UNIVERSITÀ DEGLI STUDI DI BERGAMO
Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Lesson 6.

Validation and cross-validation

DATA SCIENCE AND AUTOMATION COURSE

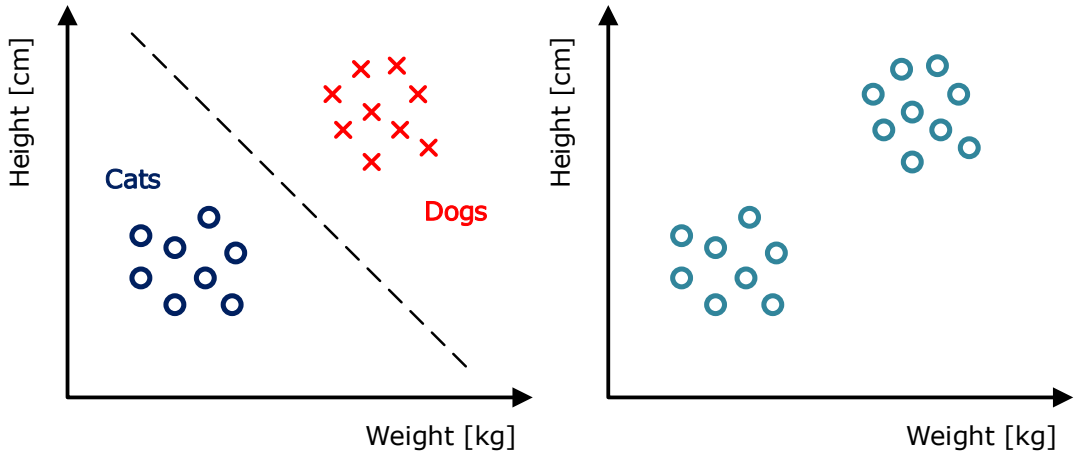
MASTER DEGREE SMART TECHNOLOGY ENGINEERING

