DTU

02450 Introduction to Machine Learning and Data Mining

Week 6: Overfitting, cross-validation and Nearest Neighbor

Georgios Arvanitidis

11 March 2025

DTU Compute, Technical University of Denmark

Today

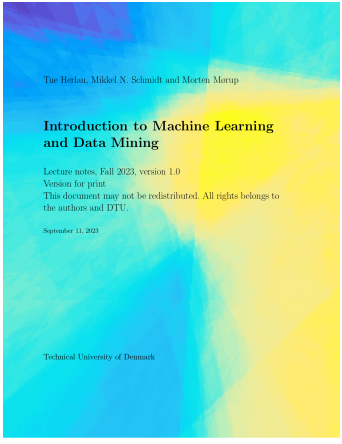
Feedback Groups of the day:

Konstantina Gkopi, Tomás García-Cardo Raskovsky, Astrid Siemens Lorenzen, Tobias Friis Martinsen, Gustav Hylsberg Jacobsen, Landon Scott Badstibner, Karoline Fonager Ulsøe Johansen, Oliver Svanholm Kliim, Bertil Emanuel Brinck Billig, Nick Gholami, Nils Maasch, Alicia Llorente Herrero, Mathias Gilbert, Lory Soumphonphakdy, Maliha Brar, Saad Mahmood, Silja Bogensberger, Eliot Abraham Colas, Nikolaos Iliakis, Diego ., Christian Hoà Crone, Sina Rahimian, Sofie Karoline Traun, Leif Erik Stolte, Ida Lindhardtsen, Cesia Niquen Tapia, Dimitar Ilev, Naia Nikoline Due, Farras Muhammad Fikry, He Shaoliang, Bertram Kjærholm Foldberg Bendsen, Shannon Taylor Mascard, Christoffer Shane Rask, Frederik Andersen, Maisha Tumelo Molepo, Joel Farré Cortes, Raquel Chaves Martinez, Natasha Dimple, Thit Thomsen, Adam Harteg Jacobsen, Mathias Sand, Kostas Papadopoulos

Reading/homework material:

Chapter 10, 12

P10.1, P10.2, P12.1



Lecture Schedule

1. Introduction
   4 February: C1,C2

Data: Feature extraction, and visualization

2. Summary statistics, similarity and visualization
   11 February: C4,C7
3. Computational linear algebra and PCA
   18 February: C3
4. Probability and probability densities
   25 February: C5, C6

Supervised learning: Classification and regression

5. Decision trees and linear regression
   4 March: C8, C9 **(Project 1 due 6 March at 17:00)**
6. **Overfitting, cross-validation and Nearest Neighbor**
   11 March: C10, C12
7. Performance evaluation, Bayes, and Naive Bayes
   18 March: C11, C13
8. Artificial Neural Networks and Bias/Variance
   25 March: C14, C15
9. AUC and ensemble methods
   1 April: C16, C17

Unsupervised learning: Clustering and density estimation

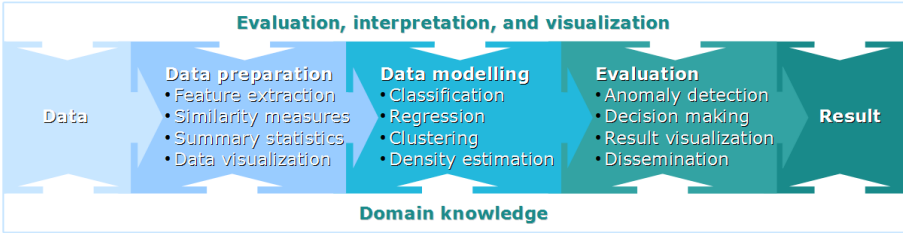
10. K-means and hierarchical clustering
    8 April: C18 **(Project 2 due 10 April at 17:00)**
11. Mixture models and density estimation
    22 April: C19, C20
12. Association mining
    29 April: C21

Recap

13. Recap and discussion of the exam
    6 May: C1-C21

Online help: Piazza
Videos of lectures: `https://panopto.dtu.dk`
Streaming of lectures: Zoom (link on DTU Learn)

Learning Objectives



Learning Objectives

- 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
- Explain how K-Nearest Neighbours can be used to classify data

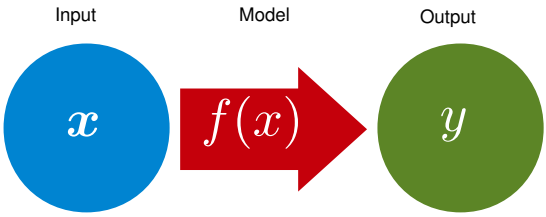
Roadmap

- Introduce errors
  - Training error
  - Test error
  - Generalization error
- Introduce cross-validation
  - **Basic cross validation** for **performance evaluation**
  - **Cross-validation** for **model selection**
  - **Two-level cross-validation** for **model selection and performance evaluation**
- Nearest Neighbor methods

