# Applied Machine Learning in Health Sciences 2023
-
# Introduction, linear regression, model evaluation & resampling

Peter Mondrup Rasmussen

CFIN

pmr@cfin.au.dk

# Course outline

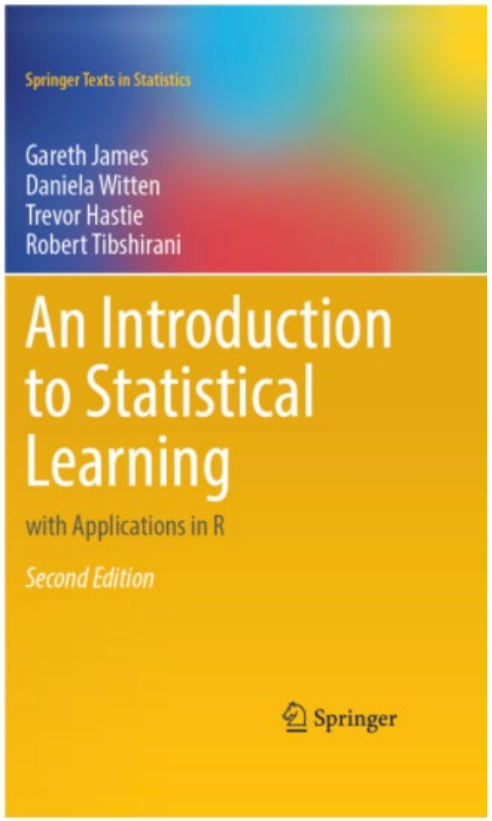
- **Five course days**
  - 9th Jan 2023: Introduction. Supervised learning, linear regression.
  - 11th Jan 2023: Supervised learning, logistic regression, regularization support vector machines.
  - 13th Jan 2023: Supervised learning, neural networks, group work.
  - 16th Jan 2023: Unsupervised learning, dimension reduction techniques, clustering techniques.
  - 18th Jan 2023: Supervised learning, group work. Q&A session, start working on machine learning project.
- **Before each course day**
  - Prepare according to the lecture plan. Lecture plan and links to optional video material available on Brightspace.
- **Course days**
  - Lectures, exercises (conceptual, theoretical, computer), discussion.
- **After each course day**
  - Finish exercises.
- **Course evaluation – exam assignment consists of two parts**
  - Part I: Assignment/exercise portfolio based on in-class exercise work. Aim to complete this part during the course days/after the individual course days.
  - Part II: Report disseminating a small machine learning project.
  - **Deadline:** Hand in part I and part II (single pdf document with two parts) at Brightspace before **3rd February 2023 23:59**.

# Course outline

- **Course evaluation**
  - **Part I**
    - Assignment/exercise portfolio based on in-class exercise work. Aim to complete this part during the course days (and after the individual course days).
    - Prepare answers to all exercise checkpoints. Include figures, results, descriptions, code snippets, etc.. Create/write the document as you work on the exercises throughout the course days.
    - Short concise answers, explain with your own words and demonstrate your understanding of the topics.
    - Remember to backup your document.
  - **Part II**
    - Report disseminating a small machine learning project.
    - Work on your own data or data provided in the course.
    - Formulate research question(s).
    - Apply machine learning techniques covered in the course.
      **Both a supervised- and an unsupervised technique should be used.**
    - Report format: IMRAD (introduction, methods, results and discussion).
    - Concise report. Introduce your small ML project and project research question, describe the data and machine learning methods, present results (incl. tables, figures), discuss results using the field's terminology, include code in appendix.
  - **Format to hand in (upload on Brightspace)**
    - Single pdf document with two parts (part I and part II). Name + contact information on first page. File naming: **`participantName_exam_aml2023.pdf`**

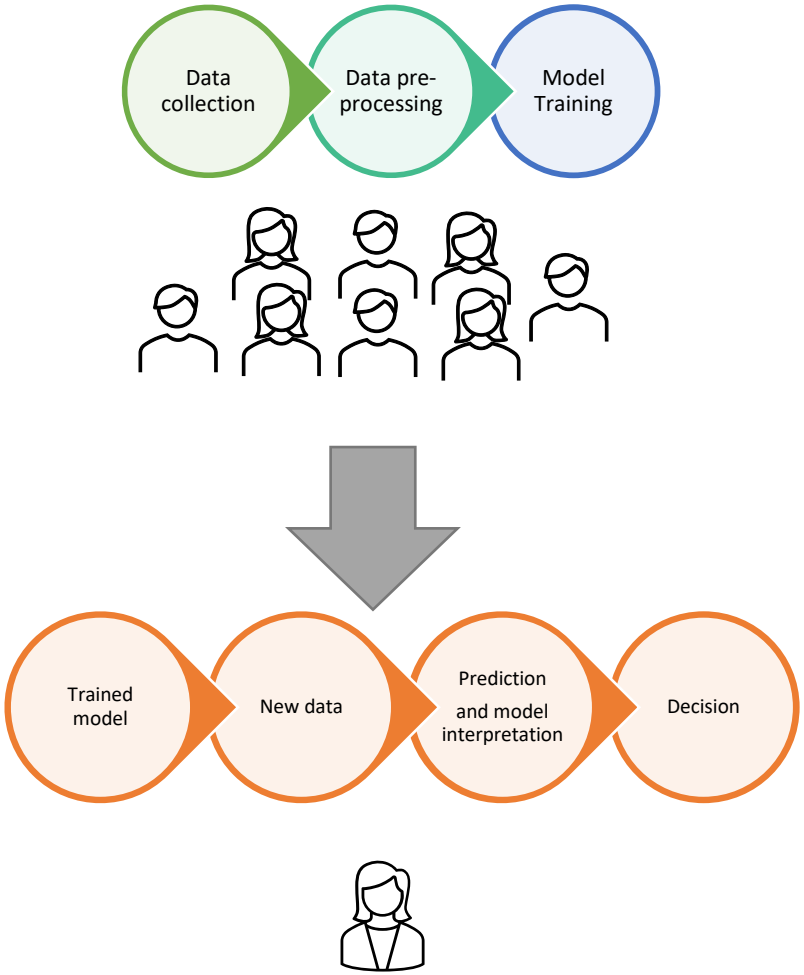
# Course outline

- Text book www.statlearning.com




- Emphasis is not on (detailed) technical aspects.
- Become an informed user of machine learning techniques and be able to understand the models, intuitions, strengths and waknesses.
- Contribute to YOUR research field by the use of machine learning techniques.

# Introduction




- Very dynamic filed that lies at the interface between computer science and statistics.
- Naming differences: Machine learning, statistical learning, statistical machine learning, …
- Often the models and techniques are the same but the academical background and focus of authors/practioneers may be differ (computer science vs. statistics).
- But often common goals:
  - Build models from data that can be used for making predictions and decisions.
  - Discover structure and new representations from data.

# Machine learning in health sciences

- Genetics and genomics https://doi.org/10.1038/nrg3920
  - Automatic annotation of transcription sequences e.g. automatic recognize location of transcription start sites.
  - Use gene expression profiles to predict outcome e.g. disease.
  - Discover new functional elements.
- Imaging, e.g. neuroimaging
  - Functional neuroimaging, study brain function.
  - Image segmentation, e.g. tumor detection, volume measurement.
  - Disease detection, outcome prediction.
- Histology
  - Cell type classfication, in e.g. tissue or blood sample.
  - Tumor detection (image segmentation).
- Intensitve care unit
  - Predict hospitalization time at ICU, predict progression.

# Data sets

Body density data
CSF biomarker data

# Data sets in the course

- In the course we work on two *synthetic data* sets eliminating the risk of identifying individuals, i.e. GDPR ect. does not apply to these data.

- The data sets has been generated from population models of the original data.

- Technically, multivariate Gaussian distributions was fitted to the original data→ statistical properties such as feature means, between feature covariances, differences in feature means or covariances across subject groups are preserved.

- Similar number of features and number of observations as in original data sets → machine learning tasks and difficulty (e.g., signal to noise ratio) resembles what we would have observed when analyzing the original data.

# Data sets in the course – body density data

- Body Density Data
- Source: http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_BMI_Regression
- Measurement of body fat is inconvenient and costly. The data set contains estimates of the percentage of body fat $PBF$ determined by underwater weighing and various body circumference measurements for 252 men.
- Percentage of body fat for an individual can be estimated once body density $D$ has been determined Siri, W.E. (1956)

$$PBF = \frac{495}{D} - 450$$

- Can we build a model, that allows us to **predict body density from body circumference measurements**?

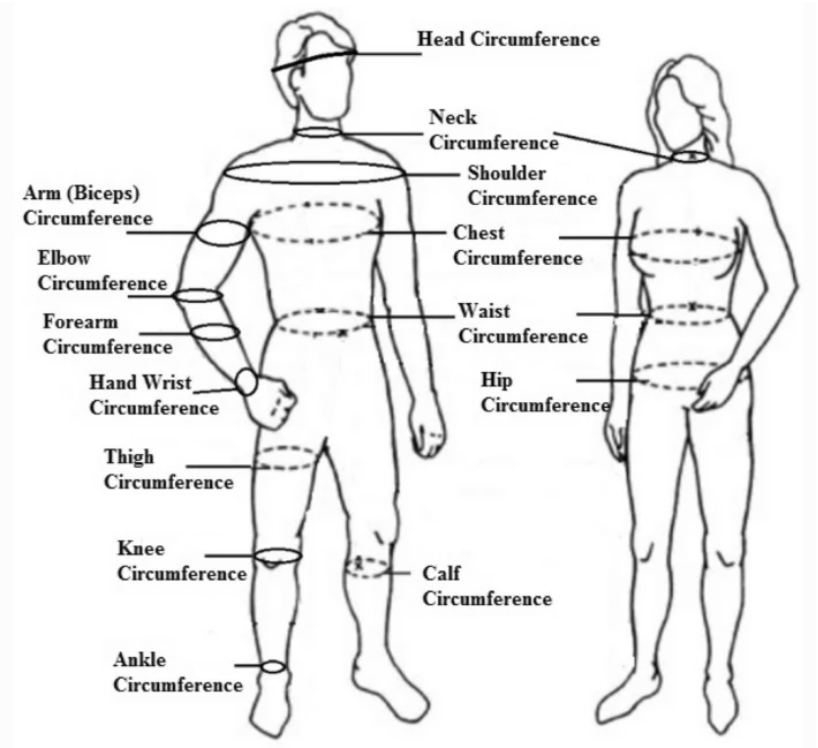
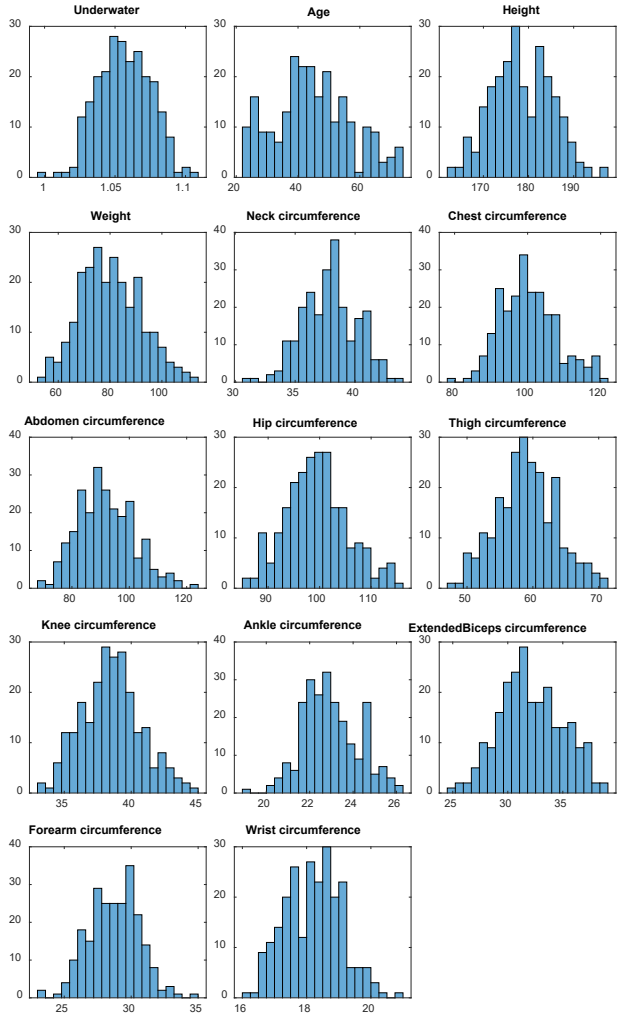