TEACHER
Mirko Mazzoleni

PLACE
University of Bergamo

Review of the previous lessons

- **Supervised vs. Unsupervised**



- **VC Dimension**

$$N \geq 10 \cdot d_{VC}$$

- **VC Generalization bound**

$$E_{out} \leq E_{in} + \Omega$$

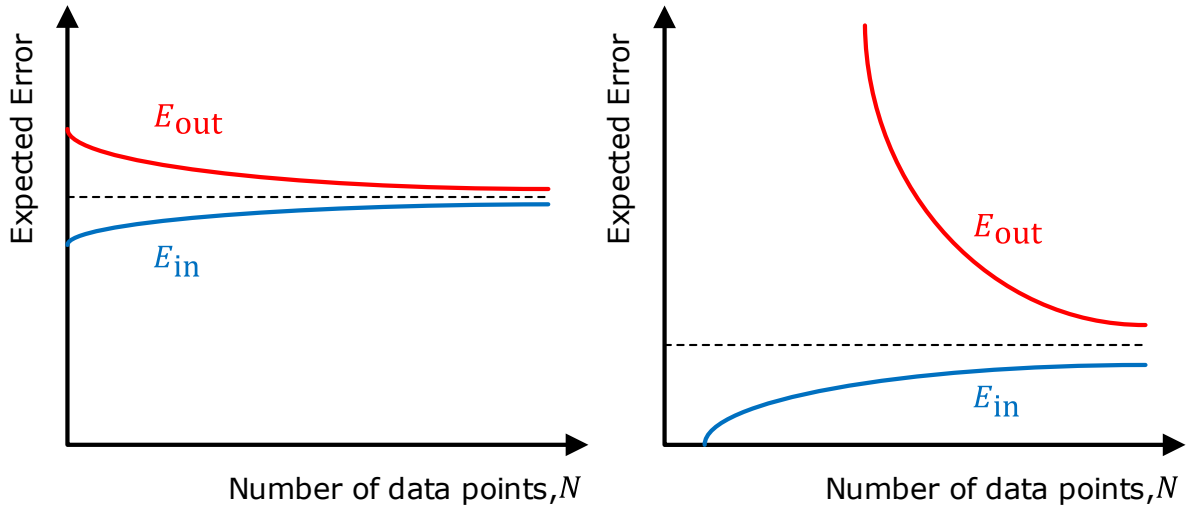
- **Bias and variance**

Expected value of E_{out} w.r.t. $\mathcal{D}$

$$= \textbf{bias} + \textbf{variance}$$

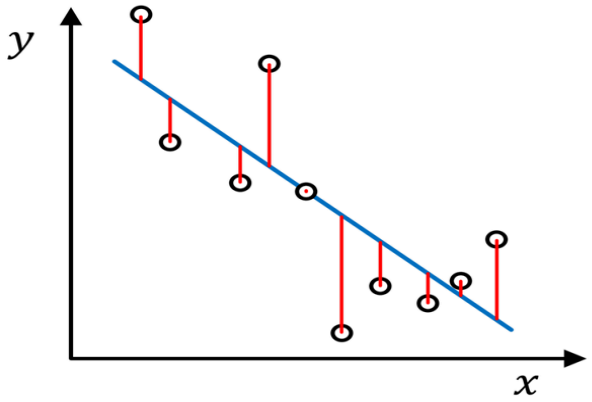
$$g^{\mathcal{D}}(\boldsymbol{x}) \rightarrow \bar{g}(\boldsymbol{x}) \rightarrow f(\boldsymbol{x})$$

- **Learning curves**



Review of the previous lessons

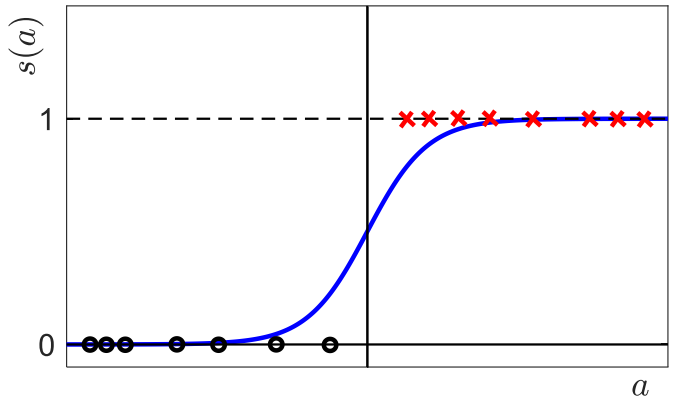
- **Linear regression model**



$$h(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\theta}$$

- **Logistic regression model**

Predict the probability of the class $y = 1$ given $\boldsymbol{x}$, $P(y = 1|\boldsymbol{x})$



$$h(\boldsymbol{x}) = s(\boldsymbol{x}^T\boldsymbol{\theta})$$

$$s(a) = \frac{1}{1+e^{-a}}$$

$$s(\boldsymbol{x}^T\boldsymbol{\theta}) \equiv \pi$$

- **Linear regression algorithm**

Minimize in-sample squared error

$$E_{in}(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) = \frac{1}{N}(X\boldsymbol{\theta} - Y)^T(X\boldsymbol{\theta} - Y)$$

$$\widehat{\boldsymbol{\theta}} = (X^TX)^{-1}X^TY$$

- **Logistic regression algorithm**

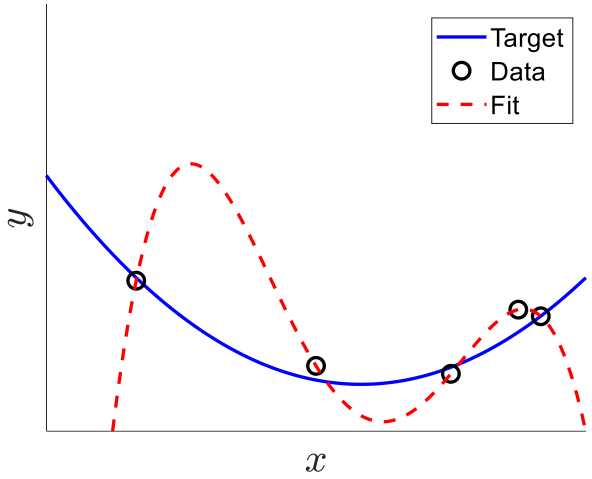
Minimize in-sample cross-entropy error - MLE

$$E_{in}(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) = -\sum_{i=1}^{N}(\, y(i)\ln\pi(i) + (1 - y(i)\ln[1 - \pi(i)]\,)$$

- **Gradient descent** $\hat{\theta}(t+1) = \hat{\theta}(t) - \alpha\nabla J(\theta)\Big|_{\theta=\widehat{\theta}(t)}$