Supervised learning

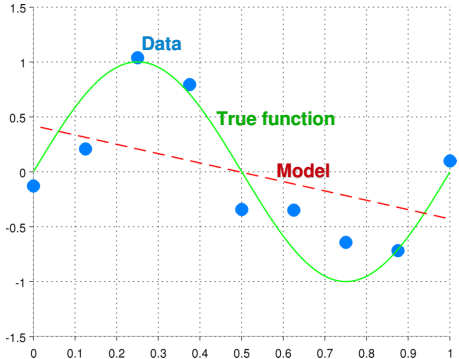
Mapping between domains

- Classification: Discrete output
- Regression: Continuous output



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

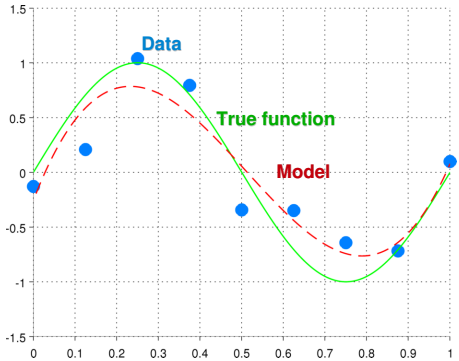
- Bad fit
- **Too simple model**



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

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

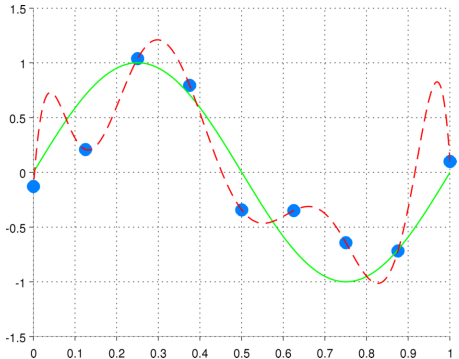
- Reasonable fit
- **Reasonable model**



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

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

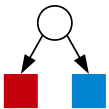
- Perfect fit
- **Too complex model**

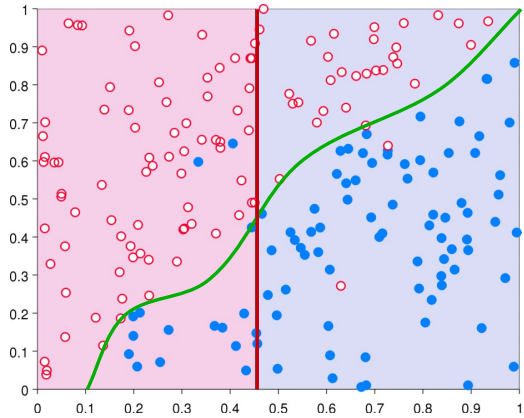


$$f(x) = w_0 + w_1 x + \cdots + w_8 x^8$$

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

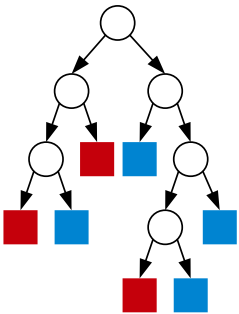
- Bad fit
- **Too simple model**

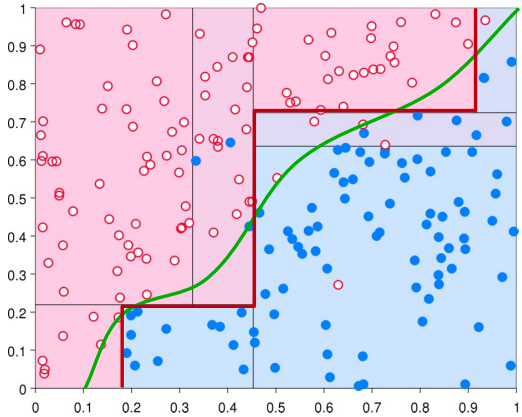




Why are there "multiple models"? Example: Classification tree

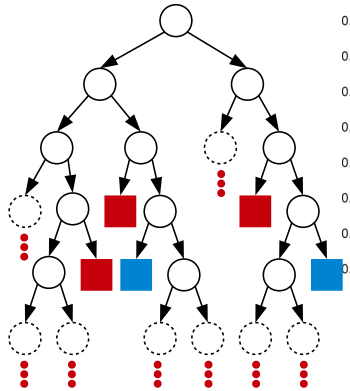
- Reasonable fit
- **Reasonable model**

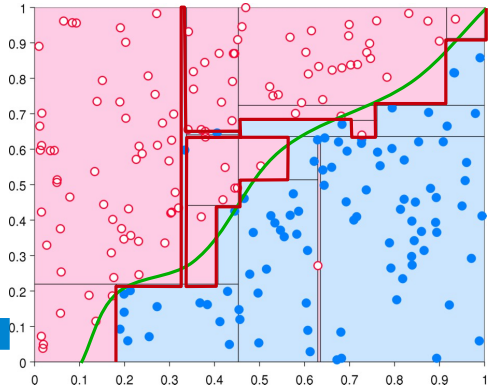




Why are there "multiple models"? Example: Classification tree

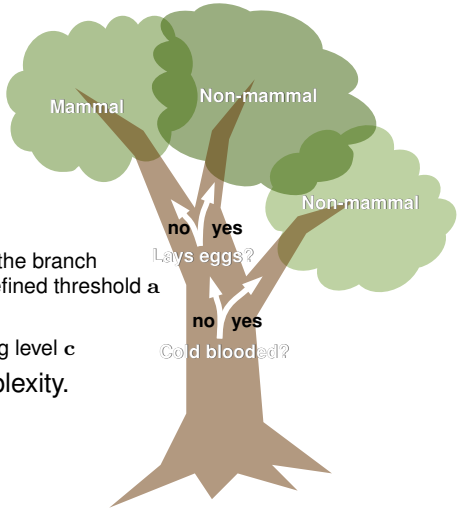
- Perfect fit
- **Too complex model**



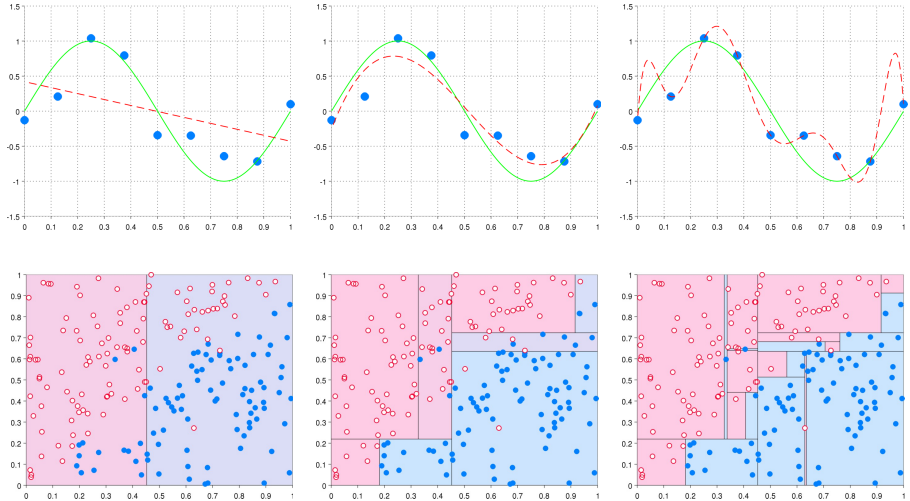


Decision trees

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

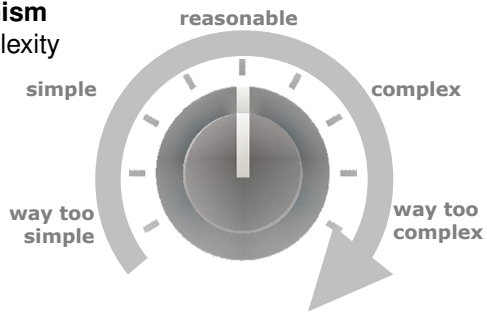


Model Overfitting



Control the model complexity

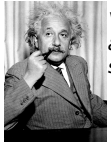
- Find **parameter** or **mechanism** in model that controls complexity



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)

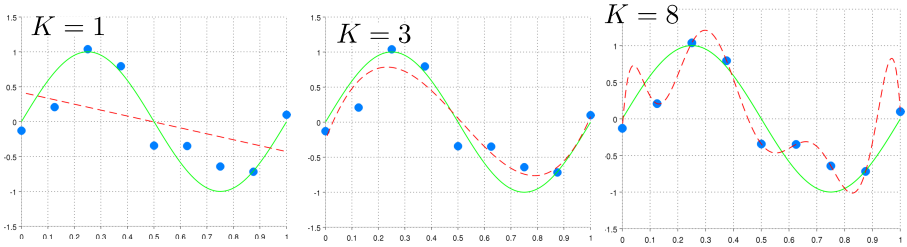
https://commons.wikimedia.org/wiki/File:William_of_Ockham.png



