02450: Introduction to Machine Learning and Data Mining

Overfitting and performance evaluation

DTU Compute
Department of Applied Mathematics and Computer Science

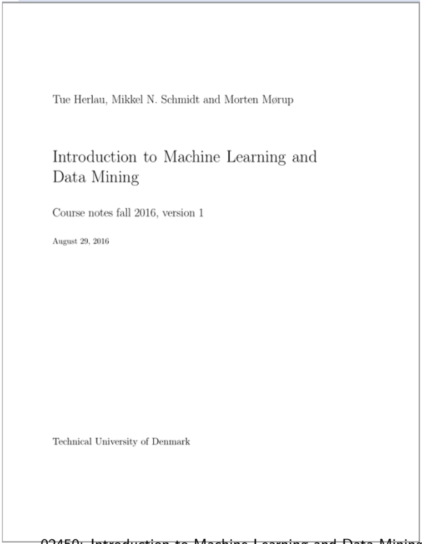
Reading Material

Reading material:

C9

Feedback Groups of the day:

- Christian Tarning-Andersen, Mirrin Snel
- Ulrika Boulund, Kristin J. Lillekjendlie
- Niklas Hansson, Mathias Sondrup, Mallory Maline
- Jannick Lønver, Emilie Lildholdt
- Oliver Brandt, Martin Johnsen, Jonas Waaben
- Ioulia Markou, Jacob Jon Hansen, Sebastiano Piccolo
- Ioannis Kavadakis, Athina Tsagkari
- Helga Svala Sigurðardóttir, Anna


Tue Herlau, Mikkel N. Schmidt and Morten Mørup

Introduction to Machine Learning and Data Mining

Course notes fall 2016, version 1

August 29, 2016

Technical University of Denmark

Lecture Schedule

1. Introduction
   30 August: C1

Data: Feature extraction, and visualization

2. Data and feature extraction
   6 September: C2, C3
3. Measures of similarity and summary statistics
   13 September: C4
4. Data Visualization and probability
   20 September: C5, C6

Supervised learning: Classification and regression

5. Decision trees and linear regression
   27 September: C7, C8 **(Project 1 due before 13:00)**
6. **Overfitting and performance evaluation**
   4 October: C9
7. Nearest Neighbor, Bayes and Naive Bayes
   11 October: C10, C11
8. Artificial Neural Networks and Bias/Variance
   25 October: C12, C13
9. AUC and ensemble methods
   1 November: C14, C15

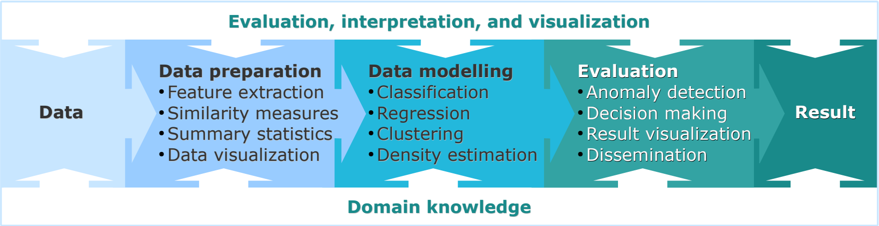
Unsupervised learning: Clustering and density estimation

10. K-means and hierarchical clustering
    8 November: C16 **(Project 2 due before 13:00)**
11. Mixture models and association mining
    15 November: C17, C18
12. Density estimation and anomaly detection
    22 November: C19

Recap

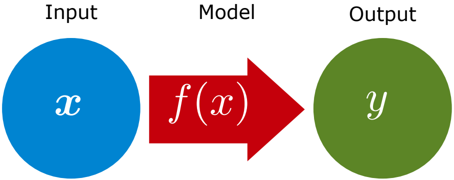
13. Recap and discussion of the exam
    29 November: C1-C19 **(Project 3 due before 13:00)**

Data modeling framework



After today you should be able to:
Explain the difference between training, test and generalization error
Explain how cross-validation can be used for (i) performance evaluation (ii) model selection
Apply forward and backward selection
Test the significance of classifiers

Supervised learning



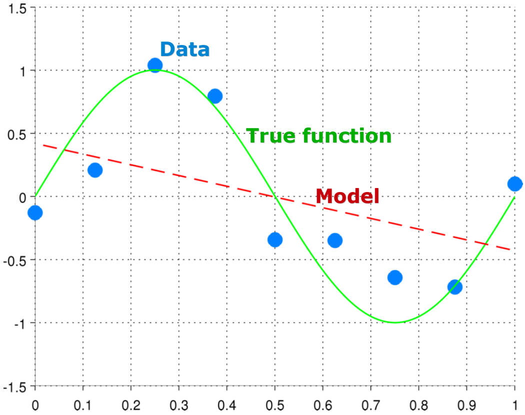
- **Mapping between domains**
  - Classification: Discrete output
  - Regression: Continuous output

Roadmap for today:

- Introduce errors:
  - **training error**
  - **test error**
  - **generalization error**
- Introduce cross-validation techniques
  - **basic cross-validation** for **performance evaluation**
  - **cross-validation** for **model selection**
  - **two-layer cross-validation** for **model selection AND performance evaluation**
- Statistical evaluation of the performance of classifiers
  - **Evaluation of a single classifier**
  - **Comparing two classifiers**