Review of the previous lessons

- **Overfitting**



$E_{in} = 0 \qquad E_{out} = \textbf{huge}$

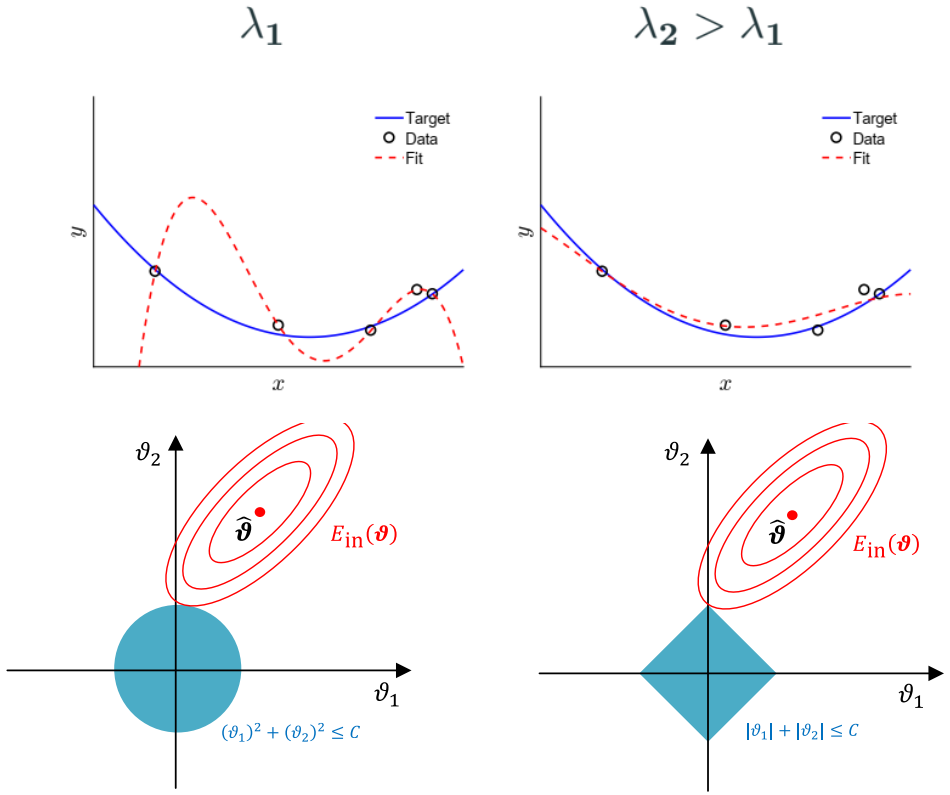
- **Bias and variance plus noise**

Expected value of E_{out} w.r.t. $\mathcal{D}$

$= \textbf{bias} + \textbf{variance} + \textbf{noise}$

- **Regularization**

$$E_{aug}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\left(h(\boldsymbol{x}(i),\boldsymbol{\theta}) - f(\boldsymbol{x}(i))\right)^2 + \lambda \cdot \sum_{j=1}^{d-1}(\theta_j)^2$$



Outline

1. Validation
2. Model selection
3. Cross-validation
4. Summary

Outline

1. **Validation**
2. Model selection
3. Cross-validation
4. Summary

Validation

The out of sample error can be seen as: $E_{out}(h) = E_{in}(h) + \text{overfit penalty}$

Regularization

$$E_{out}(h) = E_{in}(h) + \underbrace{\text{overfit penalty}}$$

Regularization estimates this quantity

Validation

$$\underbrace{E_{out}(h)} = E_{in}(h) + \text{overfit penalty}$$

Validation estimates this quantity

Validation set

The idea of a **validation** set is to estimate the model performance out of sample

1. Remove a **subset** from the training data → this subset is not used for training
2. Train the model on the remaining training data → the model will be trained on *less* data
3. Evaluate the model performance on the **held-out** set → this is an unbiased estimation of the out-of-sample error
4. Retrain the model on **all** the data