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

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

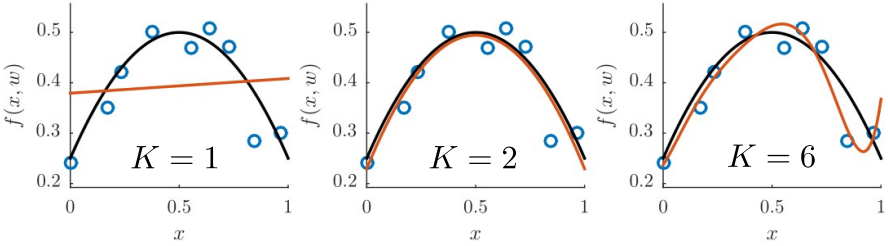


Solution: Assess model performance correctly and select best model

Training error

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

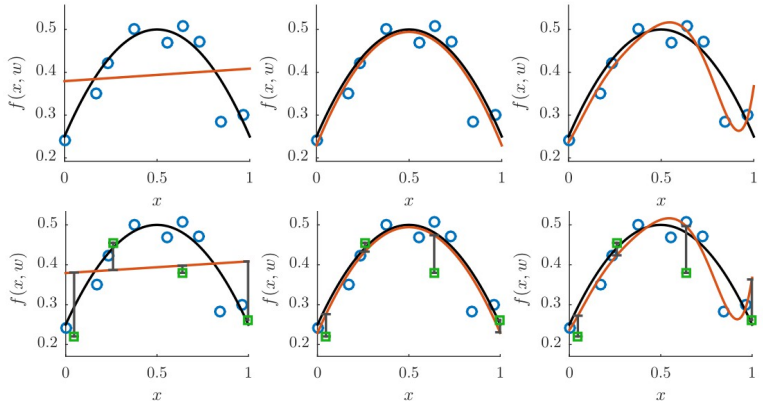
$$\mathcal{M}_1 = \{\text{1'st order polynomial}\}$$
$$\mathcal{M}_2 = \{\text{2'nd order polynomial}\}$$
$$\mathcal{M}_3 = \{\text{6'th order polynomial}\}$$



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

Test error

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

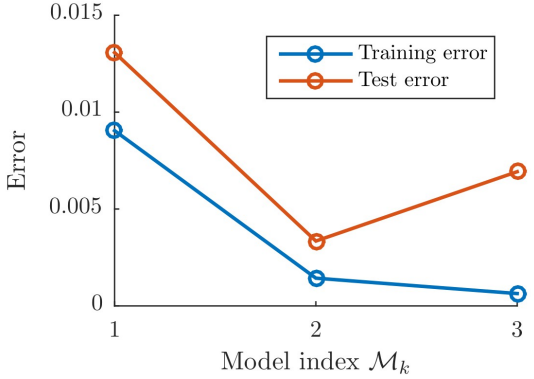


$$E_{\mathcal{M}_k}^{\text{train}} = \frac{1}{N^{\text{train}}} \sum_{i \in \mathcal{D}^{\text{train}}} (y_i - f_{\mathcal{M}_k}(x_i, \boldsymbol{w}))^2.$$

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

Overfitting

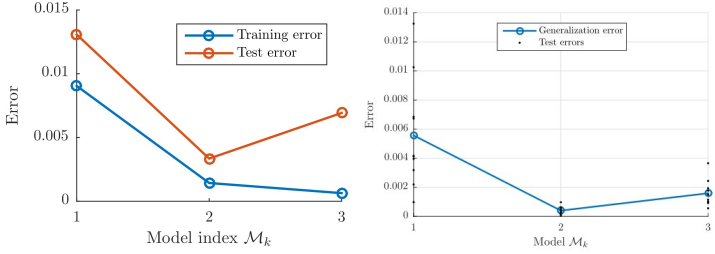
- **Overfitting** is that the training error usually decreases for overly complex models while the test error increases.
- Test error is more representative of the true error.
- **Never, ever validate a model on the same data it was trained upon!**



Generalization error

- The generalization error is the test error evaluated over an infinitely large test set
- The generalization error is the "true performance" of the trained model
  - Train model $\mathcal{M}$ on the available dataset $\mathcal{D}$ to get prediction rule $\boldsymbol{f}_{\mathcal{M}}$
  - Compute $E_{\mathcal{M}}^{\text{gen}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}\left[L(\boldsymbol{y}, \boldsymbol{f}_{\mathcal{M}}(\boldsymbol{x}))\right]$
- If we somehow had many test sets $\mathcal{D}_1, \ldots, \mathcal{D}_k$

$$E_{\mathcal{M}}^{\text{gen}} \approx \frac{1}{K}\sum_{k=1}^{K} E_{\mathcal{M},\mathcal{D}_k}^{\text{test}}$$



Basic cross-validation

- **Purpose: Estimate the generalization error**

Basic cross-validation

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

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

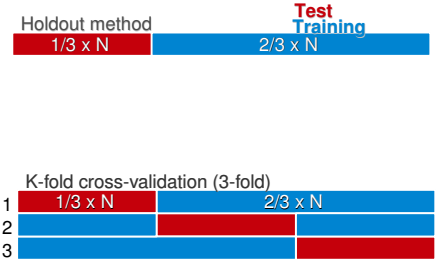
Basic cross-validation

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

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

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

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



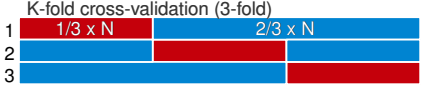
Basic cross-validation

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

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

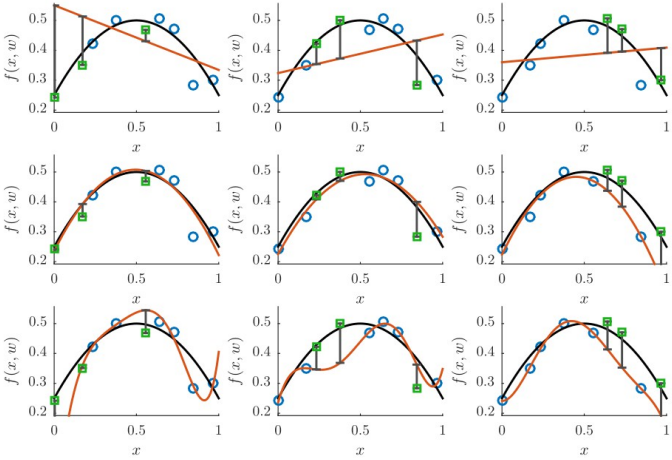


  - **Leave-one-out**: Partitions dataset into N parts. Let each observation be a test set and the other $N-1$ training sets (K-fold with $K=N$)

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

Cross-validation (1-layer)

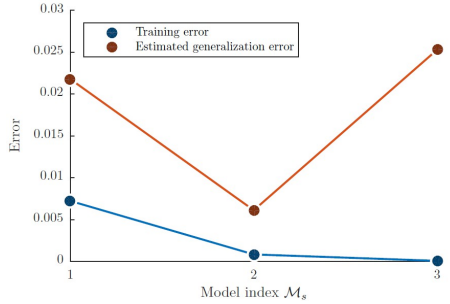
- $K = 3$ fold cross-validation for the three linear regression models
  Vertically: The three models
  Horizontally: The three cross-validation folds