Why are there "multiple models"? Example: Linear regression

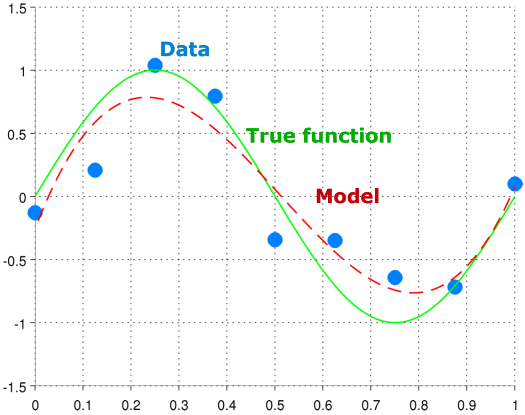
- Bad fit
- **Too simple model**



$$f(x) = w_0 + w_1 x$$

Why are there "multiple models"? Example: Linear regression

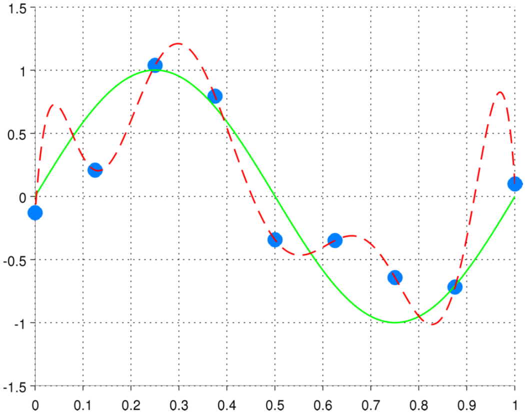
- Reasonable fit
- **Reasonable model**



$$f(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

Why are there "multiple models"? Example: Linear regression

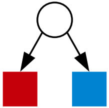
- Perfect fit
- **Too complex model**

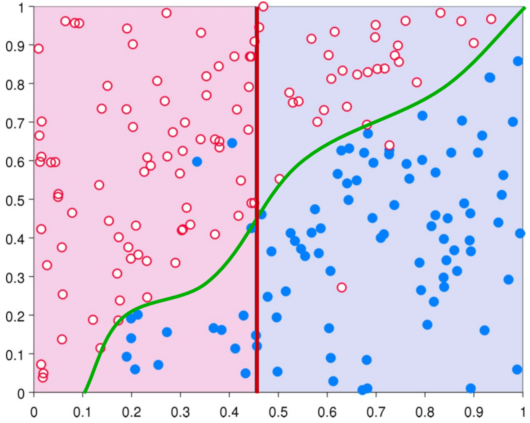


$$f(x) = w_0 + w_1 x + \cdots + w_8 x^8$$

Why are there “multiple models”?
Example: Classification tree

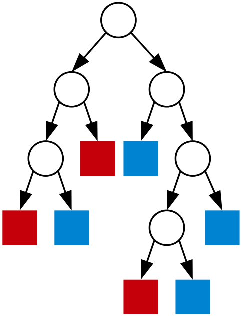
- Bad fit
- **Too simple model**

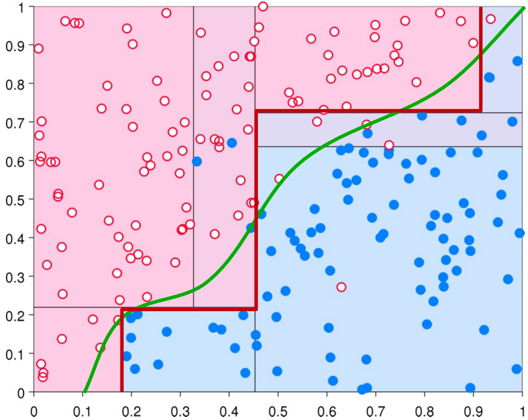




Why are there "multiple models"? Example: Classification tree

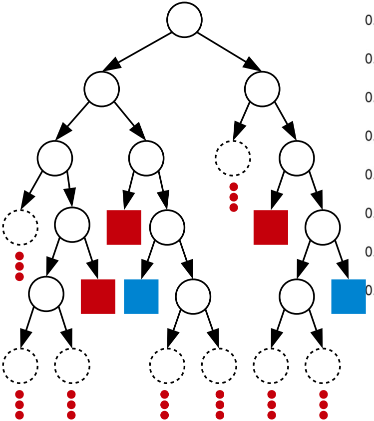
- Reasonable fit
- **Reasonable model**

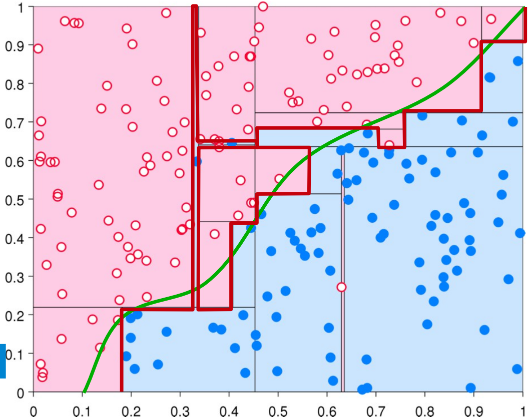




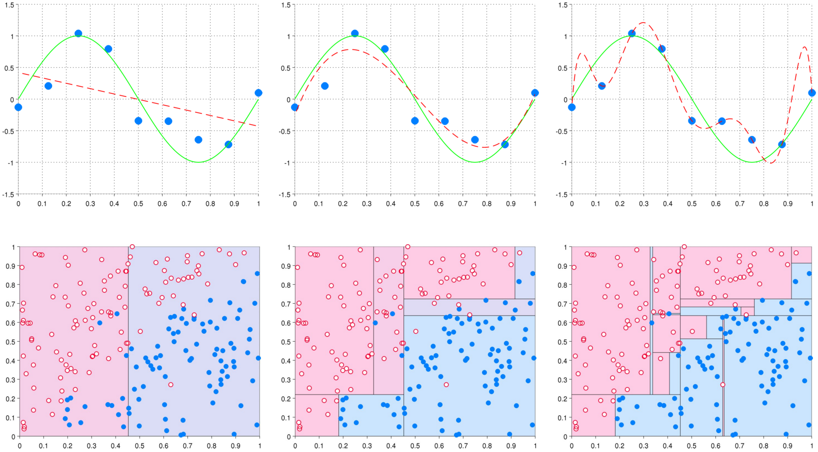
Why are there "multiple models"? Example: Classification tree

- Perfect fit
- **Too complex model**



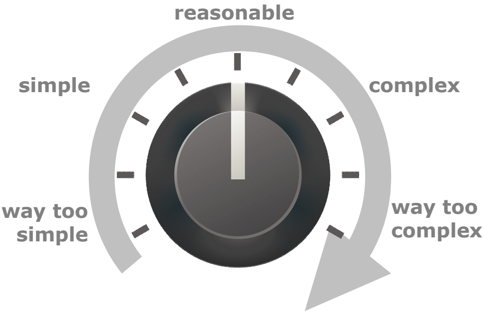


Model overfitting



Control the model complexity

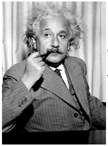
- Find **parameter** or **mechanism** in model that controls complexity



Lex Parsimoniae, Law of parsimony

Given two models with same predictive performance, the simpler model is preferred over the more complex model - William of Ockham (1288-1347) (paraphrased)

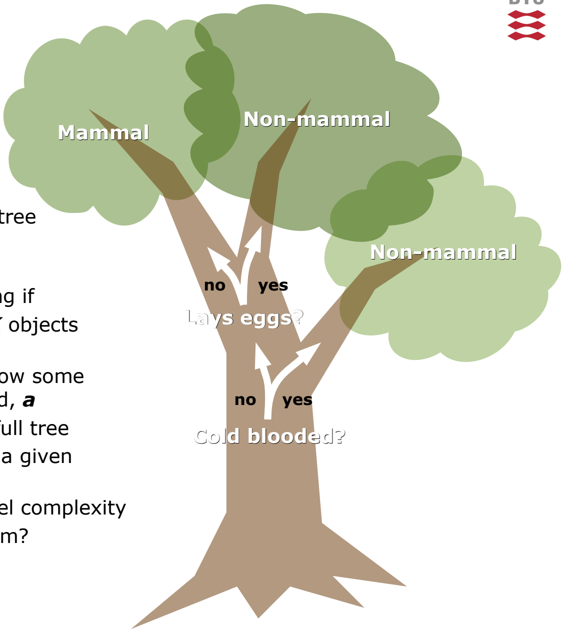
https://commons.wikimedia.org/wiki/File:William_of_Ockham.png



"Everything should be made as simple as possible, but not simpler" - Einstein

Decision trees

- Hunts algorithm
  - Continue splitting until each node is pure
  - Results in a very complex tree (overfitting)
- **Control complexity**
  - **Pre-pruning**: Stop splitting if
    - There is less than ***K*** objects on the branch
    - Impurity gain is below some predefined threshold, ***a***
  - **Post-pruning**: Generate full tree
    - Cut off branches to a given pruning level, ***c***
- ***K, a,*** and/or ***c*** determine model complexity
  - How should we choose them?