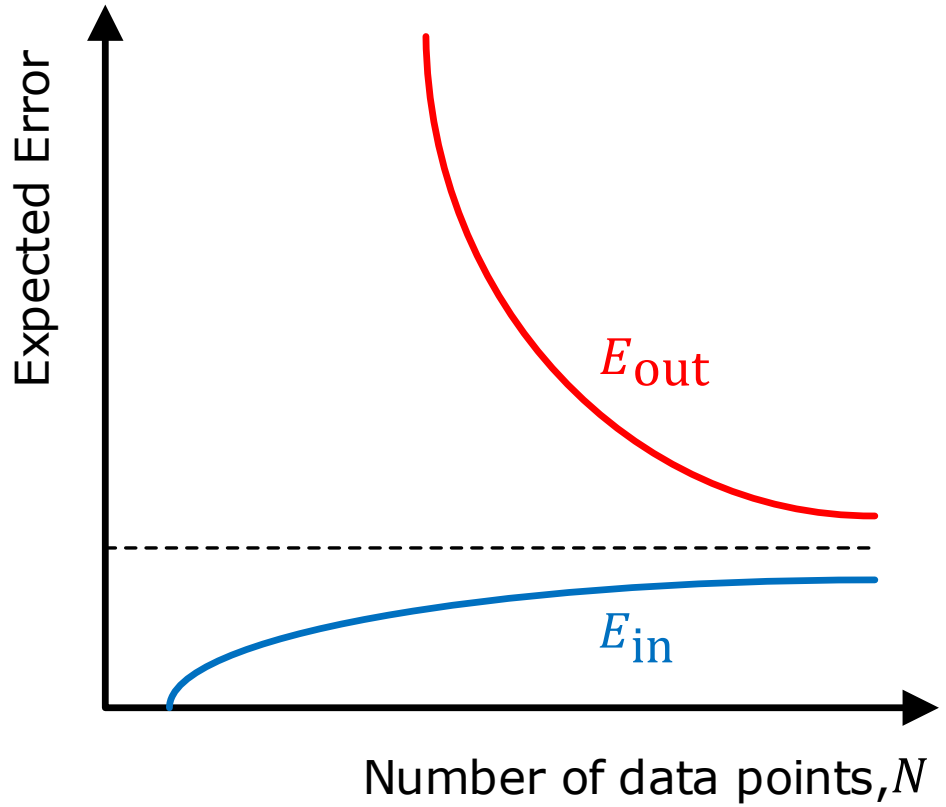
K is taken out of N

Given the dataset $\mathcal{D} = \{(\boldsymbol{x}(1), y(1)), \dots, (\boldsymbol{x}(N), y(N))\}$

K points: **validation** $N - K$ points: **training**

$\mathcal{D}_{val}$ $\mathcal{D}_{train}$

- **Small** K: bad estimate of E_{out}
- **Large** K: possibility of learning a bad model (learning curve)