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)

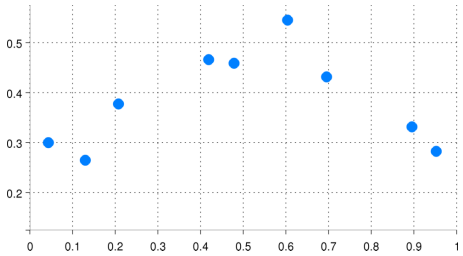
- **K-fold cross validation for model selection, the algorithm**

Algorithm 4: K-fold cross-validation for model selection

Require: K, the number of folds in the cross-validation loop
Require: $\mathcal{M}_1, \ldots, \mathcal{M}_S$. The S different models to select between
Ensure: $\mathcal{M}_{s^*}$ the optimal model suggested by cross-validation
  for $k = 1, \ldots, K$ splits **do**
    Let $\mathcal{D}_k^{\text{train}}$, $\mathcal{D}_k^{\text{test}}$ the k'th split of $\mathcal{D}$
    for $s = 1, \ldots, S$ 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 $\mathcal{D}_k^{\text{test}}$
    end for
  end for
  For each s compute: $\hat{E}_{\mathcal{M}_s}^{\text{gen}} = \sum_{k=1}^{K} \frac{N_k^{\text{test}}}{N} E_{\mathcal{M}_s,k}^{\text{test}}$
  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

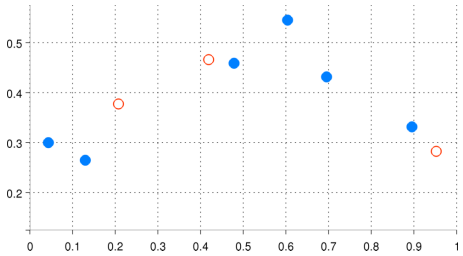
Holdout method

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



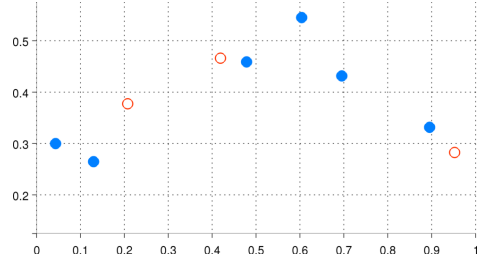
Holdout method

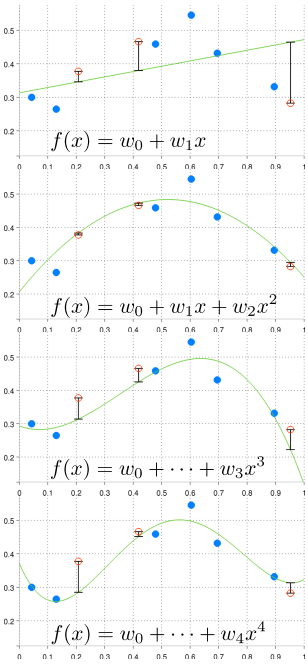
- Randomly choose a subset of data ppitns to be in a **test set**
  – For example choose 1/3 of the points
- The rest is the **training set**



Holdout method

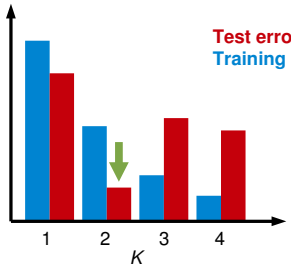
- Using the **training set**
  – Train the model for different complexities
- Using the **test set**
  – Compute the test error
- Choose the model with lowest **test error**

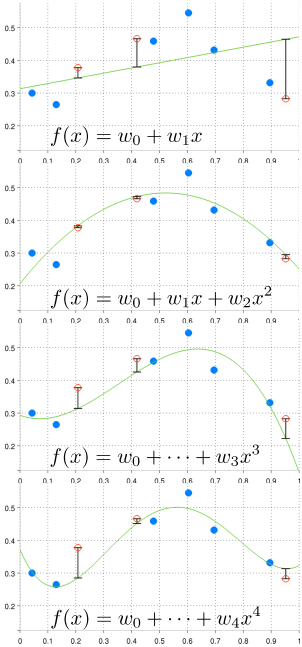




Holdout method

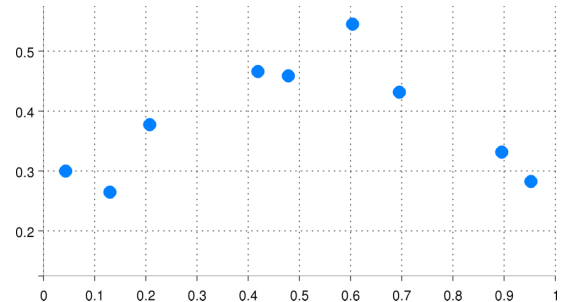
- Using the **training set**
  – Train the model for different complexities
- Using the **test set**
  – Compute the test error
- Choose the model with lowest **test error**