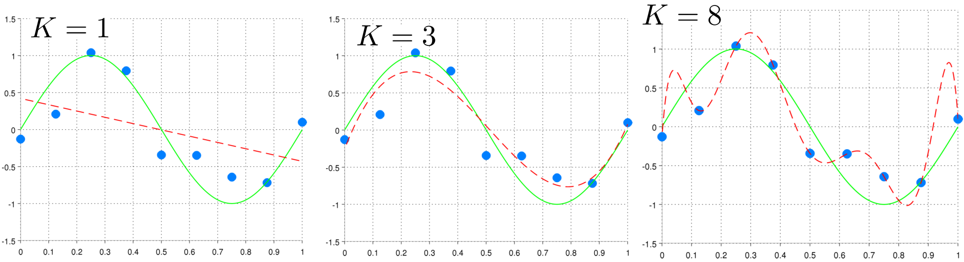


Linear regression

- Linear regression on non-linearly transformed inputs (polynomials)

$$f(x) = w_0 + w_1 x + \cdots + w_8 x^8$$

  - **Control complexity**: Choose a suitable value for *K*

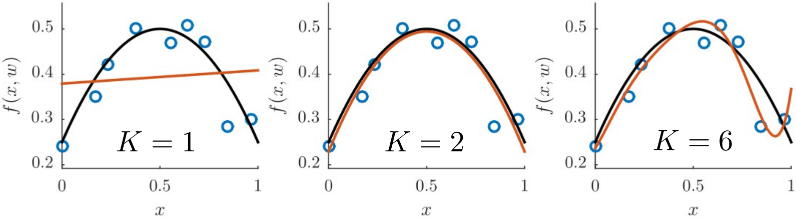


Solution:
Assess model performance correctly and select best model

Training error

- Suppose we train 3 models on a dataset of 9 observations

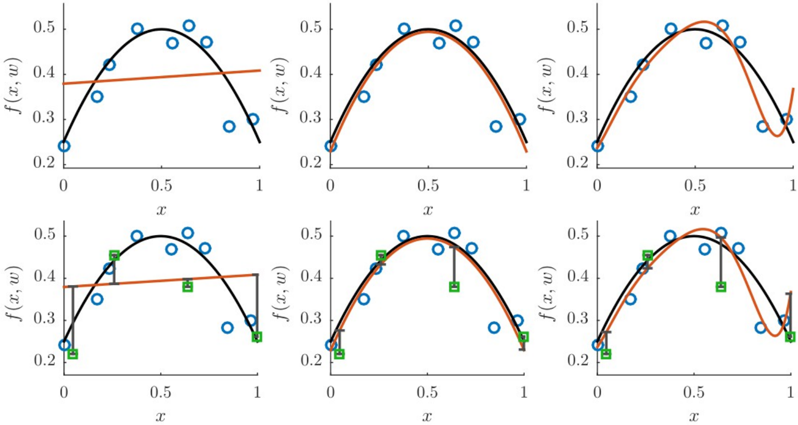
$$\mathcal{M}_1 = \{\text{1'st order polynomial}\}$$
$$\mathcal{M}_2 = \{\text{2'nd order polynomial}\}$$
$$\mathcal{M}_3 = \{\text{6'th order polynomial}\}$$



$$E^{\text{train}}_{\mathcal{M}_k} = \frac{1}{N^{\text{train}}} \sum_{i \in \mathcal{D}^{\text{train}}} (y_i - f_{\mathcal{M}_k}(x_i, \boldsymbol{w}))^2.$$

Test error error

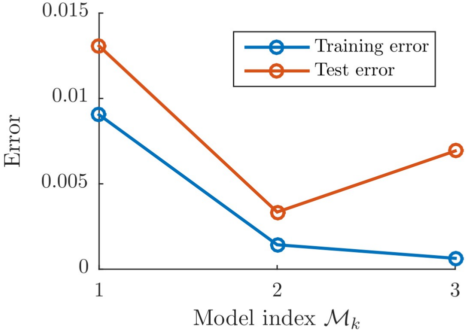
- Test error is obtaining by testing the trained models on new data



$$E^{\text{train}}_{\mathcal{M}_k} = \frac{1}{N^{\text{train}}} \sum_{i \in \mathcal{D}^{\text{train}}} (y_i - f_{\mathcal{M}_k}(x_i, \boldsymbol{w}))^2.$$

$$E^{\text{test}}_{\mathcal{M}_k} = \frac{1}{N^{\text{test}}} \sum_{i \in \mathcal{D}^{\text{test}}} (y_i - f_{\mathcal{M}_k}(x_i, \boldsymbol{w}))^2$$

Overfitting

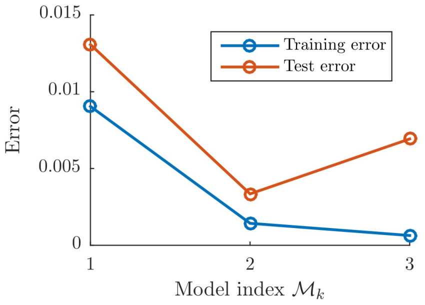


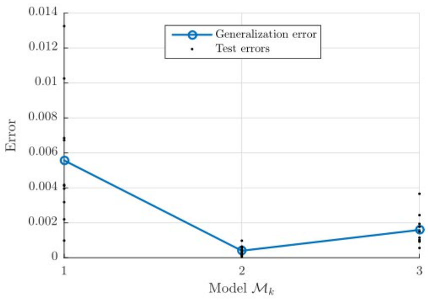
- **Overfitting** is that the training error usually decreases for overly complex models while the test error increases
- Test error is the more true error
- **Never, ever validate a model on the same data is was trained upon**

Generalization error

- The generalization error is the test error evaluated over infinitely many test sets
- **The generalization error is the "true performance" of our model**

$$E_{\mathcal{M}}^{\text{gen}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim p}\left[L(\boldsymbol{y}, \boldsymbol{f}_{\mathcal{M}}(\boldsymbol{x}))\right]$$
$$= \int d\boldsymbol{x} d\boldsymbol{y}\ p(\boldsymbol{x}, \boldsymbol{y}) L(\boldsymbol{y}, \boldsymbol{f}_{\mathcal{M}}(\boldsymbol{x}))$$