K is put back into N

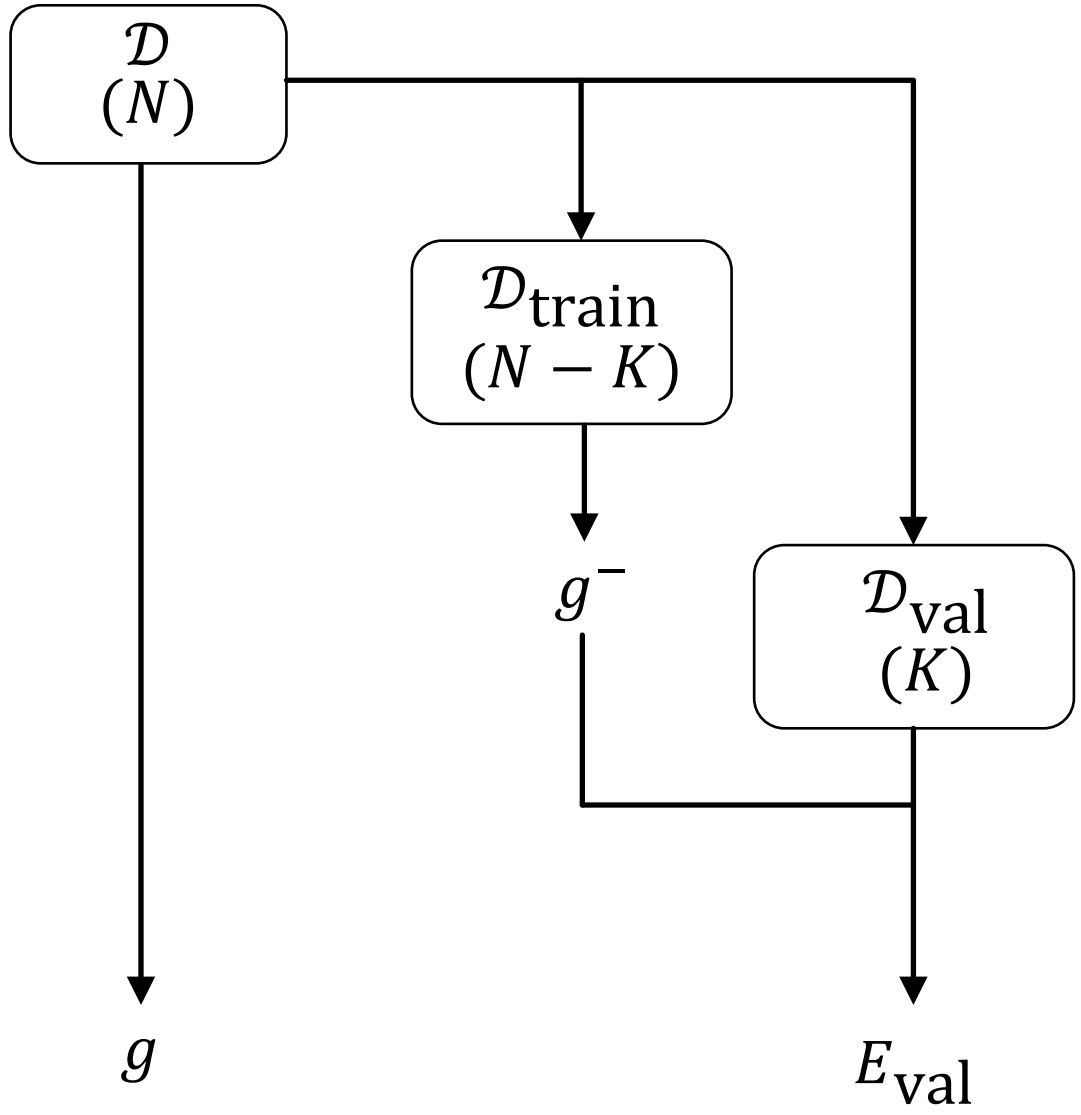
$$\begin{array}{ccccc} \mathcal{D} & \rightarrow & \mathcal{D}_{\text{train}} & \cup & \mathcal{D}_{\text{val}} \\ \downarrow & & \downarrow & & \downarrow \\ N & & N-K & & K \end{array}$$

$$\mathcal{D} \Longrightarrow g \qquad \mathcal{D}_{\text{train}} \Longrightarrow g^{-}$$

$$E_{\text{val}} = E_{\text{val}}(g^{-})$$

Rule of thumb

$$K = \frac{N}{5}$$



Outline

1. Validation

2. **Model selection**

3. Cross-validation

4. Summary

Model selection

The most important use of a validation set is **model selection:**

- Choose between a linear model and a nonlinear one
- Choice of the order of the polynomial
- Choice of the regularization parameter
- Any other choice that affects the learning of the model

If the validation set is used to perform choices (e.g. to select the regularization parameter λ), then it **no longer** provides an unbiased estimate of E_{out}

There is the need of a third dataset: the **test set,** with which to measure the model performance E_{test}