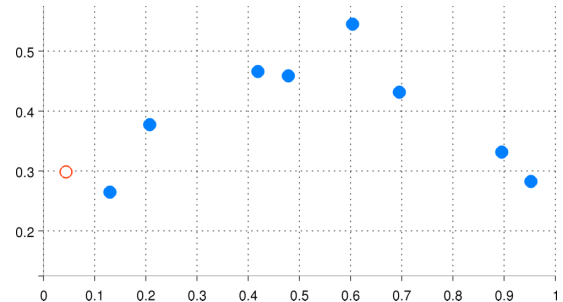
Leave-one-out

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



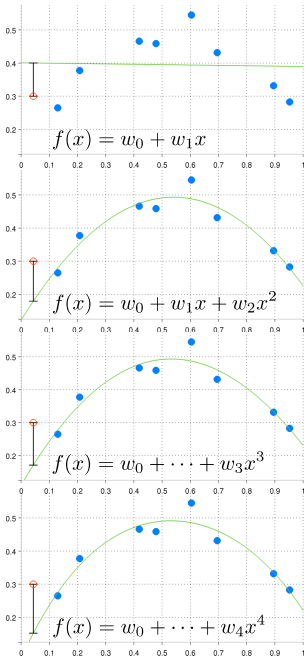
Leave-one-out

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

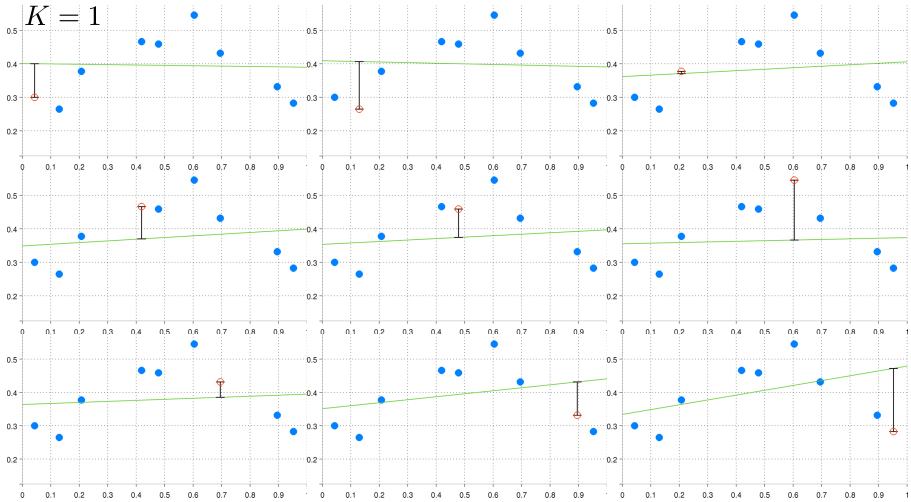


Leave-one-out

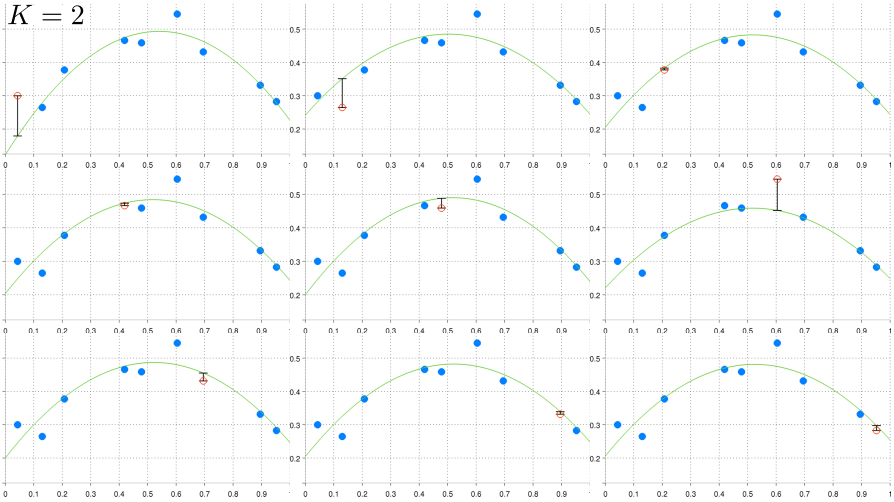
- Using the **training set**
  – Train the model for different complexities
- Using the **test set**
  – Compute the test error
- **Repeat for all data points**
  – All data points get to be test set
  – Compute **average test error**



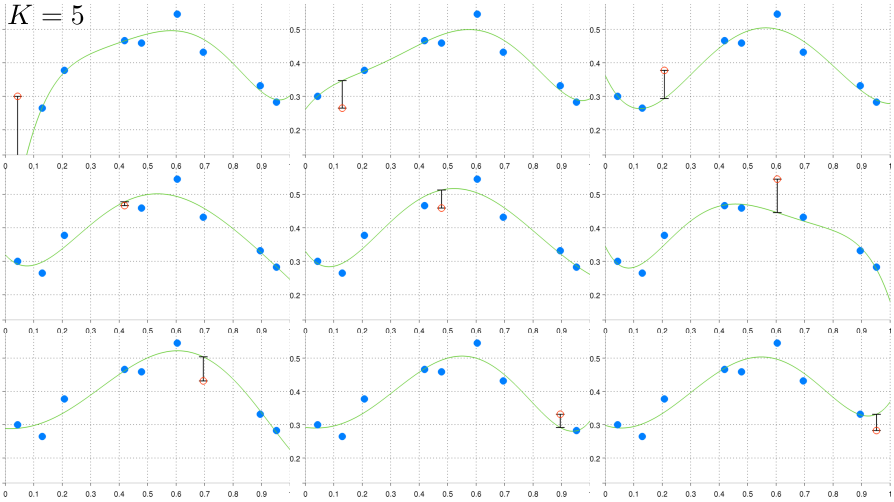
Leave-one-out



Leave-one-out



Leave-one-out

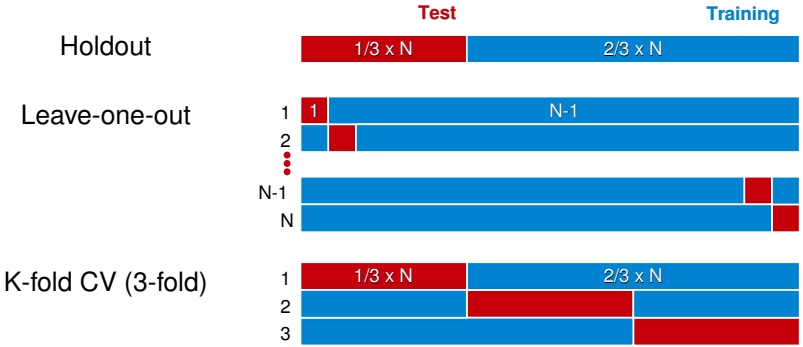


Leave-one-out

- 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**

So which method should I choose?

- Holdout is computationally least intensive, but not very data efficient
- Leave-one-out is very computationally intensive, and the estimates are highly correlated
- Recommendation: K-fold for $K = 5, 10$ or holdout if problem very large.



Quiz 1, Cross validation (Spring 2012)

Feature(s)	Training MSE	Test MSE
A	2.0	2.2
B	1.8	1.9
C	1.6	1.7
D	1.9	2.1
A and B	1.7	2.0
A and C	1.3	1.8
A and D	1.4	1.5
B and C	1.5	1.6
B and D	1.7	1.8
C and D	1.5	2.0
A and B and C	1.2	2.1
A and B and D	1.1	2.0
A and C and D	1.0	2.3
B and C and D	1.2	2.5
A and B and C and D	0.9	2.8

Consider a neural network regression problem with four attributes denoted A, B, C and D. A neural network with two hidden units is trained using different combinations of the attributes. The neural network is trained on 50% of the data and tested on the remaining 50% of the data using the hold-out method. In the table is given the training and test performance of the neural network for the different combinations of attributes. Which of the following statements is *incorrect*

A. Hold-out 50% of the data is more computationally efficient than 5 fold cross-validation.

B. Leave one-out cross-validation gives a poor estimate of the generalization error as only one observation is part of the test set at a time.

C. The size of the training set in 10 fold cross-validation is larger than the size of the training set in 5 fold cross-validation.