Image source: Kalkışım, Ş.N., Çan, M.A., Erden, A. *et al.* Relationships between anthropometric measurements, muscle strength and body awareness. *Acta Neurol Belg* (2021)

# Data sets in the course – body density data

- Input data $X$: $n = 252$ observations and $p = 13$ features:
  - Age Height Weight
    Neck_circumference
    Chest_circumference
    Abdomen_circumference
    Hip_circumference
    Thigh_circumference
    Knee_circumference
    Ankle_circumference
    ExtendedBiceps_circumference
    Forearm_circumference
    Wrist_circumference
- Output/response: Body density.
- Measurement units: See link at previous slide.

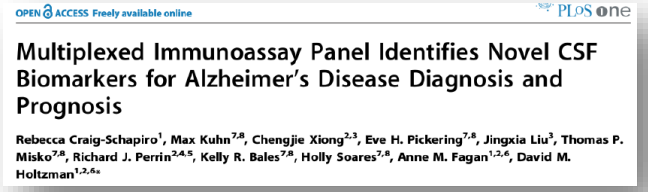


# Data sets in the course – CSF Biomarker data set

- CSF Biomarkers
- Original data available in the R package: *Applied Predictive Modeling*. https://cran.r-project.org/web/packages/AppliedPredictiveModeling (the publicly available data is a modified version of the values used for the publication)
- $n = 333$ observations with $p = 130$ features.
- Two groups: 242 controls, 91 congnitive impaired.
- We will work on 100 participants (group proportions preserved).
- Six APOE genotypes E2E2, E2E3,E2E4,E3E3,E3E4,E4E4 were combined into a single feature APOE+ indicating if at least one E4 gene is present.

# Application example

- Early detection of Alzheimer's disease based on CSF biomarkers


Model

**Predict** (at individual subject level)
*Control vs. impaired*

**Inference**
*Important features?*

# Types of Machine Learning

Supervised learning
Unsupervised learning

# Supervised learning

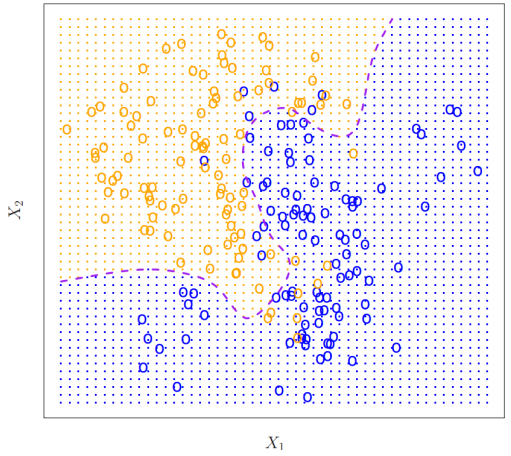
**Data:** Available is a set of $n$ inputs / predictor measurements $x_i, i = 1, \dots, n$ and for each of these there is an associated output / response measurment $y_i$.

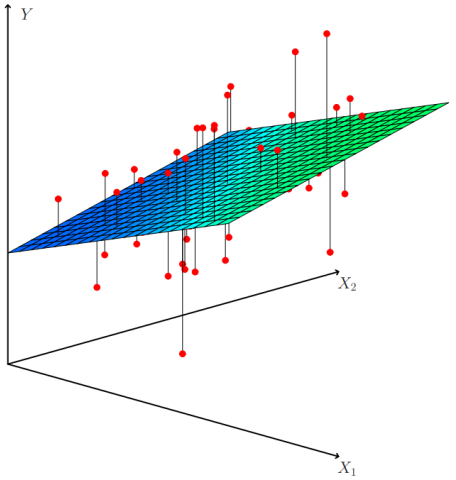
**Classification:** Responses $y_i$ are discrete class labels.

**Regression:** Repsonses $y_i$ are continuous.

**Goal:**

- **Prediction:** Build a model that correctly can predict the output corresponding to a new measurement $x^*$.
- **Inference:** Identify imortant features to the model's decisions.



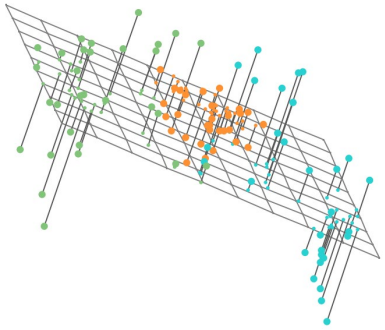



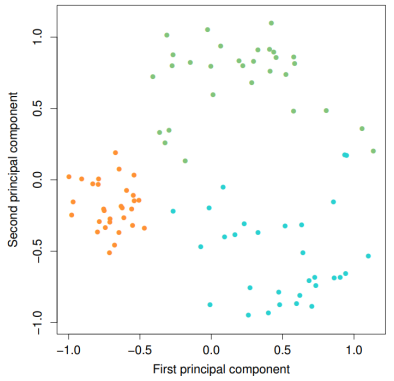
# Unsupervised learning

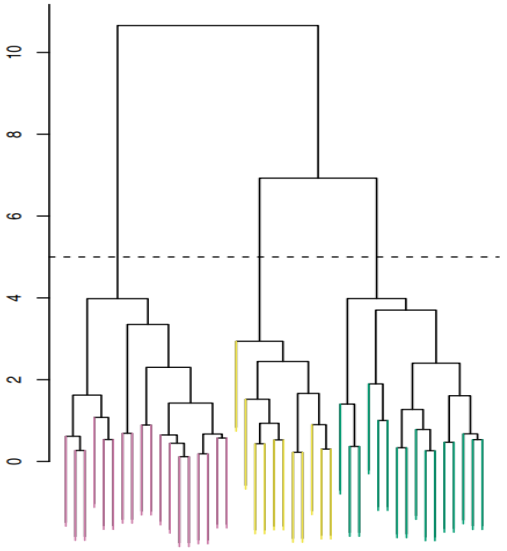
**Data:** Available is a set of $n$ inputs / predictor measurements $x_i, i = 1, \ldots, n$ but no associated output / response measurments $y_i$.

**Goal:**

- **Understand the relationship between variables or observations, discover structure, construct another representation of data:** Dimension reduction techniques, cluster analysis.





# Estimating a model

Supervised learning

# Supervised learning – estimating a model

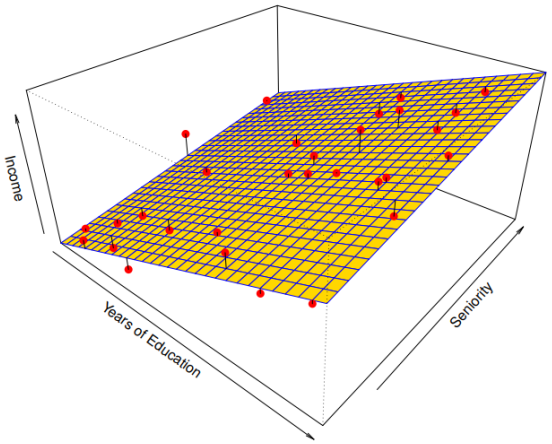
Suppose that the relationship between a *quantitative* response $Y$ and a predictor variable $X = (X_1, X_2, \ldots, X_p)$ is

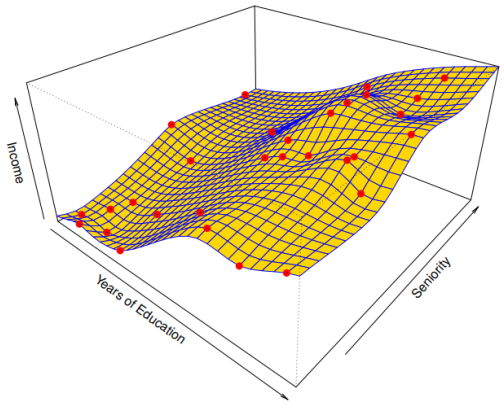
$$Y = f(X) + \epsilon,$$

where $f(X)$ is an unknown function of the $p$ predictor variables and $\varepsilon$ is a random error term, that is independent of $X$.

The function $f(X)$ is unknown, and our goal is find an estimate $\hat{f}(X)$ of this function so that we can predict $Y$ using

$$\hat{Y} = \hat{f}(X).$$



# Supervised learning – estimating a model

- To quantify the accuracy of $\hat{Y}$ as a predictor for $Y$ it is common (in regression) to use the squared difference between observations and model predictions

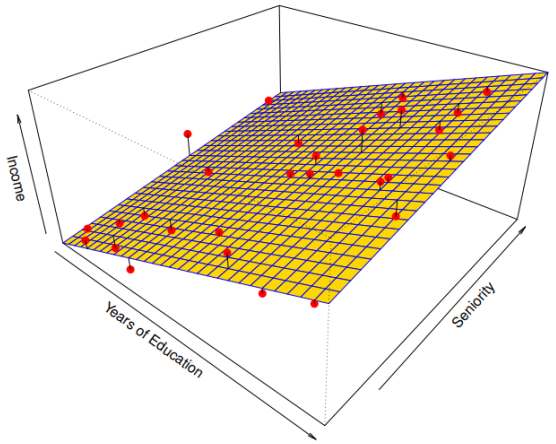
$$\left(Y - \hat{Y}\right)^2.$$

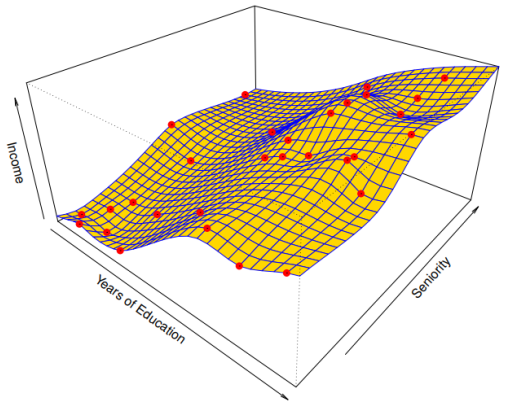
- The average or expected value of this squared difference can be decomposed in to two terms

$$E\left(Y - \hat{Y}\right)^2 =$$

$$\left[f(X) - \hat{f}(X)\right]^2 + Var(\epsilon).$$

- The first term represents a reducible error and can be reduced by building a more accurate mode. The last term represents the variance associated with the random error and cannot be reduced regardless how good our model is.