Using $\mathcal{D}_{val}$ more than once

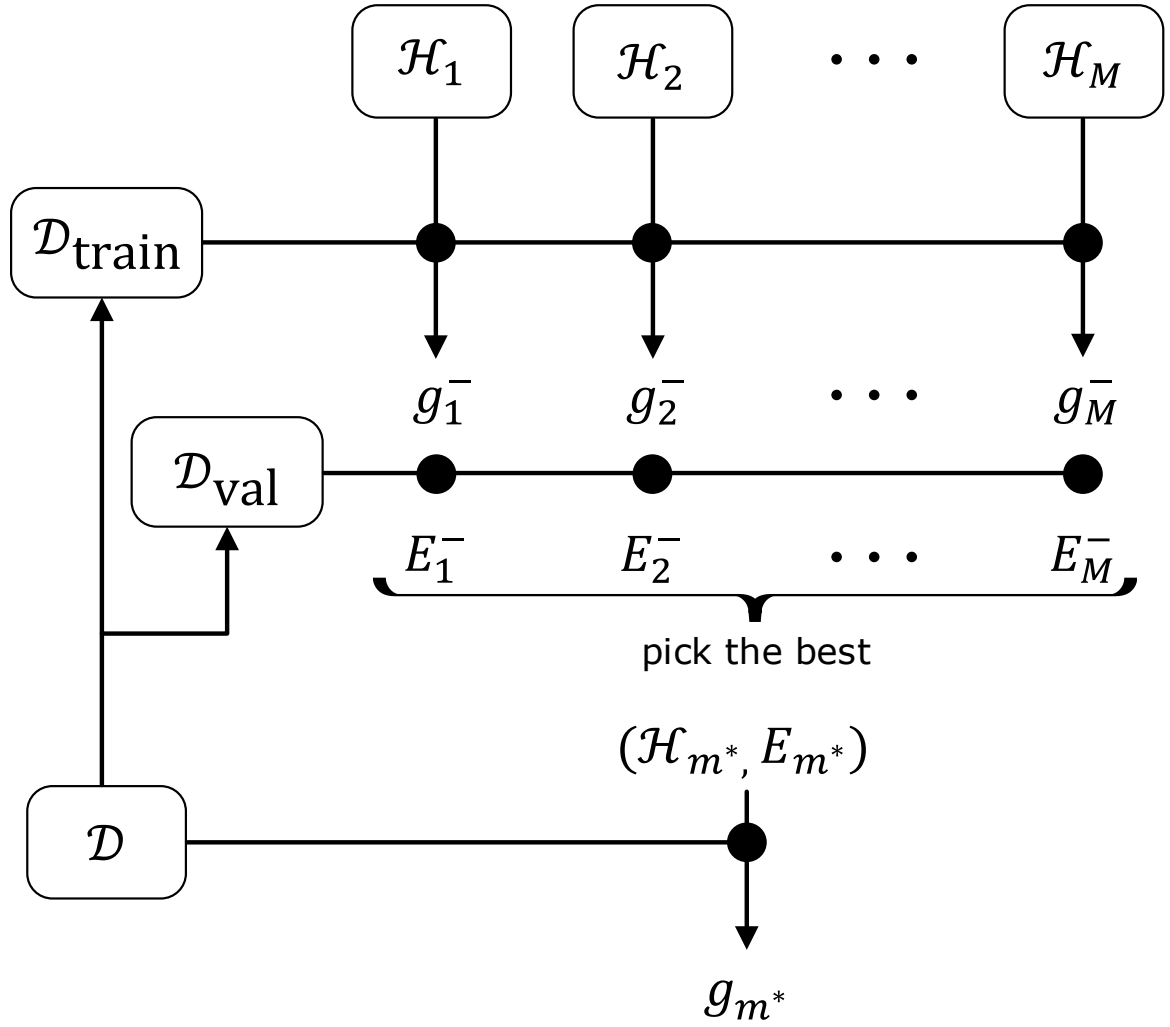
M models $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_M$

Use $\mathcal{D}_{train}$ to learning g_m^- for each model

Evaluate g_m^- using $\mathcal{D}_{val}$

$$E_m = E_{val}(g_m^-) \qquad m = 1, \ldots, M$$

Pick the model $m = m^*$ with smallest E_m



How much bias

For the M models $\mathcal{H}_1, \mathcal{H}_2, \dots, \mathcal{H}_M$, $\mathcal{D}_{val}$ is used for "training" on the **finalist model set:**

$$\mathcal{H}_{val} = \{g_1^-, g_2^-, \dots, g_M^-\}$$

- The validation performance of the final model is $E_{val}(g_{m^*}^-)$
- This quantity **is biased** and not representative of $E_{out}(g_{m^*}^-)$, just as the in sample error E_{in} was not representative of E_{out}
- What happened is that $\mathcal{D}_{val}$ has become the "training set" for $\mathcal{H}_{val}$
- The risk is to **overfit the validation set**

Data contamination

Error estimates: $E_{in}, E_{val}, E_{test}$

Contamination: Optimistic bias in estimating E_{out}

- **Training set:** totally contaminated
- **Validation set:** slightly contaminated
- **Test set:** totally “clean”

Outline

1. Validation

2. Model selection

3. **Cross-validation**

4. Summary

The dilemma about K

The following chain of reasoning:

$$E_{out}(g) \approx E_{out}(g^-) \approx E_{val}(g^-)$$

(small K) (large K)

Highlights the dilemma in selecting K

Can we have K **both** small and large?