D. Not all observations are used for testing using the hold-out method.

E. Don't know.

Forward selection

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

- Suppose we want to do linear regression
- As usual, we have M attributes

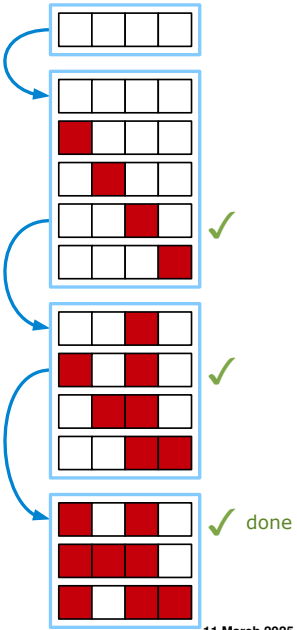
$$x_1, x_2, \ldots, x_M$$

- We can control model complexity by using a subset of attributes
  - Large subset: Complex model; hard to interpret
  - Small subset: Too simple model
- In general, we can construct 2^M models; often far too many
- Sequential feature selection allow us to efficiently search the model space

Sequential feature selection

Forward selection

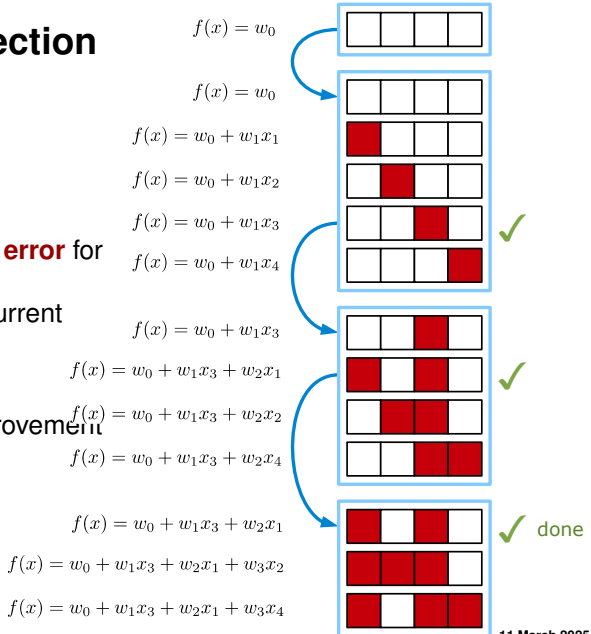
- Start with no feature
- Compute **cross-validation error** for
  – Current feature subset
  – All subsets equal to the current
  + one added feature
- Choose best subset
- Repeat until no further improvement



Sequential feature selection

Forward selection

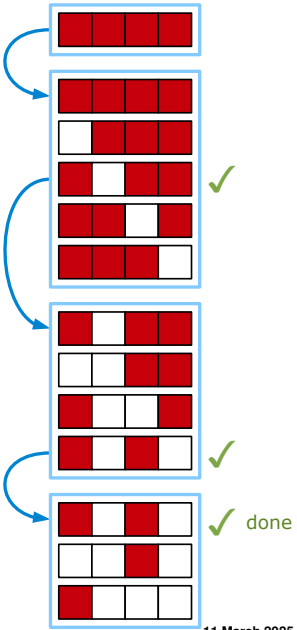
- Start with no feature
- Compute **cross-validation error** for
  – Current feature subset
  – All subsets equal to the current + one added feature
- Choose best subset
- Repeat until no further improvement



Sequential feature selection

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



Quiz 2, Forward selection (Spring 2012)

Feature(s)	Training MSE	Test MSE
A	2.0	2.2
B	1.8	1.9
C	1.6	1.7
D	1.9	2.1
A and B	1.7	2.0
A and C	1.3	1.8
A and D	1.4	1.5
B and C	1.5	1.6
B and D	1.7	1.8
C and D	1.5	2.0
A and B and C	1.2	2.1
A and B and D	1.1	2.0
A and C and D	1.0	2.3
B and C and D	1.2	2.5
A and B and C and D	0.9	2.8

Consider a neural network regression problem with four attributes denoted A, B, C and D where a neural network with two hidden units is trained using different combinations of the attributes. The table gives the training and test performance of the neural network for different combinations of attributes. Which of the following statements is *correct*?

A. Using a forward selection strategy feature B and C would be selected as the optimal model.

B. Using a forward selection strategy features A and D would be selected as the optimal model.

C. Using a forward selection strategy features A and C and D would be selected as the optimal model.

D. Using a forward selection strategy features A and B and C would be selected as the optimal model.

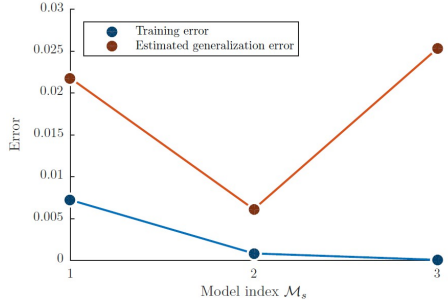
E. Don't know.

A problem with 1 level cross-validation?

- 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:

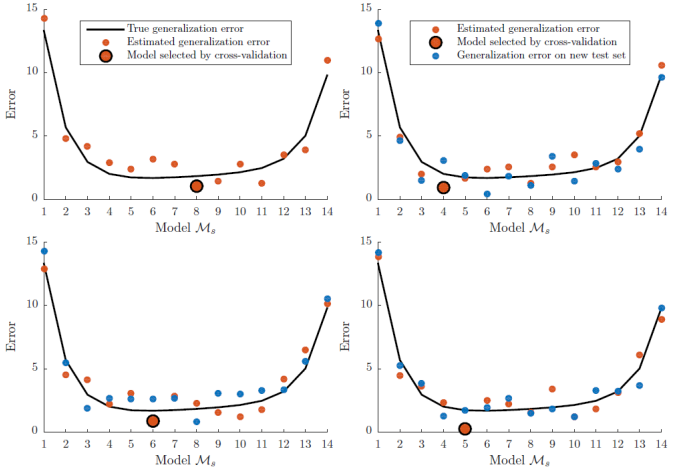
$$s^* = \arg\min_s \hat{E}^{\text{gen}}_{\mathcal{M}_s}$$

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



A problem with 1 level cross-validation?

- **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 returs 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 returs a model (the best model)
  - We can consider "one-layer cross-validation for model selection" as a single model
- **Recall**:
  - "Basic cross-validation for model performance evaluation" estimates the generalization error of a model

Two-layer cross-validation

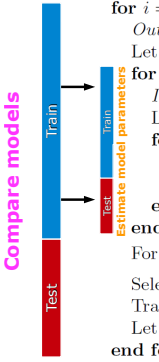
- **Purpose: Select optimal model and estimate generalization error of optimal model**
- **How?**
  - Recall "one layer cross-validation for model selection"
  - This method returs a model (the best model)
  - We can consider "one-layer cross-validation for model selection" as a single model
- **Recall**: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

Cross-validation (2-layer)

- Two-layer cross-validation, the algorithm

Algorithm 5: Two-level cross-validation

Require: K_1, K_2, folds in outer, and inner cross-validation loop respectively
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 $\mathcal{D}_i^{\text{par}}$, $\mathcal{D}_i^{\text{test}}$ be 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 $\mathcal{D}_j^{\text{train}}$, $\mathcal{D}_j^{\text{val}}$ be 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 $E_{\mathcal{M}_s,j}^{\text{val}}$ be the *validation error* of the model $\mathcal{M}_s$ when it is *tested* on $\mathcal{D}_j^{\text{val}}$
end for
end for
For each s compute: $\hat{E}_s^{\text{gen}} = \sum_{j=1}^{K_2} \frac{|\mathcal{D}_j^{\text{val}}|}{|\mathcal{D}_i^{\text{par}}|} E_{\mathcal{M}_s,j}^{\text{val}}$
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 $\mathcal{D}_i^{\text{test}}$
end for
Compute the estimate of the generalization error: $\hat{E}^{\text{gen}} = \sum_{i=1}^{K_1} \frac{|\mathcal{D}_i^{\text{test}}|}{N} E_i^{\text{test}}$



Quiz 3, two-level cross-validation (Spring 2016)

Consider a classification tree model applied to a dataset of $N = 1000$ observations. Suppose we wish to both select the optimal pruning level and estimate the generalization error of the classification tree model by cross-validation. To simplify the problem, we only consider 3 possible pruning levels:

$$3, 4, 5.$$

We opt for a two-level cross-validation strategy in which we use an inner loop of $K_2 = 5$-fold cross-validation to estimate the optimal pruning level and an outer loop of $K_1 = 10$ fold cross-validation to estimate the generalization error. That is, for each of the K_1 outer folds, the dataset is divided into a validation set and a parameter estimation set on which K_2-fold cross-validation is used to select the optimal pruning level for this outer fold.