Basic cross-validation

- **Purpose: Estimate the generalization error**

Basic cross-validation

- **Purpose: Estimate the generalization error**
- 3 variants:
  - **Holdout:** Partitions dataset in two (training, test), approximate the generalization error based on the generated test set

$$\mathcal{D} = \mathcal{D}^{\text{train}} \cup \mathcal{D}^{\text{test}}$$

$$E_{\mathcal{M}}^{\text{gen}} \approx E_{\mathcal{M}}^{\text{test}}$$

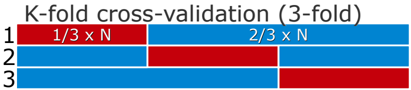
Basic cross-validation

- **Purpose: Estimate the generalization error**
- 3 variants:
  - **Holdout:** Partitions dataset in two (training, test), approximate the generalization error based on the generated test set

$$\mathcal{D} = \mathcal{D}^{\text{train}} \cup \mathcal{D}^{\text{test}}$$
$$E_{\mathcal{M}}^{\text{gen}} \approx E_{\mathcal{M}}^{\text{test}}$$

  - **K-fold:** Partitions dataset in K parts. Each part is a test set and the other K-1 training sets

$$\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_K$$
$$E_{\mathcal{M}}^{\text{gen}} \approx \frac{1}{K} \sum_{k=1}^{K} E_{\mathcal{M},k}^{\text{test}}$$



Basic cross-validation

- **Purpose: Estimate the generalization error**
- 3 variants:
  - **Holdout:** Partitions dataset in two (training, test), approximate the generalization error based on the generated test set

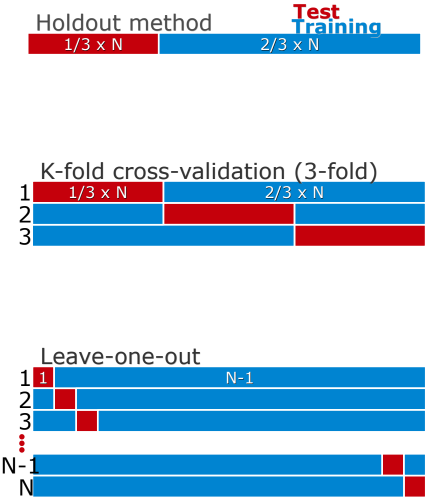
$$\mathcal{D} = \mathcal{D}^{\text{train}} \cup \mathcal{D}^{\text{test}}$$
$$E_{\mathcal{M}}^{\text{gen}} \approx E_{\mathcal{M}}^{\text{test}}$$

  - **K-fold:** Partitions dataset in K parts. Each part is a test set and the other K-1 training sets

$$\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_K$$
$$E_{\mathcal{M}}^{\text{gen}} \approx \frac{1}{K} \sum_{k=1}^{K} E_{\mathcal{M},k}^{\text{test}}$$

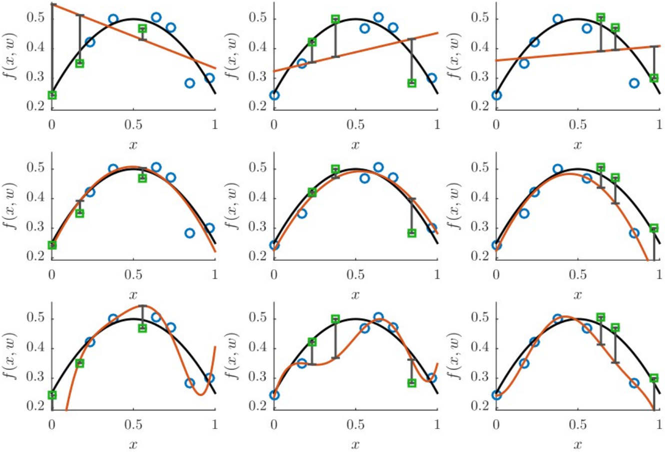
  - **Leave-one-out:** Partitions dataset into N parts. Let each observation be a test set and the other N-1 training sets (K-fold with K=N)

$$\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_N$$
$$E_{\mathcal{M}}^{\text{gen}} \approx \frac{1}{N} \sum_{k=1}^{N} E_{\mathcal{M},k}^{\text{test}}$$



Cross-validation (1-layer)

- **K=3 fold cross-validation for the three Linear-regression models**
  Vertically: The three models
  Horizontally: The three cross-validation folds

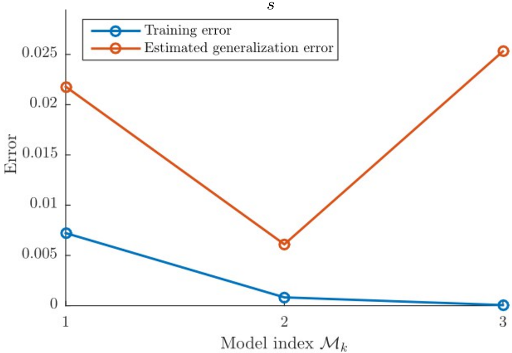


Cross-validation for model selection (1-layer)

- **Purpose: Select the best of S models**
- **The idea:**

- For each model, estimate the cross-validation error $\hat{E}^{\text{gen}}_{\mathcal{M}_1}, \ldots, \hat{E}^{\text{gen}}_{\mathcal{M}_S}$ using basic cross-validation.
- Select the optimal model $\mathcal{M}_{s^*}$ as that with the lowest error:

$$s^* = \arg\min_s \hat{E}^{\text{gen}}_{\mathcal{M}_s}$$



Cross-validation (1-layer)

- **K-fold cross-validation for model selection, the algorithm**

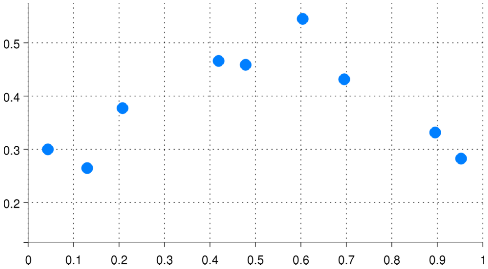
Algorithm 3: K-fold cross-validation for model selection

Require: K, the number of folds in the cross-validation loop
Require: $\mathcal{M}_1, \dots, \mathcal{M}_S$. The S different models to select between
Ensure: $\mathcal{M}_{s^*}$ the optimal model suggested by cross-validation
  for $k = 1, \dots, K$ splits **do**
    Let $\mathcal{D}_k^{\text{train}}$, $\mathcal{D}_k^{\text{test}}$ the k'th split of $\mathcal{D}$
    for $s = 1, \dots, S$ models **do**
      Train model $\mathcal{M}_s$ on the data $\mathcal{D}_k^{\text{train}}$
      Let $E_{\mathcal{M}_s,k}^{\text{test}}$ be the *test error* of the model $\mathcal{M}_s$ when it is *tested* on $\mathcal{D}_s^{\text{test}}$
    end for
  end for
  For each s compute: $\hat{E}_{\mathcal{M}_s}^{\text{gen}} = \frac{1}{K} \sum_{k=1}^{K} E_{\mathcal{M}_s,k}^{\text{test}}$
  Select the optimal model: $s^* = \arg\min_s \hat{E}_{\mathcal{M}_s}^{\text{gen}}$
  $\mathcal{M}_{s^*}$ is now the optimal model suggested by cross-validation