# Supervised learning – model accuracy

Now, suppose that the underlying model is unknown to us. Instead we have a data set with $n_{train}$ observations available for training/estimating a model: $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n),\}$

To quantify model accuracy we can use the *mean squared error*

$$\boldsymbol{MSE_{train}} = \frac{1}{n_{train}} \sum_{i=1}^{n_{train}} \left(y_i - \hat{y}(x_i)\right)^2.$$

However, instead of knowing how well our model performs on the training data we are often more interested in how our model will performs on a new/future data sample $x_0$, that is, how close is $\hat{y}(x_0)$ to $y_0$, where $(x_0, y_0)$ is test data that was not used to train the model?

What we are interested in is then the generalization error

$$\left(y_0 - \hat{y}(x_0)\right)^2,$$

where the average squared test error is computed over all possible values of $(x_0, y_0)$. Usually, the generalization error is estimated form $n_{test}$ test observations

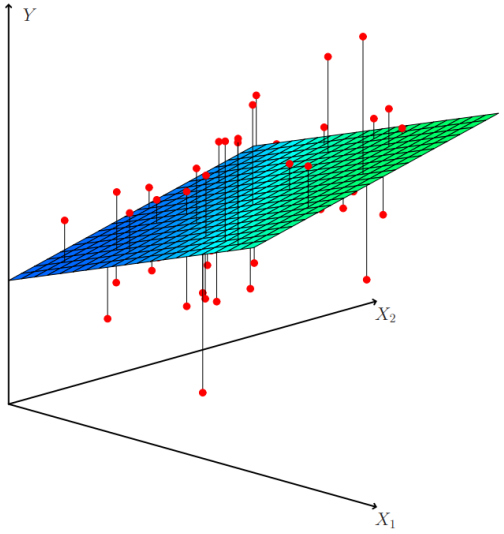
$$\boldsymbol{MSE_{test}} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \left(y_i - \hat{y}(x_i)\right)^2.$$

# Supervised learning – model accuracy

**Regression**

The mean squared error is often used as an error metrics

$$Err = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}(x_i)\right)^2.$$

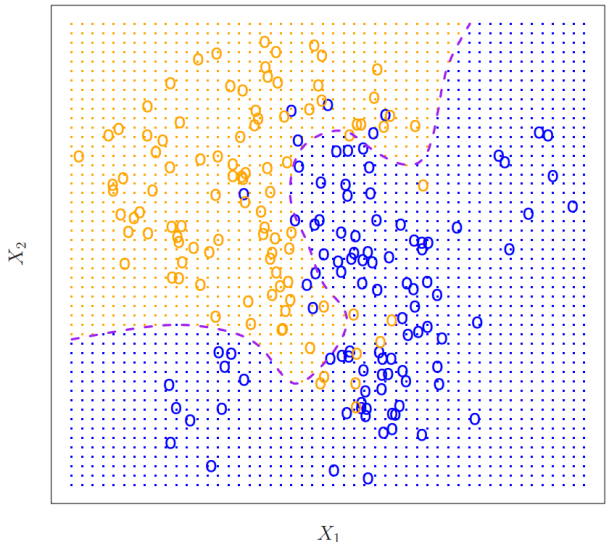


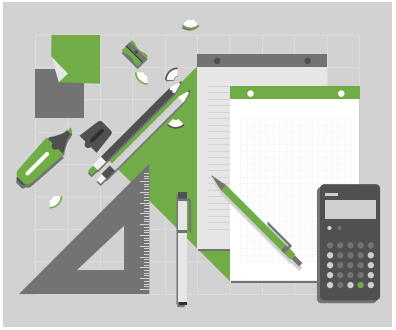

**Classification**

The error rate is often used as an error metrics

$$Err = \frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i),$$

where $I(y_i \neq \hat{y}_i)$ is an indicator variable that equals 1 if $y_i \neq \hat{y}_i$ and zero if $y_i = \hat{y}_i$.





# In-class exercises

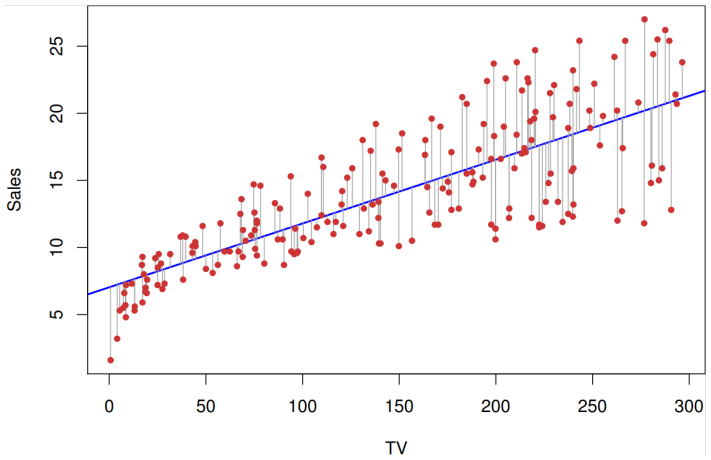
1 Introduction

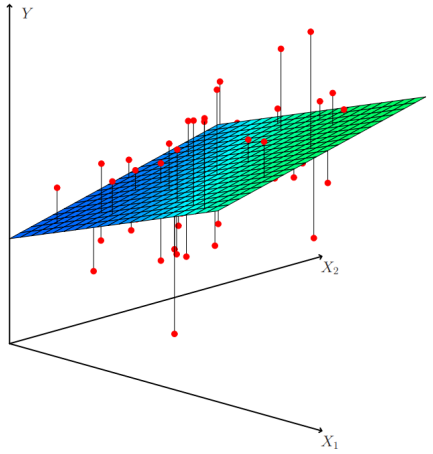
# Linear regression

# Linear regression

- Linear regression is a simple but very useful approach in supervised learning for predicting a *quantitative response*.
- ISL distinguishes between *simple*- and *multiple* linear regression, but here we will cover both these at the same time, and use the simplifying term *linear regreression.*
- The linear regression model takes the form

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$



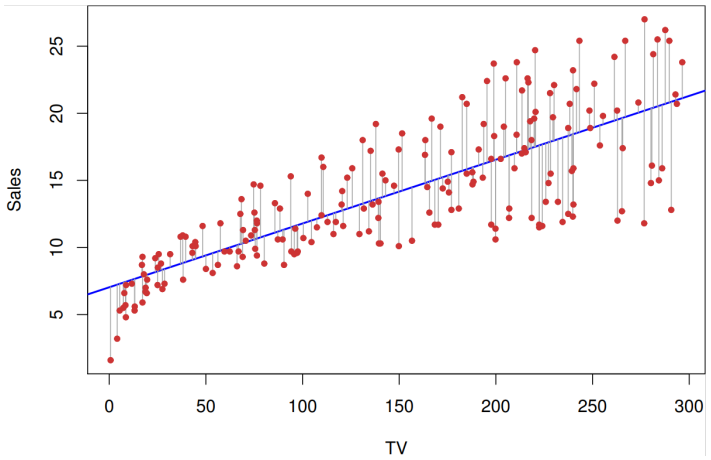



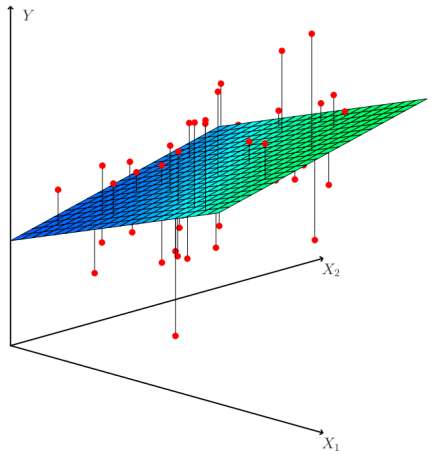
# Linear regression

- Model parameters $\beta$ are unknown and must be estimated.
- Given estimates $\hat{\beta}$ we can make predictions by

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$

- Model parameters are usually estimated by minimizing the sum of squared residuals $RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$.



# Linear regression

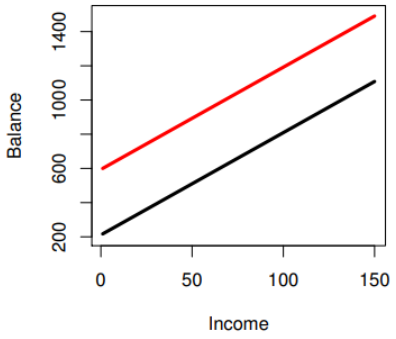
- Flexibility of the linear regression model can easily be extended, e.g.:
  - Including interaction terms

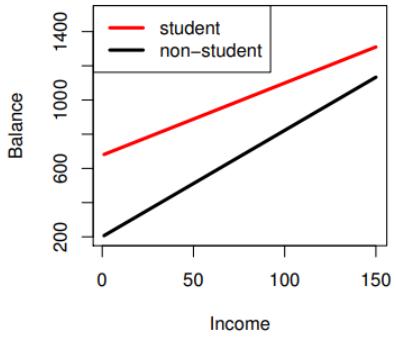
$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \epsilon$$

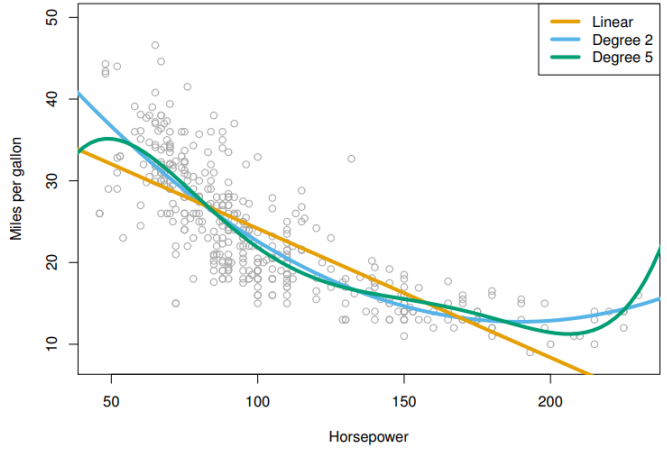
  - Polynomial regression / polynomial expansion

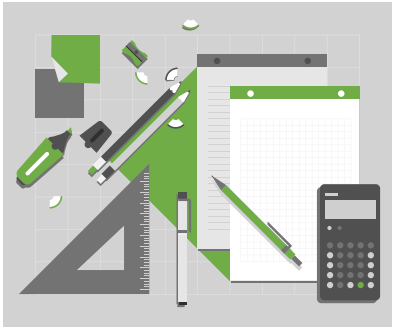
$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_1^3 + \epsilon$$

- Note that we here effectively are computing a set of new *features* which, in turn, are used in the linear regression model.