How many models do we *train* using 2-level cross-validation?

A. 50

B. 150

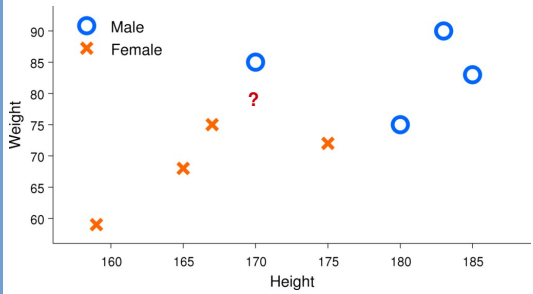
C. 160

D. 180

E. Don't know.

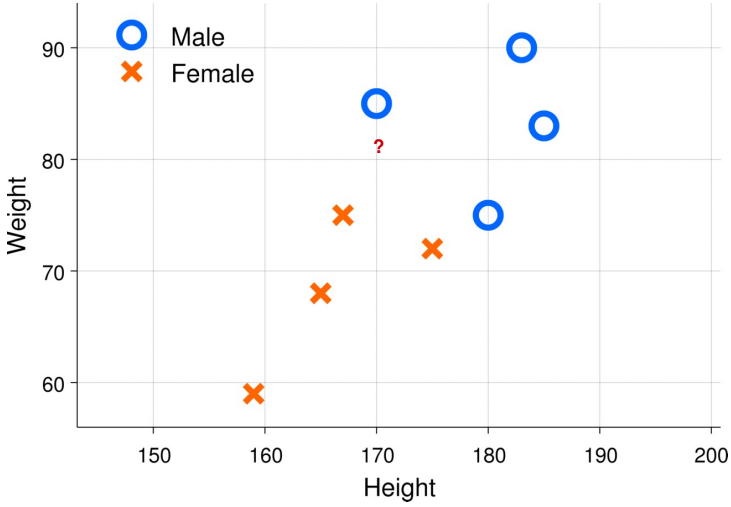
Classify gender based on height and weight

	Height	Weight	Gender
1	183	90	Male
2	180	75	Male
3	170	85	Male
4	185	83	Male
5	159	59	Female
6	167	75	Female
7	165	68	Female
8	175	72	Female
9	171	82	?



