4. $407N^2$

Reference Answer: 3

To get all the E_{cv}, we need $\frac{81}{100}N^2 \cdot 10 \cdot 25$ seconds. Then to get g_{m^*}, we need another N^2 seconds. So in total we need $\frac{407}{2}N^2$ seconds.

Summary

1. When Can Machines Learn?
2. Why Can Machines Learn?
3. How Can Machines Learn?
4. How Can Machines Learn **Better**?

Lecture 14: Regularization

Lecture 15: Validation

- Model Selection Problem
  dangerous by E_{in} and dishonest by E_{test}
- Validation
  select with $E_{\text{val}}(\mathcal{D}_{\text{train}})$ while returning $\mathcal{A}_{m^*}(\mathcal{D})$
- Leave-One-Out Cross Validation
  huge computation for almost unbiased estimate
- V-Fold Cross Validation
  reasonable computation and performance

- **next: something 'up my sleeve'**