Leave one out cross-validation

Use $N-1$ points for training and $K=1$ point for validation

$$\mathcal{D}_i = \{\boldsymbol{x}(1), y(1)\}, \dots, \{\boldsymbol{x}(i), y(i)\}, \dots, \{\boldsymbol{x}(N), y(N)\}$$

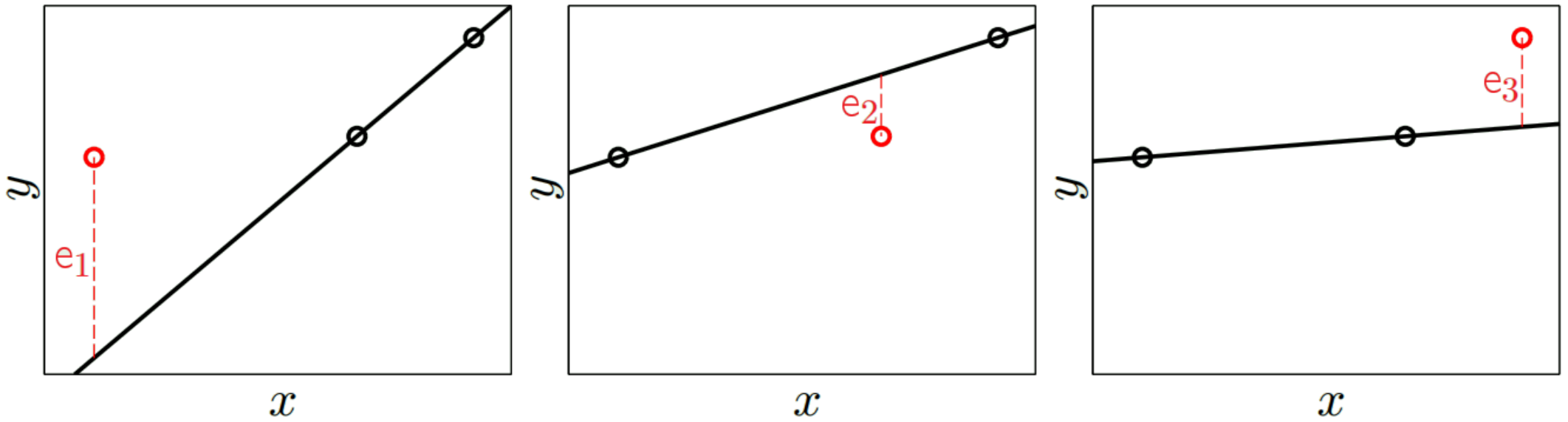
where D_i is the training set without the point i. The final hypothesis learned from $\mathcal{D}_i$ is g_i^-

The validation error on the point $\boldsymbol{x}(i)$ is $e(i) = E_{val}(g_i^-) = e\left(g_i^-\left(\boldsymbol{x}(i)\right), y(i)\right)$

It is then possible to define the **cross-validation error**

$$E_{cv} = \frac{1}{N}\sum_{i=1}^{N} e(i)$$

Cross-validation in action



$$E_{\text{cv}} = \frac{1}{3}\left(\mathrm{e}_1 + \mathrm{e}_2 + \mathrm{e}_3 \right)$$

Pictures taken from [5]

Cross-validation for model selection

Cross-validation can be used effectively to **perform model selection** by selecting the appropriate regularization parameter λ

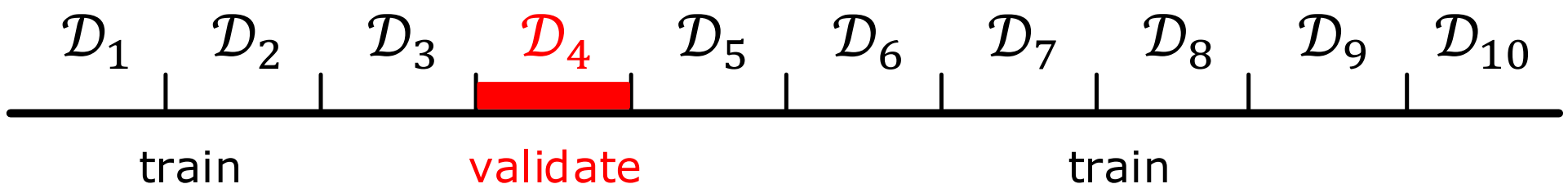
1. Define M models by choosing different values for λ: $(\mathcal{H}, \lambda_1), (\mathcal{H}, \lambda_2), \dots, (\mathcal{H}, \lambda_M)$
2. **for** each model $m = 1, \dots, M$ **do**
    2.1 Use cross-validation to obtain estimates of the out of sample error for each model
3. Select the model m^* with the smallest cross-validation error $E_{cv}(m^*)$
4. Use the model $(\mathcal{H}, \lambda_{m^*})$ and all the data $\mathcal{D}$ to obtain the final hypothesis g_{m^*}