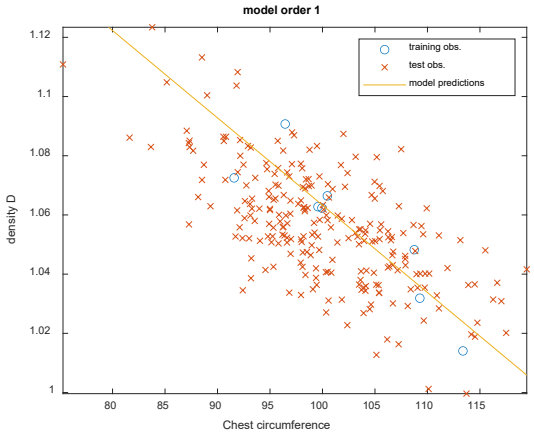


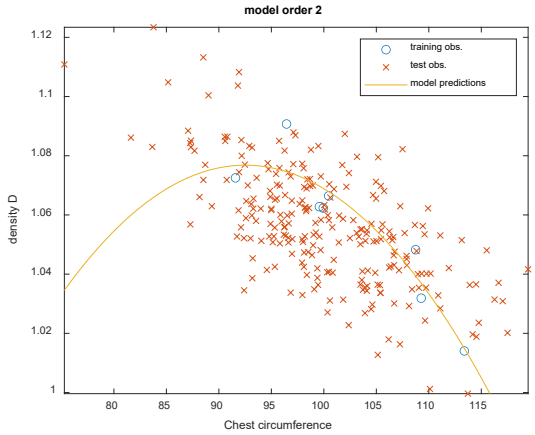
# In-class exercises

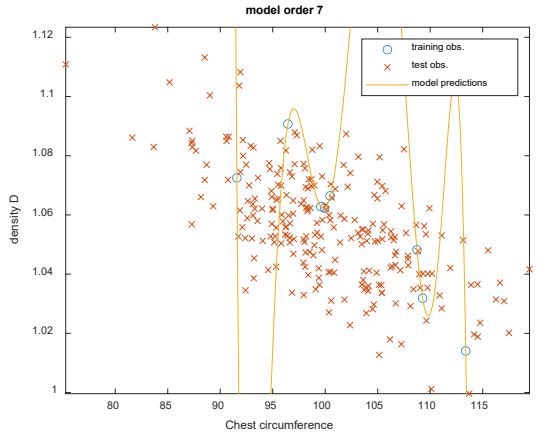
2 – Linear regression

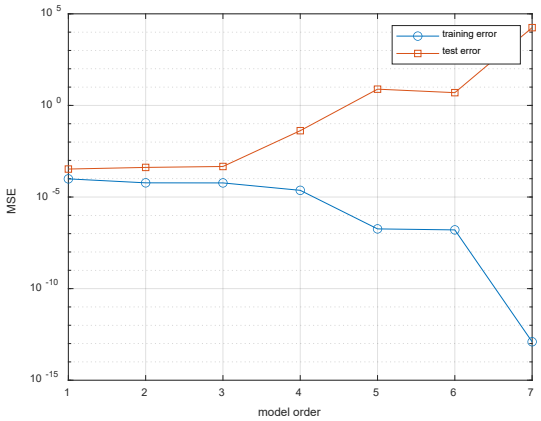
# Bias-Variance Trade-Off

# Linear regression – body density data



# Bias-Variance Decomposition

Assuming that data is generated by

$$Y = f(X) + \epsilon,$$

and that we use the squared difference to quantify the error between observed data $y_0$ and a predicted output $\hat{f}(x_0; D)$. One can decompose the expected test error at $x_0$ into three terms

$$E\left(y_0 - \hat{f}(x_0; D)\right)^2 = Var\left(\hat{f}(x_0; D)\right) + \left(Bias\left[\hat{f}(x_0; D)\right]\right)^2 + Var(\epsilon)$$

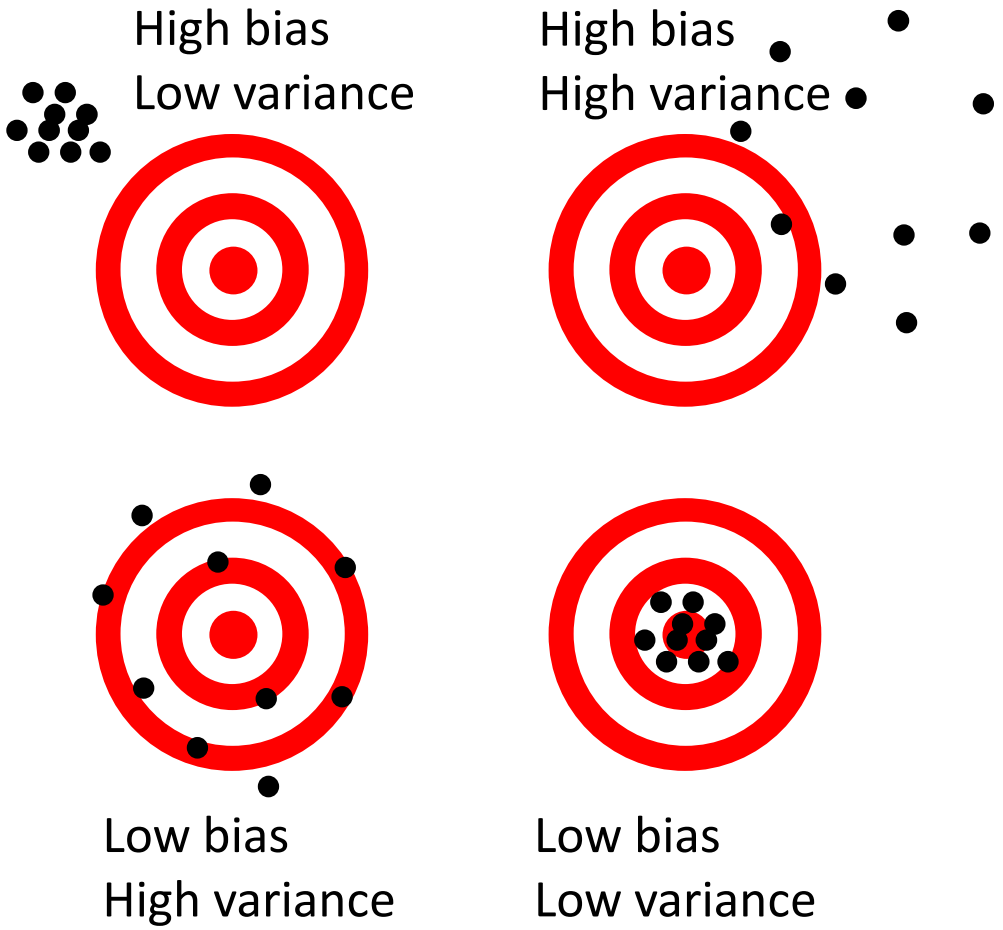
where $D$ denotes that parameters in the model has been learned from a training set. The expectation is computed over a large number of *different* training sets.

The **variance term** represents how much the estimated model will change if we estimated it on different training sets.

The **squared bias** represents how much our estimated model differs, on average, from the true model.

The **error term** represents an *irreducible error*, and corresponds to the noise variance.

# Bias-Variance Decomposition

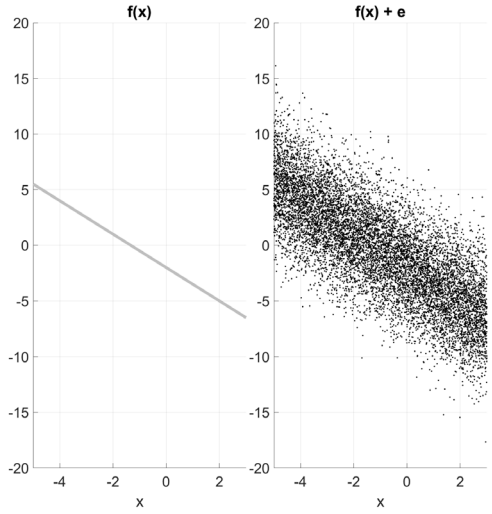


# Bias-Variance Decomposion/Trade-Off - example

Suppose that we *simulate* data according to the model

$$Y = -2 \ - 1.5X_1 + \epsilon,$$

where the random errors are Gaussian iid. with a variance equal to 9. We know that the true function is $f(x) = -2 - 1.5x_1$.



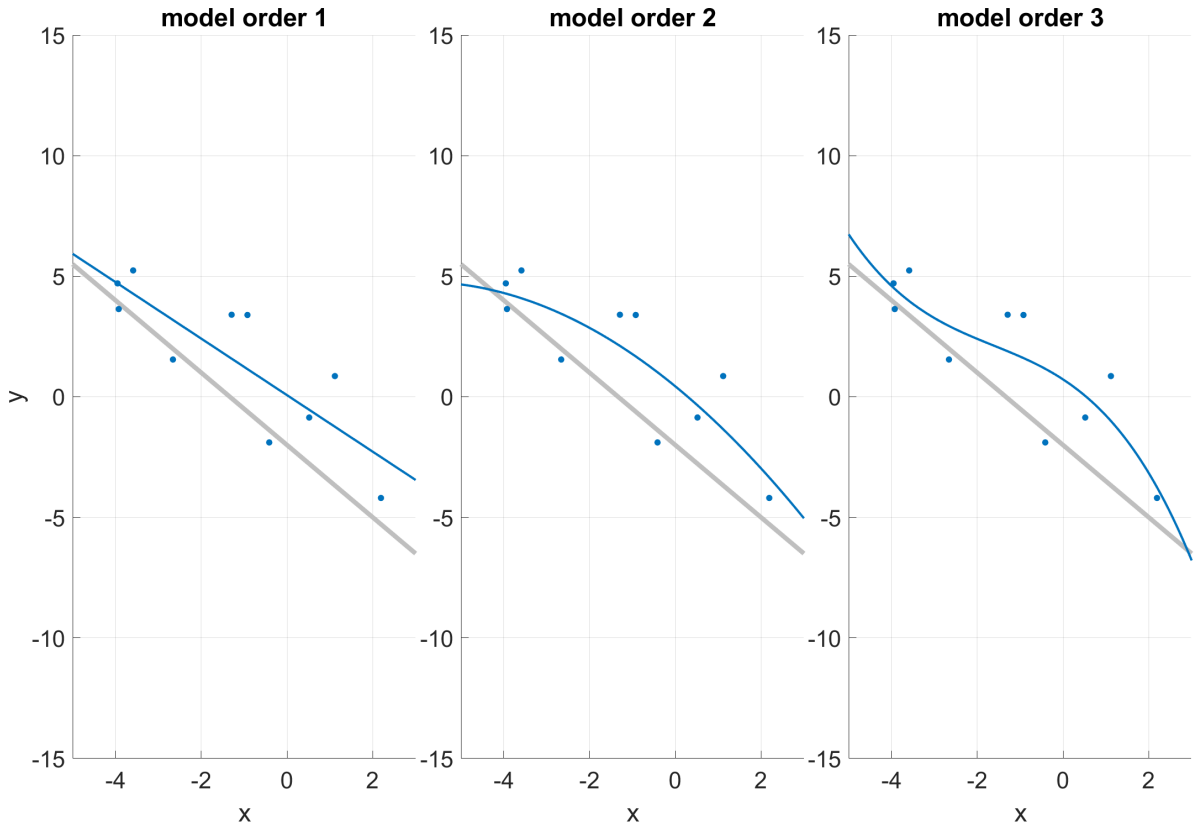
# Bias-Variance Decomposion/Trade-Off - example