---

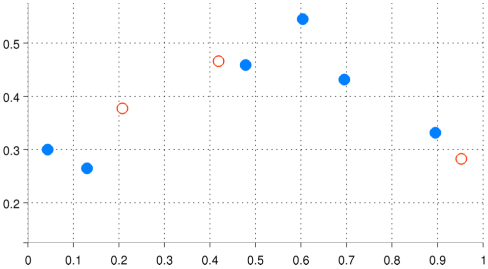
Holdout method

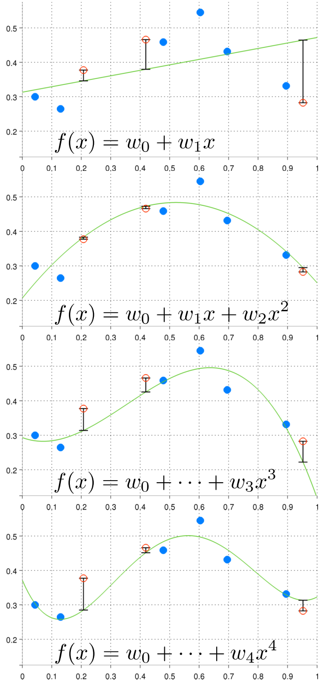
- Randomly choose a subset of data points to be in a **test set**
  - For example choose 1/3 of the points
- The rest is the **training set**



Holdout method

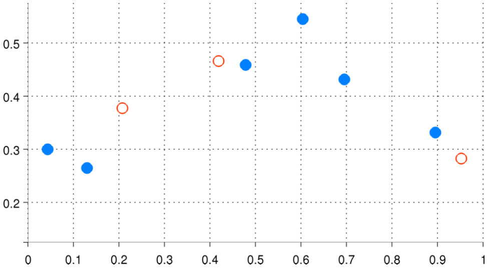
- Randomly choose a subset of data point to be in a **test set**
  - For example choose 1/3 of the points
- The rest is the **training set**





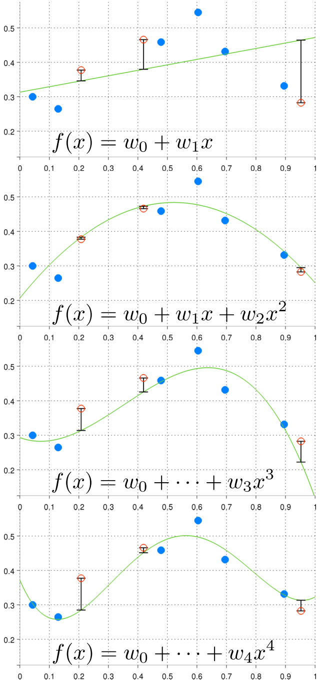
Holdout method

- Using the **training set**
  - Train the model for different complexities
- Using the **test set**
  - Compute the test error
- Choose the model with lowest **test error**



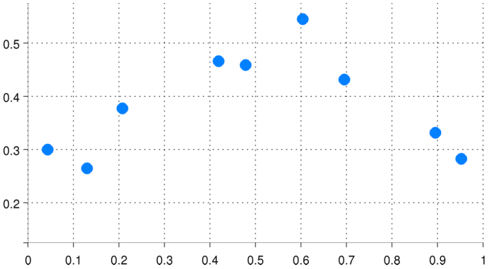
Holdout method

- Using the **training set**
  - Train the model for different complexities
- Using the **test set**
  - Compute the test error
- Choose the model with lowest **test error**



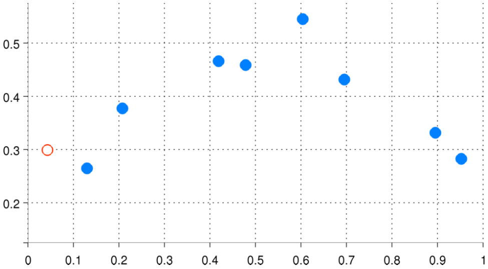
Leave-one-out

- Choose the first data point as a **test set**
- The rest is the **training set**



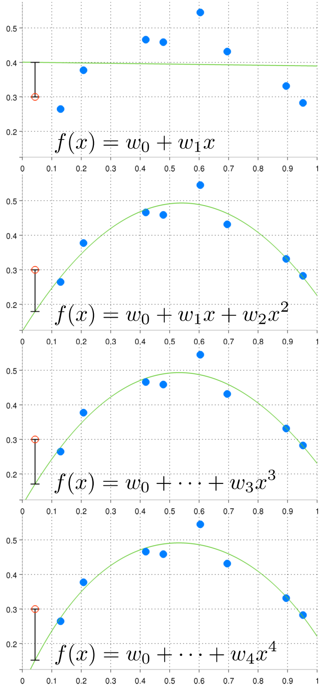
Leave-one-out

- Choose the first data point as a **test set**
- The rest is the **training set**



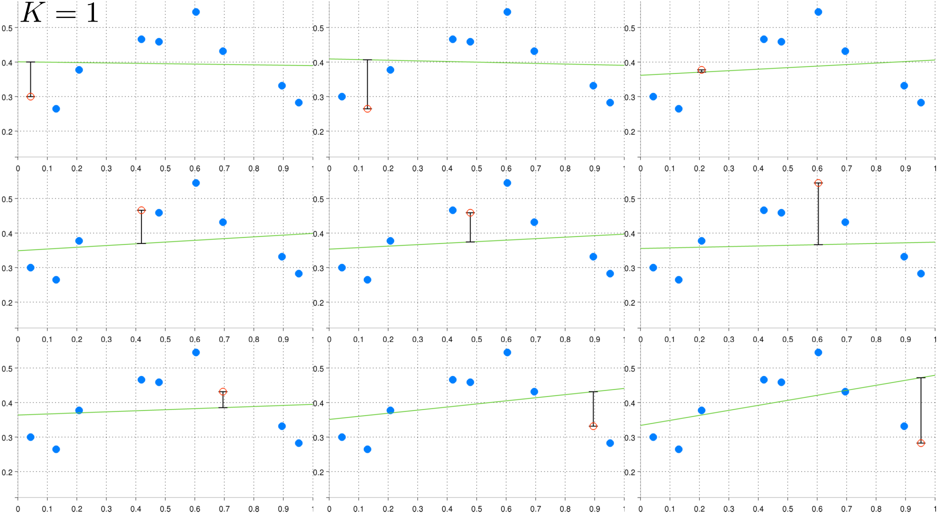
Leave-one-out

- Using the **training set**
  - Train the model for different complexities
- Using the **test set**
  - Compute the test error
- **Repeat for all data points**
  - All data points get to be test set
  - Compute **average test error**