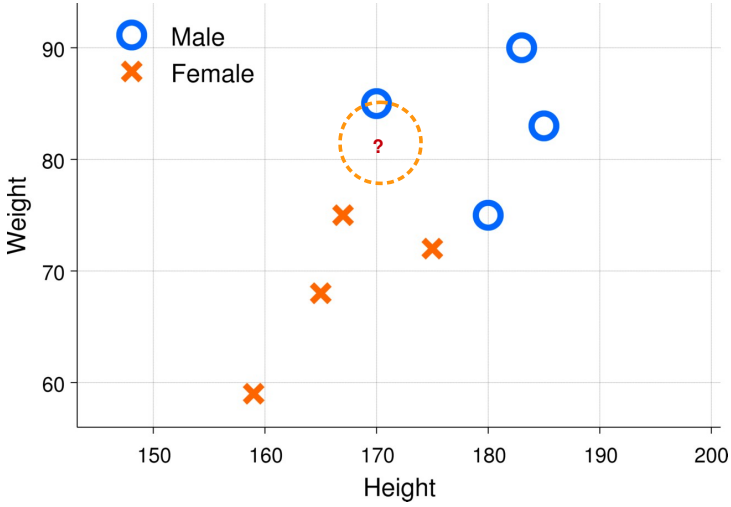
Nearest neighbor classifier

- 1 nearest neighbor



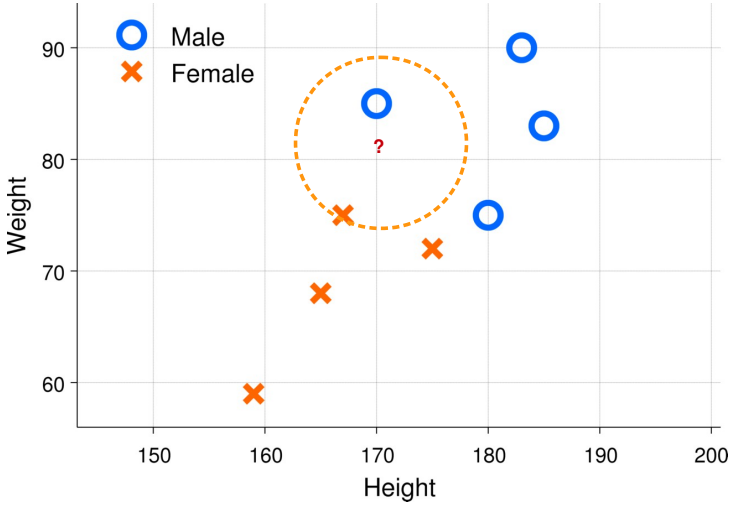
Nearest neighbor classifier

- 1 nearest neighbor



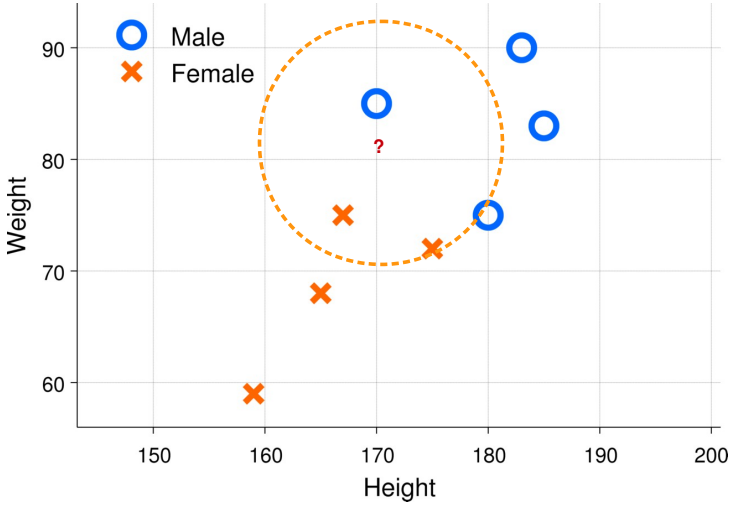
Nearest neighbor classifier

- 2 nearest neighbor



Nearest neighbor classifier

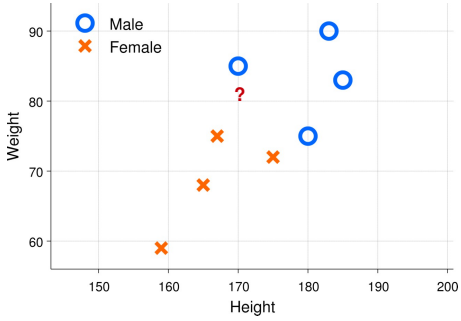
- 3 nearest neighbor



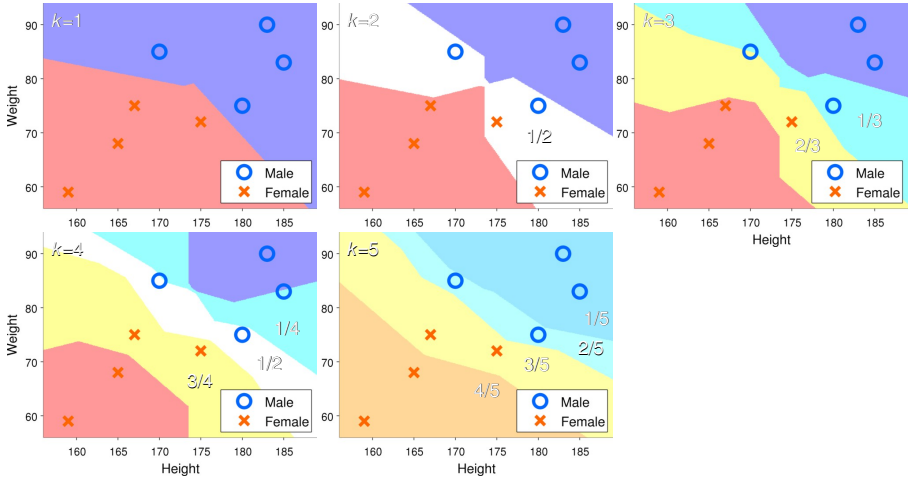
Nearest neighbor classifier

- Choose:
  – The number of neighbors, K
  – A distance measure

1. Compute distance to all other data objects
2. Find the K nearest data objects
3. Classify according to majority of neighbors



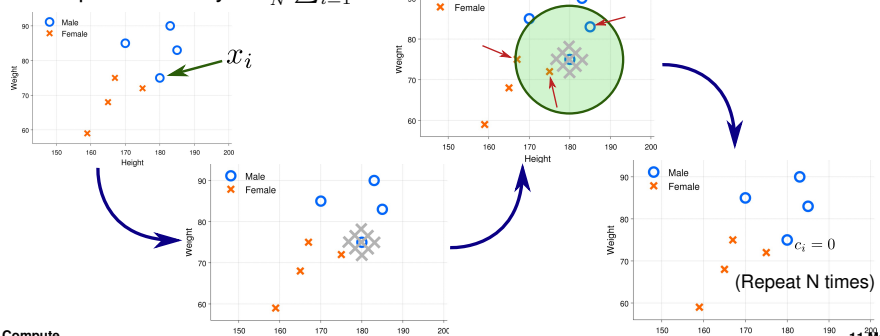
Nearest neighbor decision surface



KNN with leave-one-out CV

Leave-one-out CV is convenient with KNN

- For each observation x_i
  - Temporarily remove x_i
  - Find K nearest neighbors around x_i (not x_i itself)
  - Determine whether x_i is classified correctly based on this neighborhood, $c_i = 0$ or $c_i = 1$
- Compute accuracy as $\frac{1}{N}\sum_{i=1}^{N} c_i$



Quiz 4, KNN (Spring 2011)

- For each observation x_i
  - Temporarily remove x_i
  - Find K nearest neighbors around x_i (not x_i itself)
  - Determine whether x_i is classified correctly based on this neighborhood

		Diabetic				Normal			
		$D1$	$D2$	$D3$	$D4$	$N1$	$N2$	$N3$	$N4$
Diabetic	$D1$	0	58.5	51.6	18.1	38.0	52.5	71.7	50.7
	$D2$	58.5	0	32.1	72.6	50.5	65.0	13.2	63.8
	$D3$	51.6	32.1	0	60.5	28.4	32.9	45.3	56.3
	$D4$	18.1	72.6	60.5	0	45.9	60.4	79.8	56.8
Normal	$N1$	38.0	50.5	28.4	45.9	0	17.5	63.7	50.7
	$N2$	52.5	65.0	32.9	60.4	17.5	0	78.2	57.2
	$N3$	71.7	13.2	45.3	79.8	63.7	78.2	0	71.0
	$N4$	50.7	63.8	56.3	56.8	50.7	57.2	71.0	0

The figure shows the distance between the first four diabetic (D1–D4) and normal (N1–N4) women. What are the number of misclassified observations for leave-one-out cross validation based on 3-nearest neighbor classification when only considering the 8 observations (i.e., D1–D4 and N1–N4) in the figure?

A. None of the observations will be misclassified.

B. 2 of the observations will be misclassified.

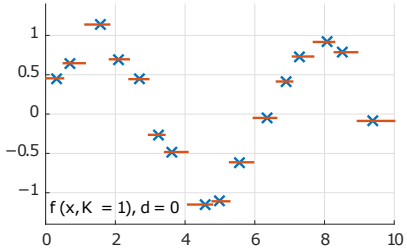
C. 6 of the observations will be misclassified.

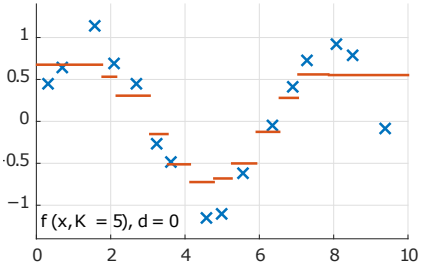
D. All of the observations will be misclassified.

KNN Regression

- Given a training set $\boldsymbol{X}, \boldsymbol{y}$
- For a test observation x predict the average y-value in the neighbourhood

$$\hat{y} = f(\boldsymbol{x}, K) = \frac{1}{K} \sum_{i \in N_{\boldsymbol{X}}(\boldsymbol{x}, K)} y_i$$





Summary

Summary

- Implications of model complexity (overfitting)
- Generalization error
  - Practical ways to estimate it (cross-validation)
  - Problem of cross-validation and how to fix it (2-layer CV)
- Feature selection
  - Forward
  - Backward
- Nearest neighbor methods
  - Classification
  - Regression
  - How to estimate K

Resources

https://towardsdatascience.com Alternative introduction to cross-validation

(https://medium.com/towards-data-science/

cross-validation-explained-evaluating-estimator-performance-e51e5430ff85)