- Now, suppose that we have three different models under consideration:

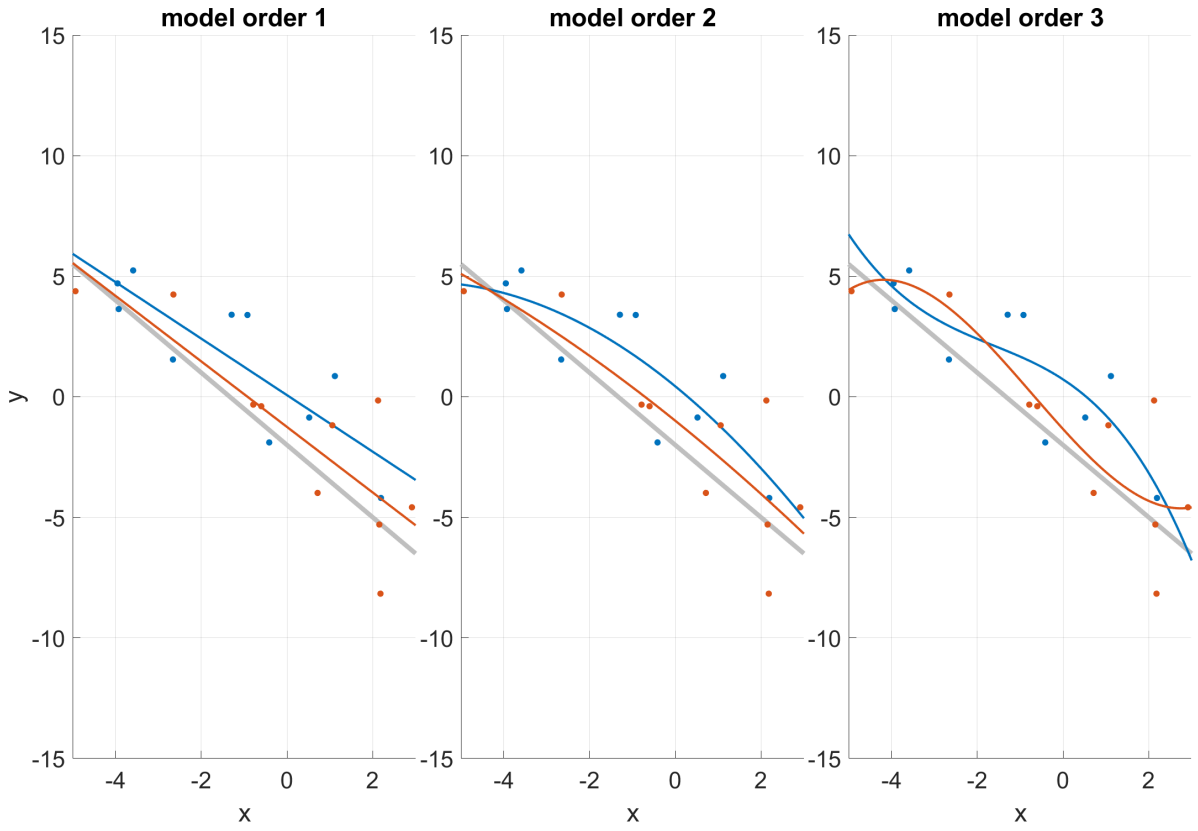
$$\text{Model order 1:} \quad Y = \beta_0 + \beta_1 X_1 + \epsilon,$$
$$\text{Model order 2:} \quad Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \epsilon,$$
$$\text{Model order 3:} \quad Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_1^3 + \epsilon$$

- Polynomial regression. Increased model complexity/flexibility with increased model order.
- Given a training data set of finite size, we will use this training data to obtain fits $\hat{\beta}$ of the model coefficients and then assess how well our model generalize to unseen data, i.e. we will estimate the generalization error/expected test error.

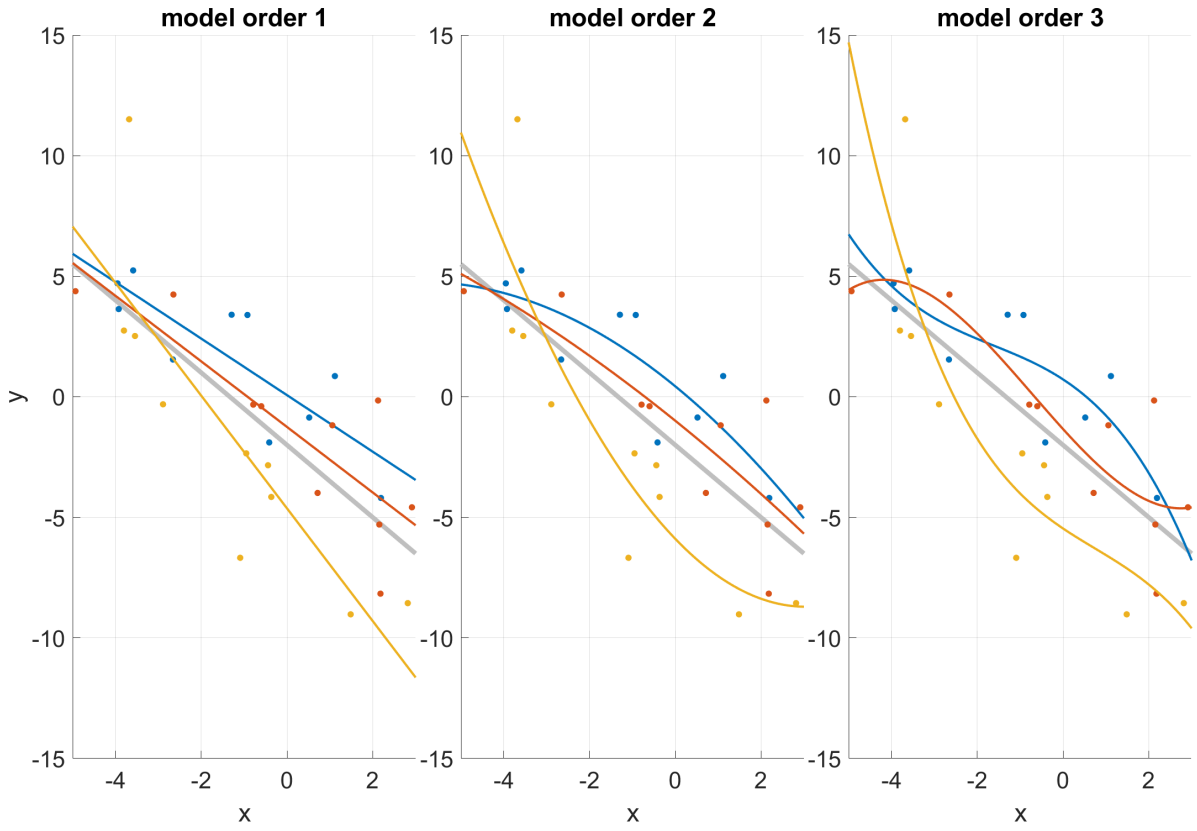
# Bias-Variance Decomposion/Trade-Off - example



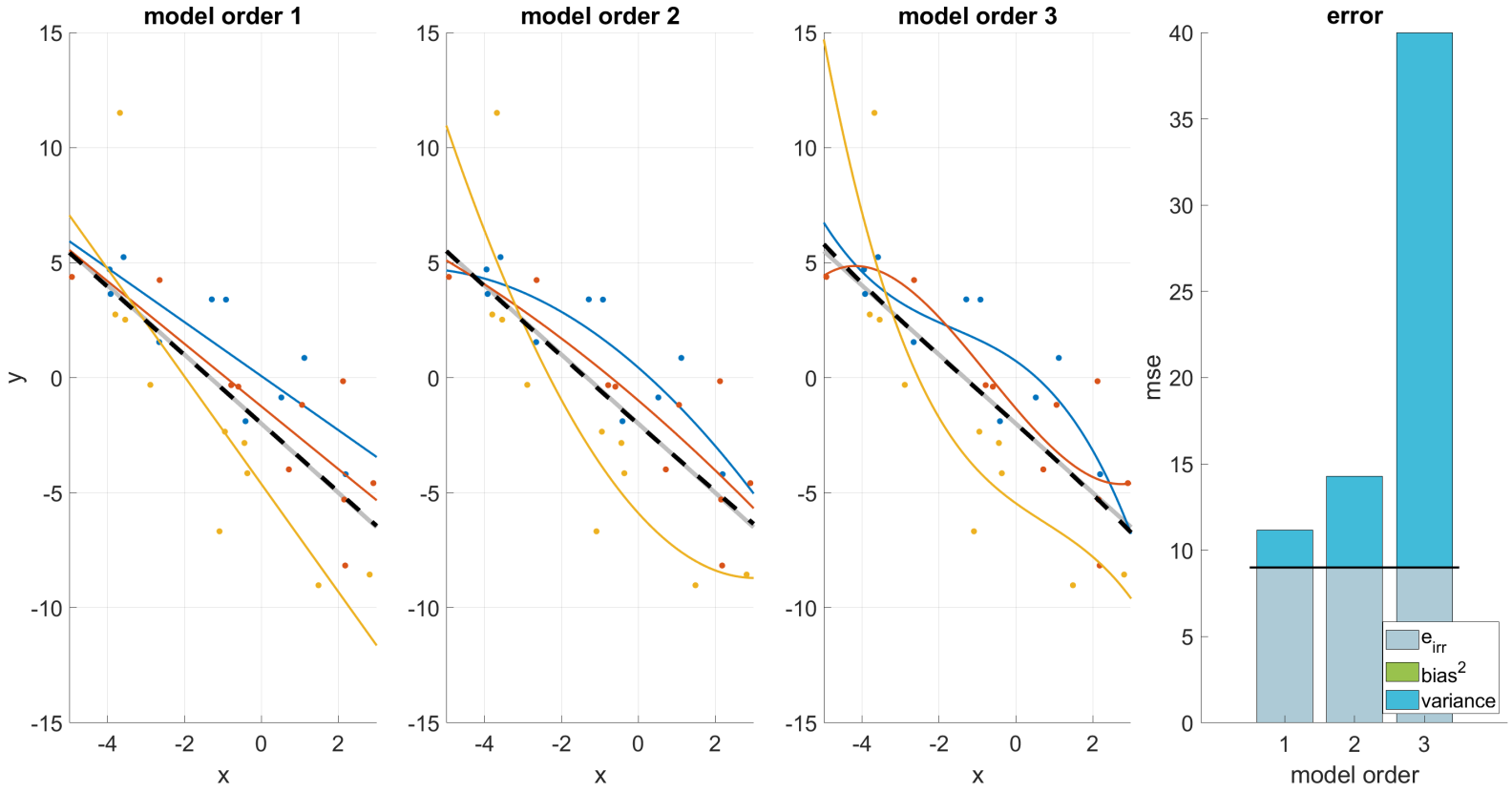
# Bias-Variance Decomposion/Trade-Off - example