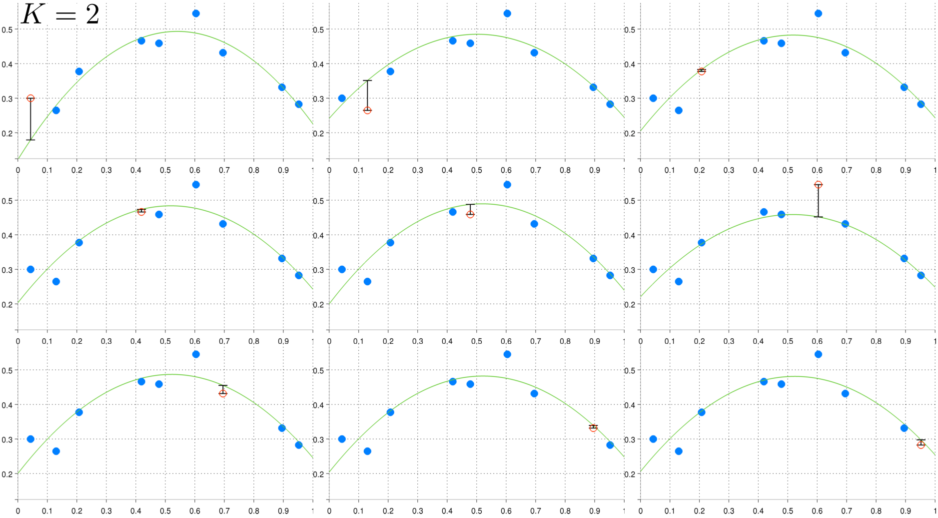


Leave-one-out

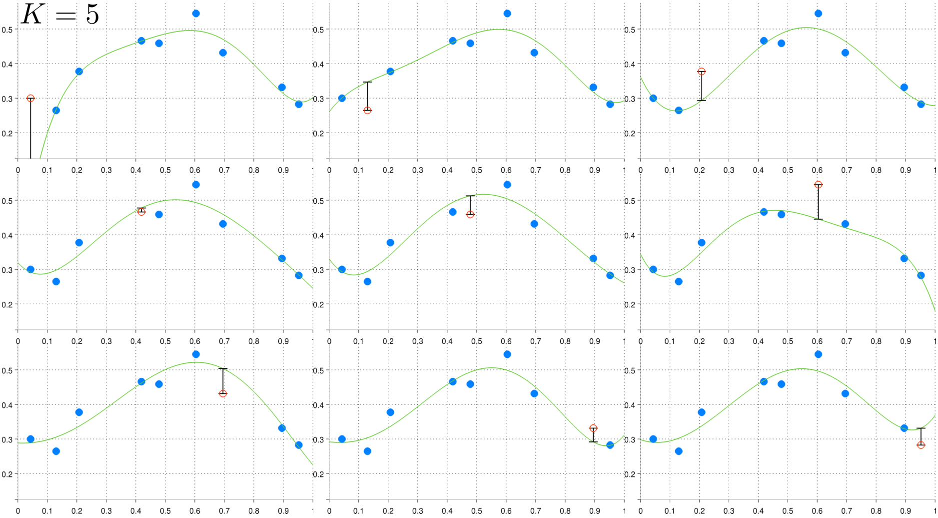
$K = 1$



Leave-one-out



Leave-one-out cross-validation

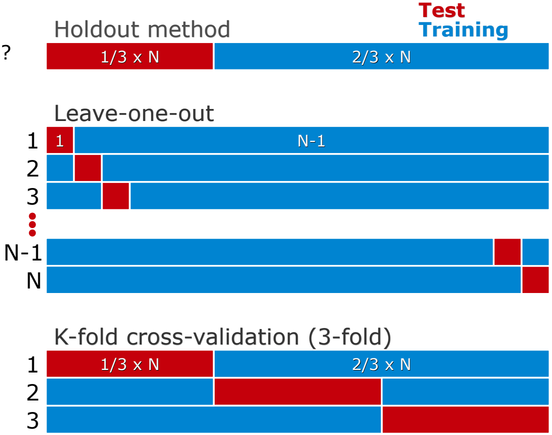


Leave-one-out

- Using the **training set**
  - Train the model for different complexities
- Using the **test set**
  - Compute the test error
- **Repeat for all data points**
  - All data points get to be test set
  - Compute **average test error**

Cross-validation methods

- Compare these three methods
  - What are their pros and cons?
- 10-fold cross-validation is very often used in pratice
  - Why do you think?

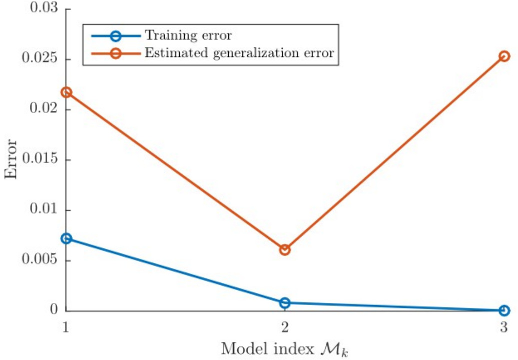


Cross-validation (1-layer, a problem?)

- For each model, estimate the cross-validation error $\hat{E}^{\text{gen}}_{\mathcal{M}_1}, \ldots, \hat{E}^{\text{gen}}_{\mathcal{M}_S}$ using basic cross-validation.

- Select the optimal model $\mathcal{M}_{s^*}$ as that with the lowest error:

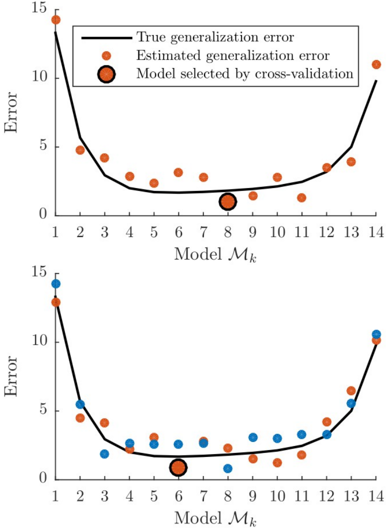
$$s^* = \arg\min_s \hat{E}^{\text{gen}}_{\mathcal{M}_s}$$



- **Is the generalization error the selected model (k=2) about 0.007?**

Cross-validation (1-layer, a problem?)