Leave more than one out

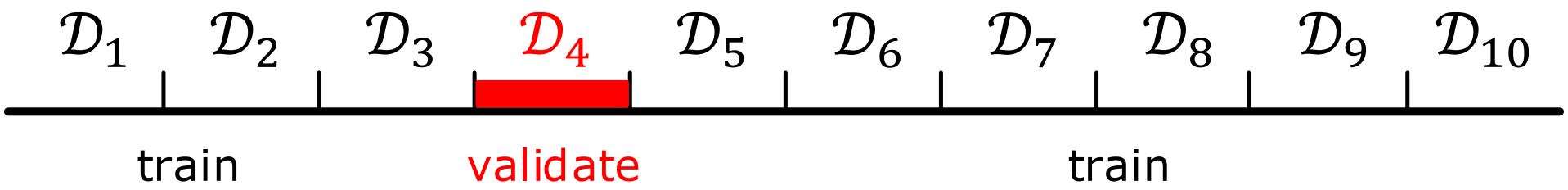
Leave-one-out cross-validation has the disadvantage that:

- It is **computationally expensive**, requiring a total of N training sessions for each of the M models
- The estimated cross-validation error estimate **has high variance**, since it is based only on one point

It is possible to reserve more points for validation by dividing the training set in **"folds"**



Leave more than one out



- This produces $\frac{N}{K}$ training session on $N - K$ points each
- A good **compromise** for the number of folds is 10

$$\mathbf{10-fold\ cross\ validation:}\ K = \frac{N}{10}$$

- Pay attention to not reduce the training set to much (look at the learning curves)

The wrong and right way to do cross-validation

Consider a classification problem with a large number of predictors [9]. A strategy could be as follows:

1. Find a subset of predictors that show strong (univariate) correlation with the class labels

2. Using just this subset of predictors, build a multivariate classifier

3. Use cross-validation to estimate the unknown tuning parameters and to estimate the out of sample error of the final model

Is this a correct application of cross validation?

NO!

The wrong and right way to do cross-validation

- The predictors have an unfair advantage, as they were chosen on the basis of **all samples** (step 1)
- Leaving samples out **after** the variables have been selected does not correctly mimic the application of the classifier to a completely independent test set
- The predictors (and therefore the model) **have already seen** the left out samples
- The data used for validation were **already used to make a choice** that involved the class labels (this is **not correct**)

Outline

1. Validation

2. Model selection

3. Cross-validation

4. **Summary**

Validation summary

- If we have a lot of data, the **best way** is to divide the dataset in training, validation and test sets
- Otherwise, we can use cross-validation
- If the data are **really scarce**, we can use formulas for choosing the optimal model complexity that use only the training set
  - ✓ Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC)
  - ✓ Structural Risk Minimization (SRM) minimizing the VC upper bound

References

1. Provost, Foster, and Tom Fawcett. "*Data Science for Business: What you need to know about data mining and data-analytic thinking*". O'Reilly Media, Inc., 2013.
2. Brynjolfsson, E., Hitt, L. M., and Kim, H. H. *"Strength in numbers: How does data driven decision making affect firm performance?"* Tech. rep., available at SSRN: http://ssrn.com/abstract=1819486, 2011.
3. Pyle, D. "*Data Preparation for Data Mining*". Morgan Kaufmann, 1999.
4. Kohavi, R., and Longbotham, R. *"Online experiments: Lessons learned".* Computer, 40 (9), 103–105, 2007.
5. Abu-Mostafa, Yaser S., Malik Magdon-Ismail, and Hsuan-Tien Lin. *"Learning from data"*. AMLBook, 2012.
6. Andrew Ng. *"Machine learning"*. Coursera MOOC. (https://www.coursera.org/learn/machine-learning)
7. Domingos, Pedro. "*The Master Algorithm".* Penguin Books, 2016.
8. Christopher M. Bishop, "*Pattern recognition and machine learning", Springer-Verlag New York, 2006.*
9. Hastie, T., Tibshirani, R.,, Friedman, J. *"The Elements of Statistical Learning"*. New York, NY, USA: Springer New York Inc, 2001.