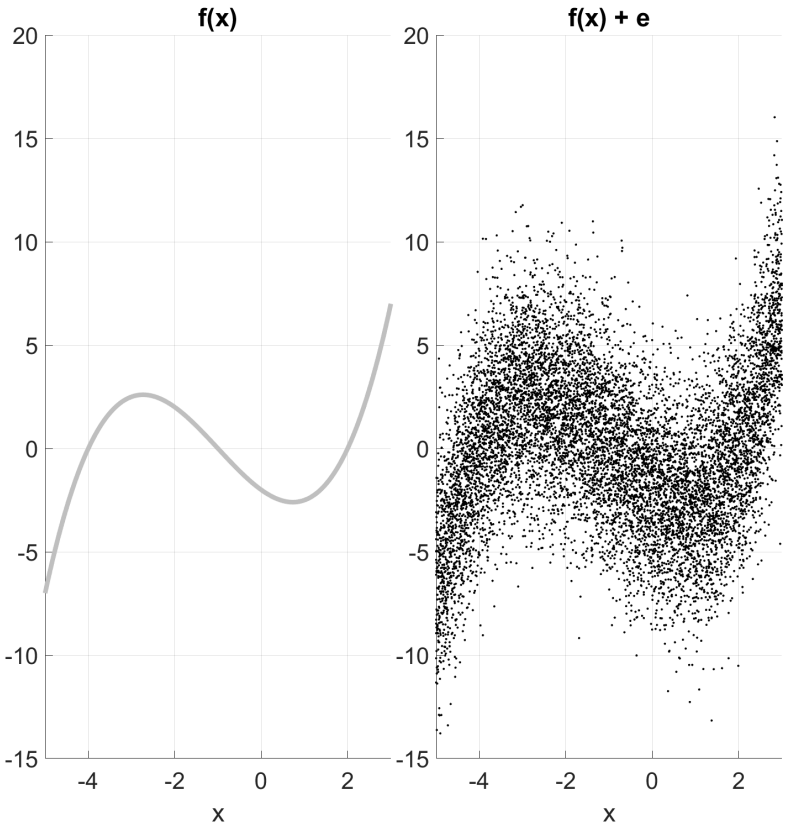
# Bias-Variance Decomposion/Trade-Off - example



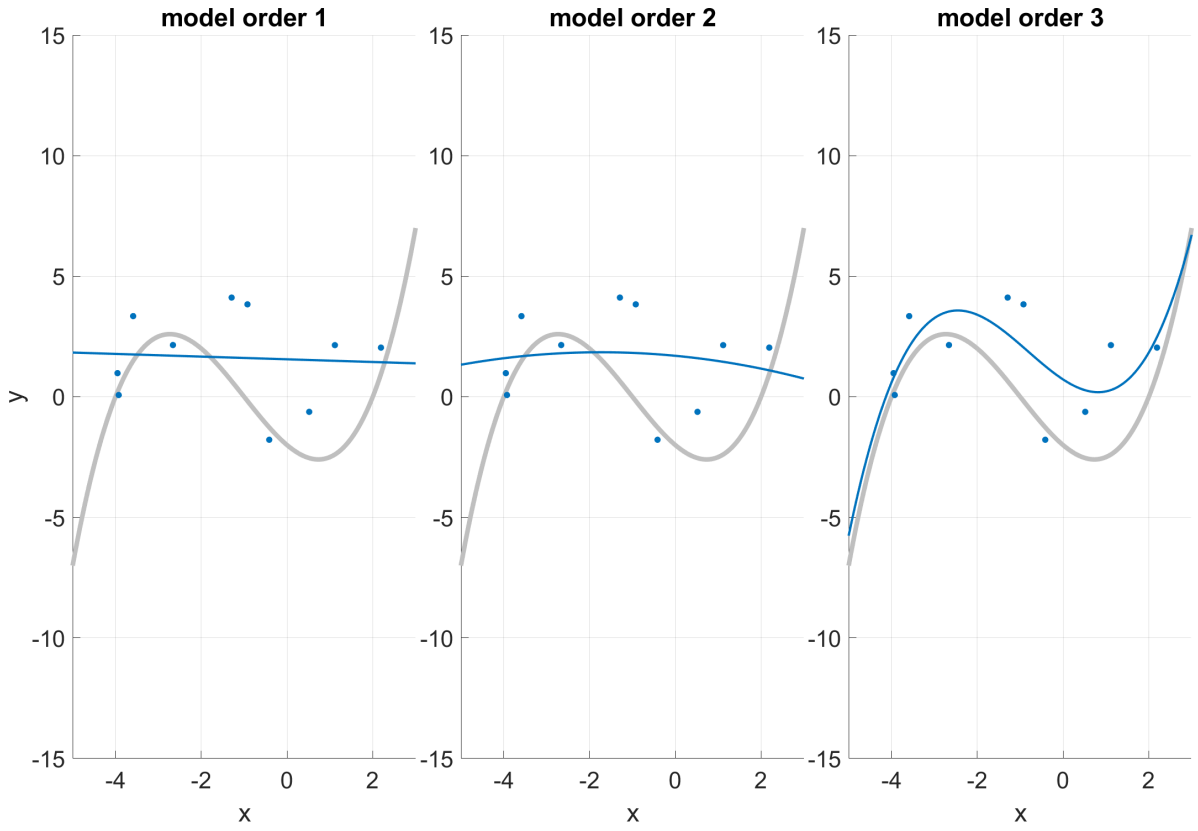
# Bias-Variance Decomposion/Trade-Off - example



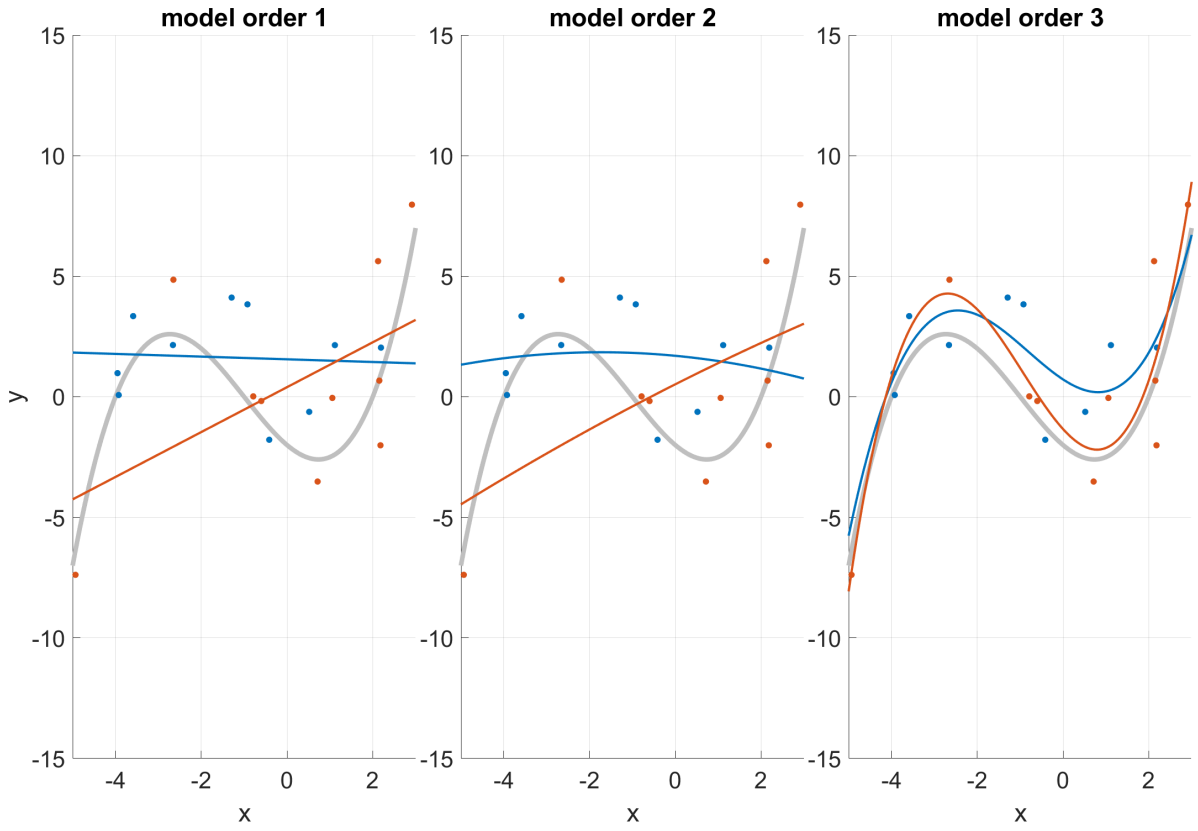
# Bias-Variance Decomposion/Trade-Off - example



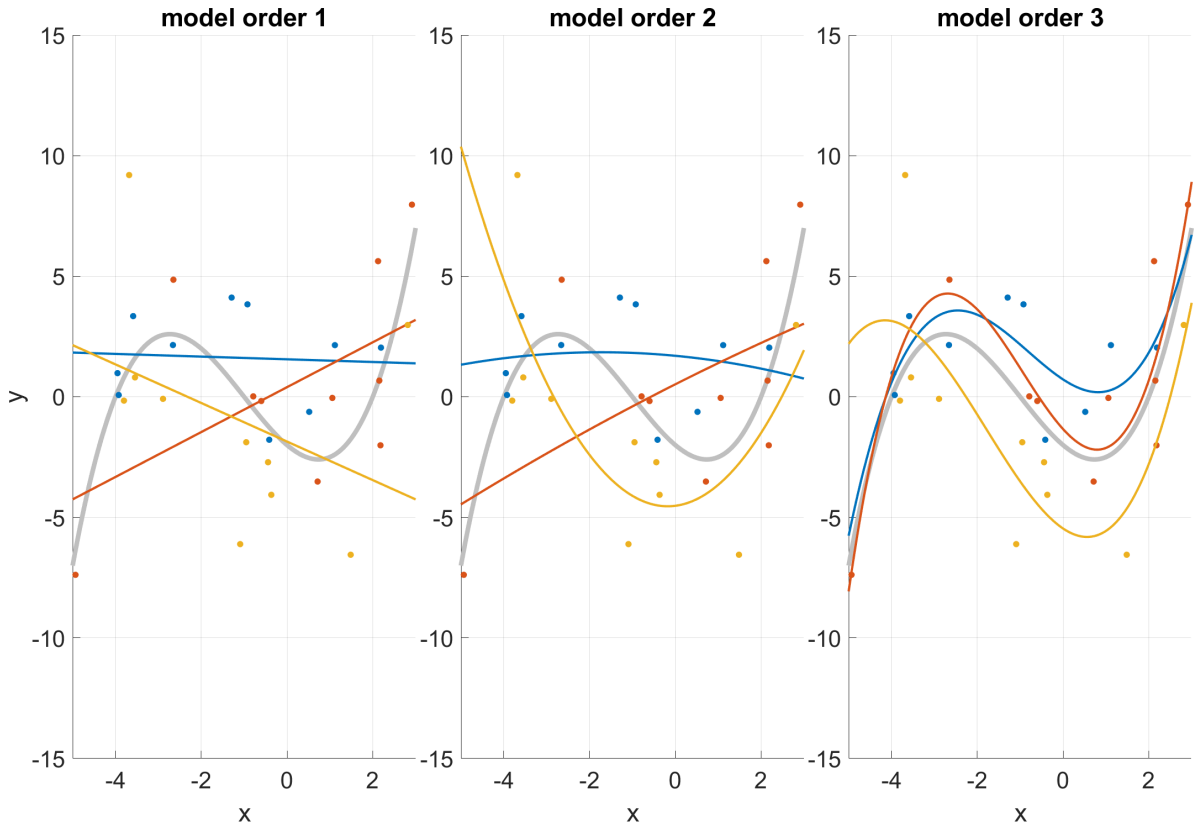
# Bias-Variance Decomposion/Trade-Off - example