- **Same as before, just with more models. Is the error of the red dot a fair estimate of the generalization error?**



Two-layer cross-validation

- **Purpose: Select optimal model and estimate generalization error of optimal model**

Two-layer cross-validation

- **Purpose: Select optimal model and estimate generalization error of optimal model**
- **How?**
  - Recall ***"one layer cross-validation for model selection"***
  - This method returns a model (the best model)
  - We can consider **"one-layer cross-validation for model selection"** as a single model

Two-layer cross-validation

- **Purpose: Select optimal model and estimate generalization error of optimal model**
- **How?**
  - Recall ***"one layer cross-validation for model selection"***
  - This method returns a model (the best model)
  - We can consider **"one-layer cross-validation for model selection"** as a single model
- **Recall:**
  - **"Basic cross-validation for performance evaluation"** estimates the generalization error of a model

Two-layer cross-validation

- **Purpose: Select optimal model and estimate generalization error of optimal model**
- **How?**
  - Recall ***"one layer cross-validation for model selection"***
  - This method returns a model (the best model)
  - We can consider **"one-layer cross-validation for model selection"** as a single model
- **Recall:**
  - **"Basic cross-validation for performance evaluation"** estimates the generalization error of a model
- **Idea:** Apply **"basic cross-validation for performance evaluation"** on the **"one-layer cross-validation for model selection"**-model to estimate it's generalization error

Cross-validation (2-layer)

- **Two-layer cross-validation, the algorithm**

Algorithm 4: Two-level cross-validation

Require: K_1, K_2, folds in outer,inner cross-validation loop
Require: $\mathcal{M}_1, \ldots, \mathcal{M}_S$: The S different models to cross-validate
Ensure: $\hat{E}^{\text{gen}}$, the estimate of the generalization error
  for $i = 1, \ldots, K_1$ **do**
    Outer cross-validation loop. First make the outer split into K_1 folds
    Let $\mathcal{D}_i^{\text{par}}$, $\mathcal{D}_i^{\text{val}}$ the i'th split of $\mathcal{D}$
    for $j = 1, \ldots, K_2$ **do**
      Inner cross-validation loop. Use cross-validation to select optimal model
      Let $\mathcal{D}_j^{\text{train}}$, $\mathcal{D}_j^{\text{test}}$ by the j'th split of $\mathcal{D}_i^{\text{par}}$
      for $s = 1, \ldots, S$ **do**
        Train $\mathcal{M}_s$ on $\mathcal{D}_j^{\text{train}}$
        Let $E_{\mathcal{M}_s,j}^{\text{test}}$ be the *test error* of the model $\mathcal{M}_s$ when it is *tested* on $\mathcal{D}_j^{\text{test}}$
      end for
    end for
    For each s compute: $\hat{E}_s^{\text{gen}} = \frac{1}{K_2} \sum_{j=1}^{K_2} E_{\mathcal{M}_s,j}^{\text{test}}$
    Select the optimal model $\mathcal{M}^* = \mathcal{M}_{s^*}$ where $s^* = \arg\min_s \hat{E}_s^{\text{gen}}$
    Train $\mathcal{M}^*$ on $\mathcal{D}_i^{\text{par}}$
    Let E_i^{test} be the *test error* of the model $\mathcal{M}^*$ when it is *tested* on $\mathcal{D}_i^{\text{val}}$
  end for
  Compute the estimate of the generalization error: $\hat{E}^{\text{gen}} = \frac{1}{K_1} \sum_{i=1}^{K_1} E_i^{\text{test}}$

Feature subset selection

- Let's say we want to do linear regression
  - We have a large number of attributes

$$x_1, x_2, \ldots, x_M$$

- Using all attributes results in a too complex model
  - **Control complexity**: Choose a subset of attributes
    - Small subset = Simple model
    - Large subset = Complex model

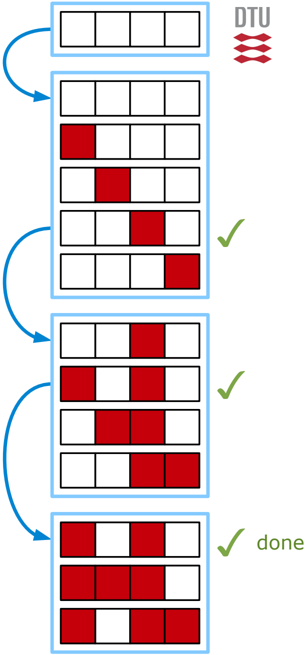
- **How many different ways can we choose a subset?**
  - How many models must be compared for
    - M=4
    - M=10
    - M=100

$$f(x) = w_0$$
$$f(x) = w_0 + w_1 x_1 + w_2 x_{27} + w_3 x_{88}$$
$$f(x) = w_0 + w_1 x_{19} + w_2 x_{76}$$
$$f(x) = w_0 + w_1 x_{19} + w_2 x_{76} + w_3 x_{88}$$
$$f(x) = w_0 + w_1 x_1 + w_2 x_{27} + w_3 x_{19}$$
$$f(x) = w_0 + w_1 x_{27} + w_2 x_{88}$$
$$\vdots$$

Sequential feature selection

Forward selection

- Start with no features
- Compute **cross-validation error** for
  - Current feature subset
  - All subsets equal to the current + one added feature
- Choose best subset
- Repeat until no further improvement



Sequential feature selection

Forward selection

- Start with no features
- Compute **cross-validation error** for
  - Current feature subset
  - All subsets equal to the current + one added feature
- Choose best subset
- Repeat until no further improvement

$$f(x) = w_0$$

$$f(x) = w_0$$

$$f(x) = w_0 + w_1 x_1$$

$$f(x) = w_0 + w_1 x_2$$

$$f(x) = w_0 + w_1 x_3$$

$$f(x) = w_0 + w_1 x_4$$

$$f(x) = w_0 + w_1 x_3$$

$$f(x) = w_0 + w_1 x_3 + w_2 x_1$$

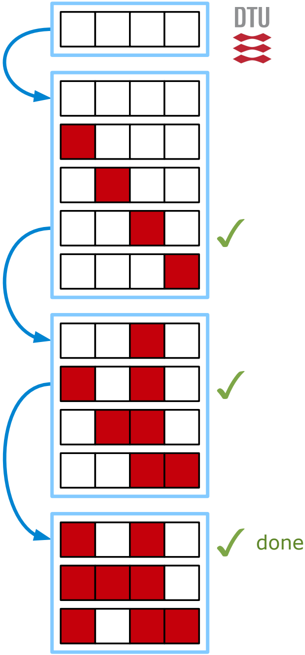
$$f(x) = w_0 + w_1 x_3 + w_2 x_2$$

$$f(x) = w_0 + w_1 x_3 + w_2 x_4$$

$$f(x) = w_0 + w_1 x_3 + w_2 x_1$$

$$f(x) = w_0 + w_1 x_3 + w_2 x_1 + w_3 x_2$$

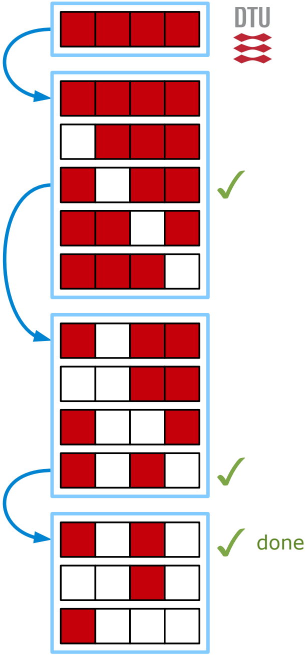
$$f(x) = w_0 + w_1 x_3 + w_2 x_1 + w_3 x_4$$



Sequential feature selection

Backward selection

- Start with all features
- Compute **cross-validation error** for
  - Current feature subset
  - All subsets equal to the current
    - one removed feature
- Choose best subset
- Repeat until no further improvement