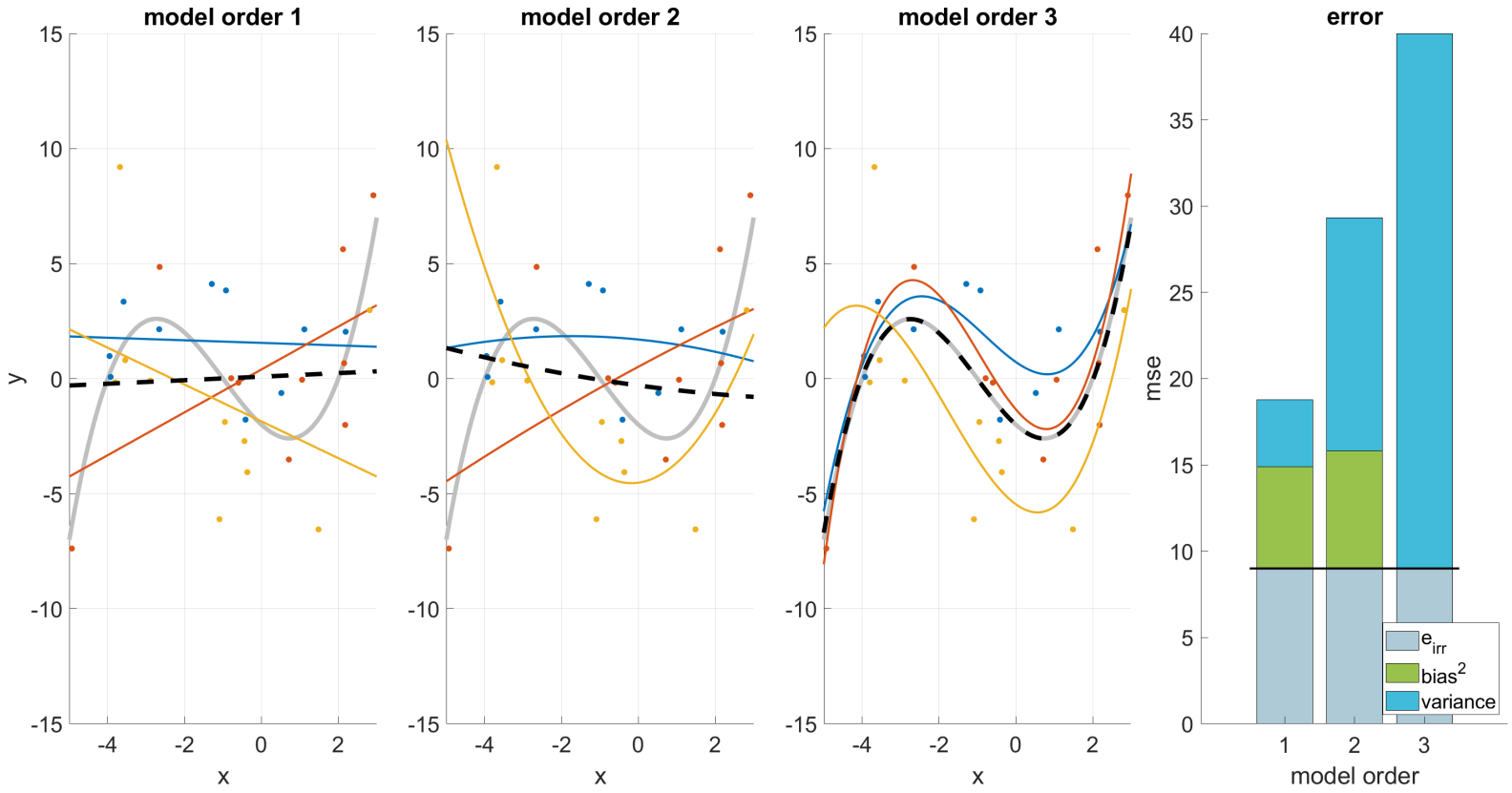
# Bias-Variance Decomposion/Trade-Off - example



# Bias-Variance Decomposion/Trade-Off - example

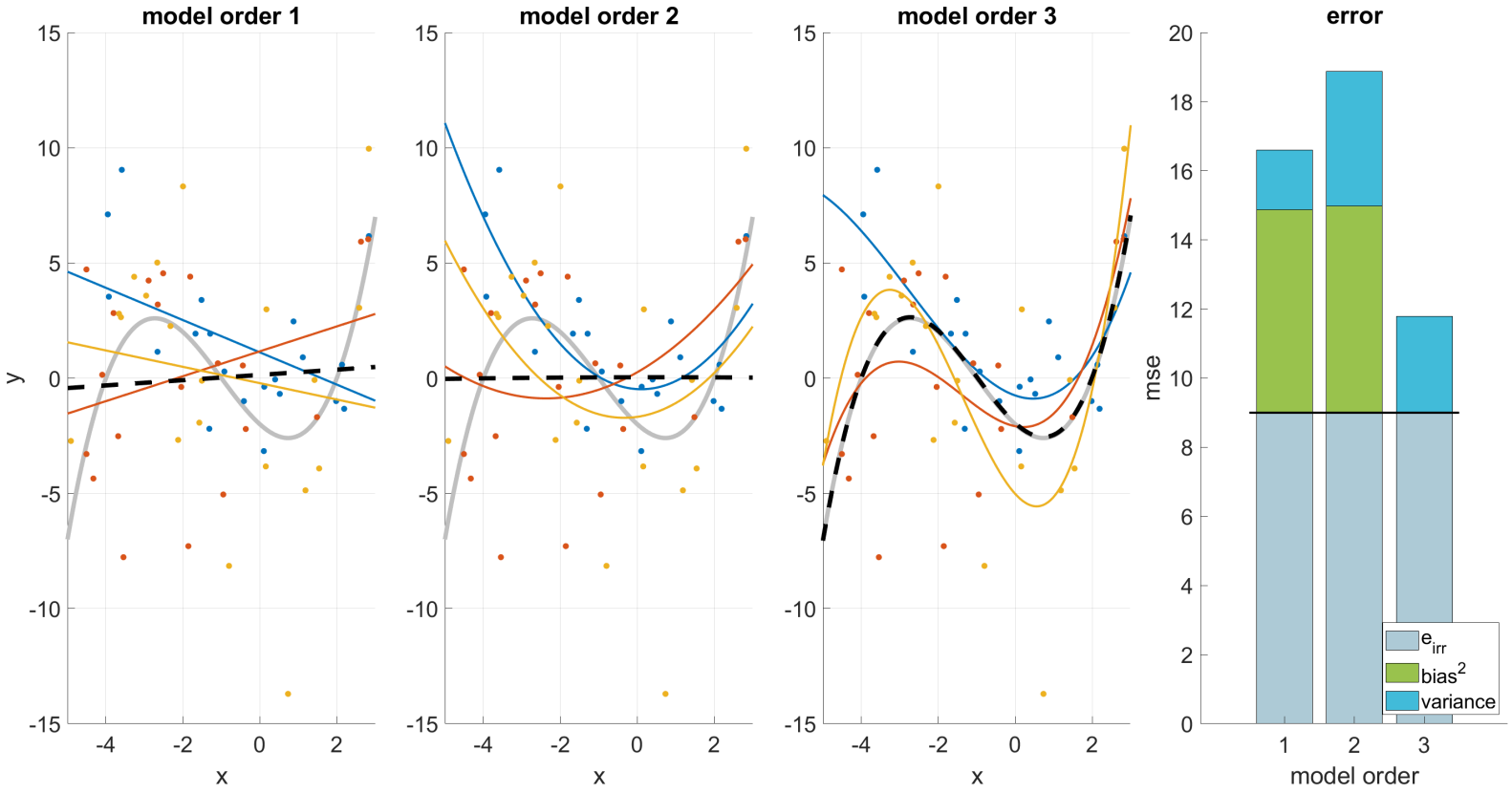


# Bias-Variance Decomposion/Trade-Off - example



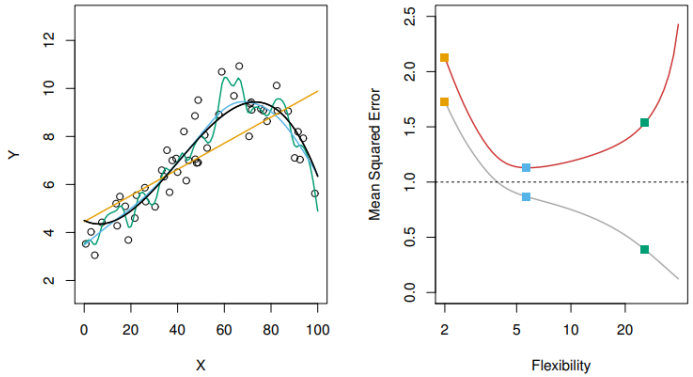
# Bias-Variance Decomposion/Trade-Off - example

Same function but now with 20 observations in the training set (instead of 10)