Feature subset selection

- **How many models do we maximally have to evaluate by forward or backward selection?**

$$x_1, x_2, \ldots, x_M$$

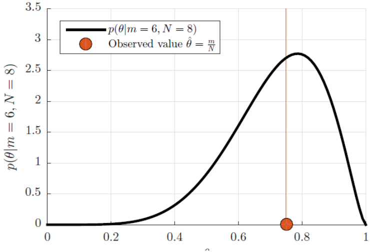
  - M=4
  - M=10
  - M=100

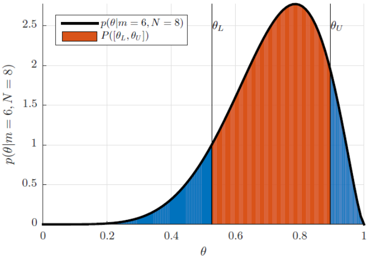
Statistical comparisons of classifiers

- **Credibility intervals**
- **Evaluation of a single classifier**
  - i.e., evaluate how significantly the classifier performs relative to random guessing
- **Comparing two classifiers**
  - i.e., is one classifier significantly better than another classifier

Credibility interval

$$P(\theta \text{ in the interval } [\theta_L, \theta_U]) = P([\theta_L, \theta_U]) = \int_{\theta_L}^{\theta_U} p(\theta|N,m)$$

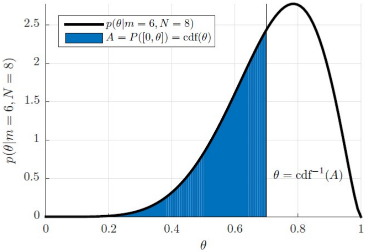




$$\mathrm{cdf}(\theta) = \int_0^{\theta} p(\theta'|N,m)d\theta'$$

$$\mathrm{cdf}(\theta_L) = \frac{\alpha}{2},$$

$$\mathrm{cdf}(\theta_U) = 1 - \frac{\alpha}{2}$$



Evaluation of a single classifier

$$p(\theta|m,N) = \frac{p(m|\theta,N)p(\theta)}{p(m|N)} = \frac{\theta^m(1-\theta)^{N-m}p(\theta)}{p(m|N)}$$

$$\text{Beta distribution: } \mathrm{Beta}(\theta|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\theta^{a-1}(1-\theta)^{b-1}$$

$$\text{Jeffrey prior: } p(\theta) = \mathrm{Beta}\left(\theta|\frac{1}{2},\frac{1}{2}\right) = \frac{1}{\Gamma(\frac{1}{2})^2}\theta^{-\frac{1}{2}}(1-\theta)^{-\frac{1}{2}}$$

Evaluation of a single classifier

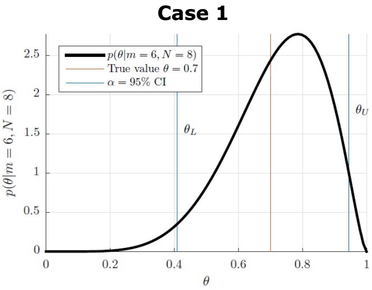
$$p(\theta|m,N) = \frac{\theta^m(1-\theta)^{N-m}p(\theta)}{p(m|N)} = \frac{1}{\Gamma(\frac{1}{2})^2}\frac{\theta^{m+\frac{1}{2}-1}(1-\theta)^{N-m+\frac{1}{2}-1}}{p(m|N)}$$

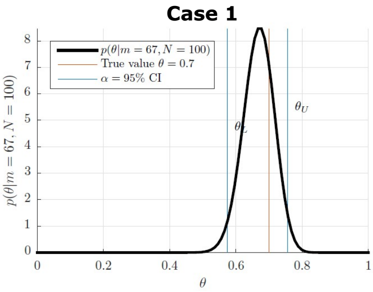
$$= \mathrm{Beta}(\theta|a,b), \quad a = m + \frac{1}{2}, \text{ and } b = N - m + \frac{1}{2}.$$

$$\theta_L = \mathrm{cdf}_B^{-1}\left(\frac{\alpha}{2}|a,b\right),$$

$$\theta_U = \mathrm{cdf}_B^{-1}\left(1-\frac{\alpha}{2}|a,b\right)$$

	N	m	a	b	θ_L	θ_U
Case 1	8	6	6.5	2.5	0.41	0.94
Case 2	100	67	67.5	33.5	0.57	0.76





Comparing two classifiers

$$E_A^{\text{gen}} - E_B^{\text{gen}} = \sum_{k=1}^{K} \frac{1}{K} z_k, \quad z_k = E_{A,k}^{\text{test}} - E_{A,k}^{\text{test}}$$

$$p(z_1, \ldots, z_K | u, \sigma^2) = \prod_{k=1}^{K} \mathcal{N}(z_k | u, \sigma^2)$$

$$p(u, \tau | \boldsymbol{z}) = \frac{p(\boldsymbol{z} | u, \tau) p(u, \tau)}{p(\boldsymbol{z})}$$

$$p(u, \tau | \boldsymbol{z}) \propto p(\boldsymbol{z} | u, \tau) p(u, \tau) = \left[\prod_{k=1}^{K} \mathcal{N}(z_k | u, \tau) \right] \frac{1}{\tau}$$

Comparing two classifiers

$$p(u|z) = \int p(u,\tau|z)d\tau \propto \int \frac{1}{\tau}\prod^{K} \mathcal{N}(z_k|u,\tau)d\tau \propto \left(1+\frac{1}{\nu}\left[\frac{u-\bar{x}}{\tilde{\sigma}}\right]^2\right)^{-\frac{\nu+1}{2}}$$

$$p_{\text{stud}-t}(x|\nu,\mu,\sigma) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)\sqrt{\pi\nu\sigma^2}}\left(1+\frac{1}{\nu}\left[\frac{x-\mu}{\sigma}\right]^2\right)^{-\frac{\nu+1}{2}}$$

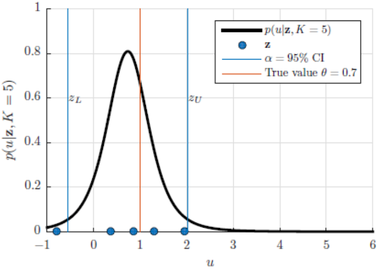
$\bar{z} = \frac{1}{K}\sum_{k=1}^{K} z_k$, $\nu = K-1$ and $\tilde{\sigma} = \sqrt{\sum_{k=1}^{K}\frac{(z_k-\bar{z})^2}{K(K-1)}}$.

$$z_L = \mathrm{cdf}_{st}^{-1}\left(\frac{\alpha}{2}|\nu,\bar{z},\tilde{\sigma}\right),$$

$$z_U = \mathrm{cdf}_{st}^{-1}\left(1-\frac{\alpha}{2}|\nu,\bar{z},\tilde{\sigma}\right)$$

	K	ν	$\bar{z}$	$\tilde{\sigma}$	θ_L	θ_U
Case 1	5	4	0.7340	0.46	-0.55	2.02
Case 2	10	9	1.4960	0.40	0.60	2.40

Case 1



Case 2