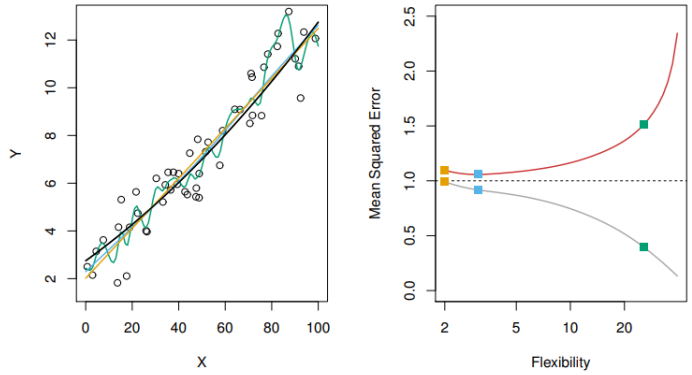


# Bias-Variance Decomposion/Trade-Off

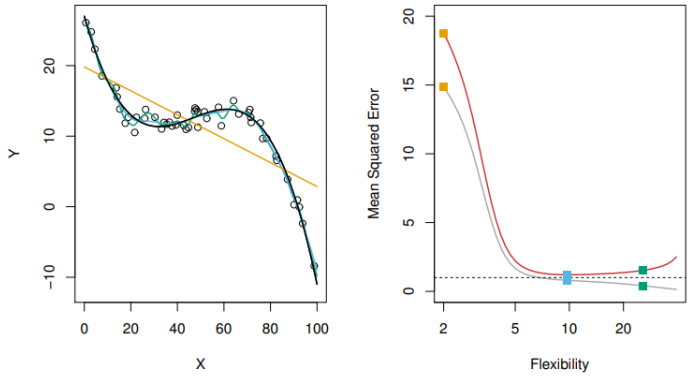
A



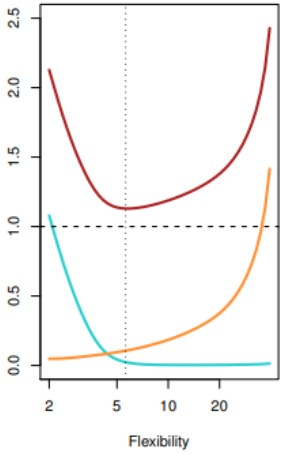

B



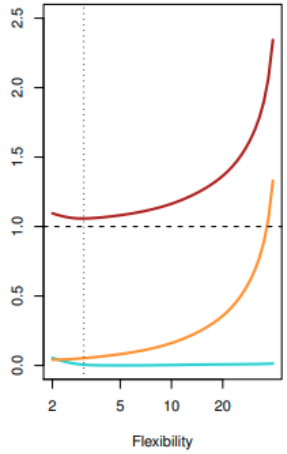

C



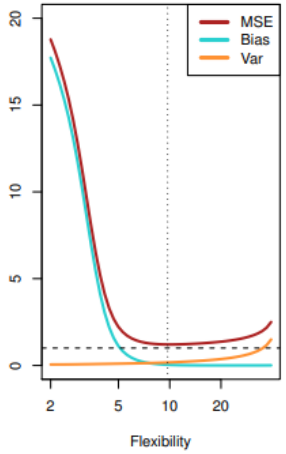

A




B



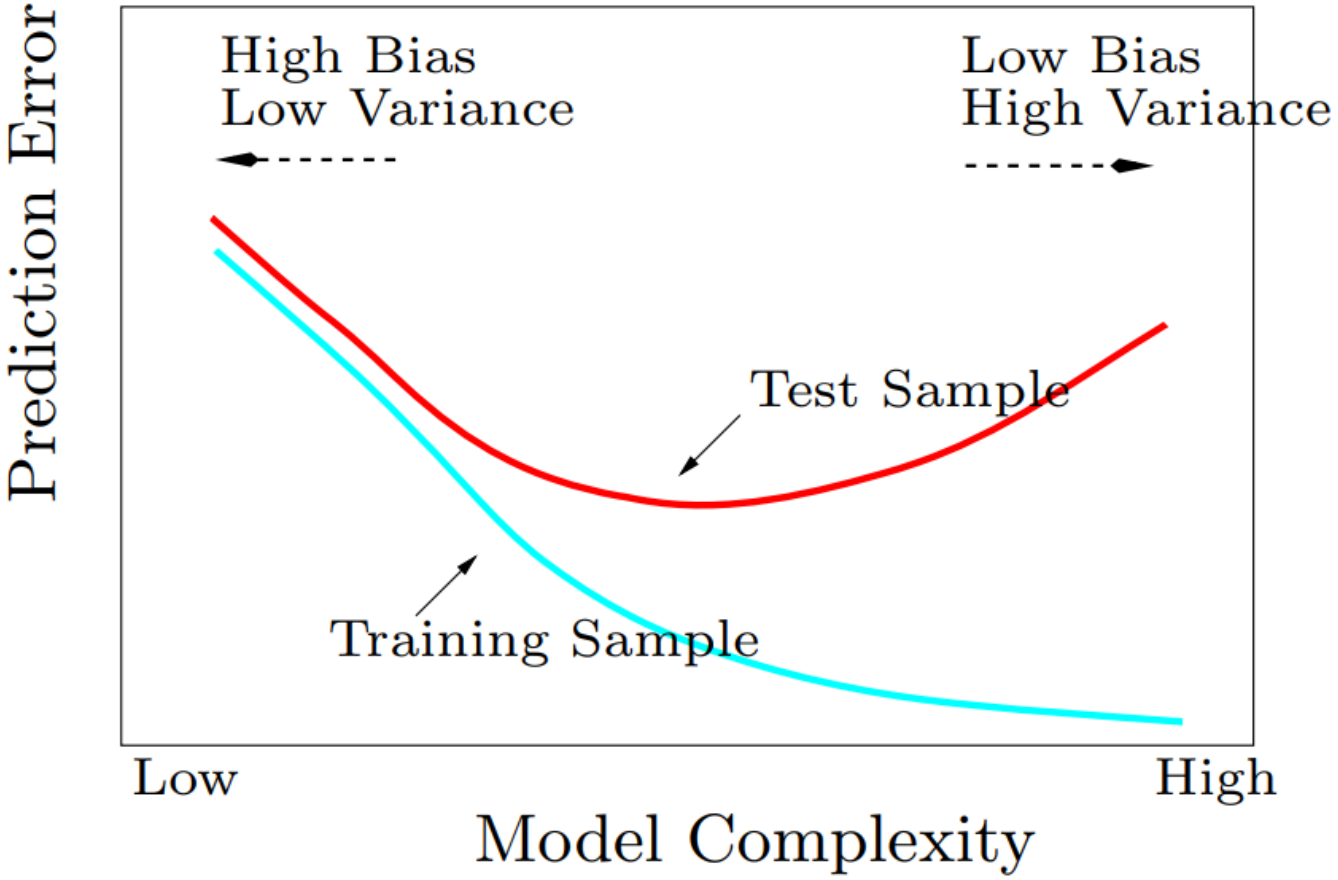

C

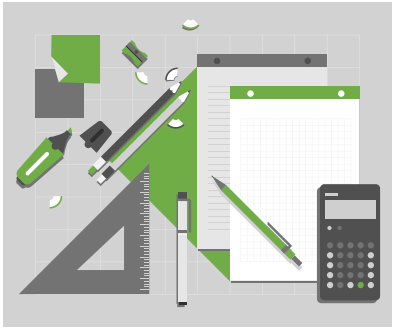


# Bias-Variance Decomposion/Trade-Off

- The Bias-Variance Decomposition/Trade-Off provides a **conceptual framework** for understanding/analyzing a **model's generalization error**.
- In **simulated data** and in **theoretical analyses**, we can **quantitatively** perform the **Bias-Variance Decomposition**. We could, for example, study how bias and variance change with model complexity/flexibility, with model type or how bias and variance change with the size of our training data set.
- In analysis of **real-world data**, we **do not know the *true* underlying model** that created the data, and we **only** have a **limited amount of data available for model training and testing**. In this case, we cannot perform the Bias-Variance Decomposiont to quantitatively estimate bias and variance. But we can use the framework as a **conceptual basis** for a **qualitative interpretation** of e.g. how **test error** changes with model flexibility.

# Bias-Variance Trade-Off





# In-class exercises

3 Bias-variance decomposition

# Model evaluation and resampling

Model Selection
Model Evaluation
Cross-Validation

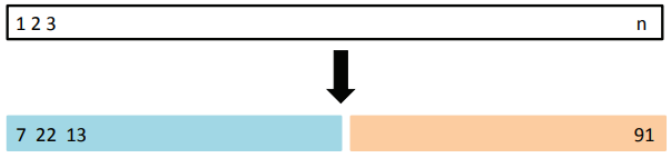
# Model selection and model evaluation

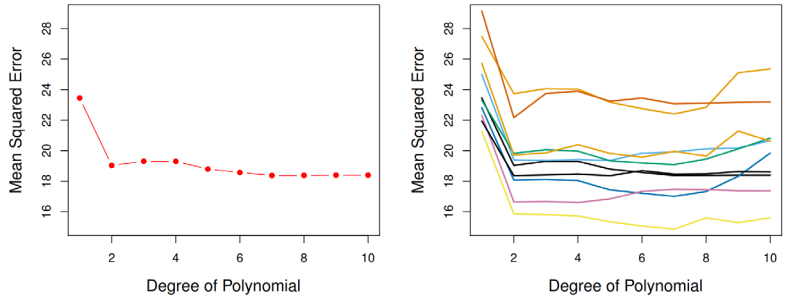
- When building models major objectives are *model selection* and *model evaluation*.
  - When building machine learning models there are no universal model that always will perform well on all types of modeling tasks and available data sets. *Model Selection* is the process of selecting a suitable model type and model flexibility for a particular data set at hand.
  - *Model evaluation/assessment* is the process of evaluating the prefomance of a model. This is typically done by evaluating the model's generalization performance by estimating the model's test error.
- In resampling one repeatedly draw samples from a data set to obtain additional information about a fitted model.
- Cross-validation is a popular resampling method used in model evaluation and model selection.

# Validation set (test set) approach

1. Split the data set with $n$ observations into two parts (perhaps with comparable sizes) – a training set and a validation set (a.k.a. a test set).
2. The model is fit on the training set and the fitted model is then applied to samples in the validation set. The error between the predicted responses on the validation set and the observed responses is then used as an estimate of the test error.

- May be highly dependent on particular observations in the two partitions.
- Often most useful if plenty of data is available.



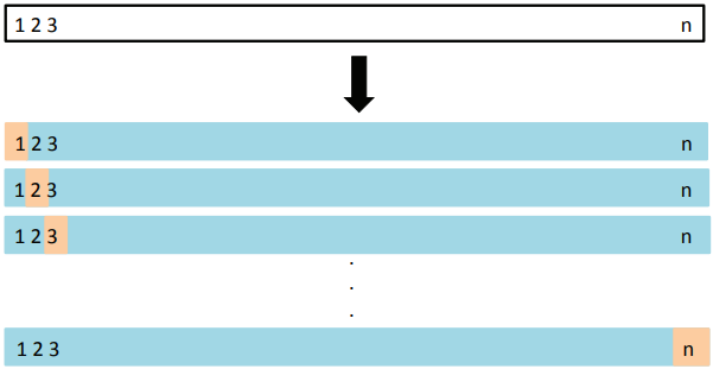



# Leave-One-Out Cross-Validation

1. Split the data set with $n$ observations into two parts a training set with $n-1$ observations and a validation set with 1 observation.
2. The model is fit on the training set and the fitted model is then applied the observation in the validation set. The error between the predicted response and the observed response is then used as an estimate of the test error $Err_i$.
3. The process 1-2 is repeated $n$ times, each time holding out a new observation for testing. The test error is then estimated by averaging across the individual estimates

$$CV_n = \frac{1}{n}\sum_{i=1}^{n} Err_i$$

- Provides almost an unbiased estiamte of the test error but may be computationally expensive for large data sets or large models.

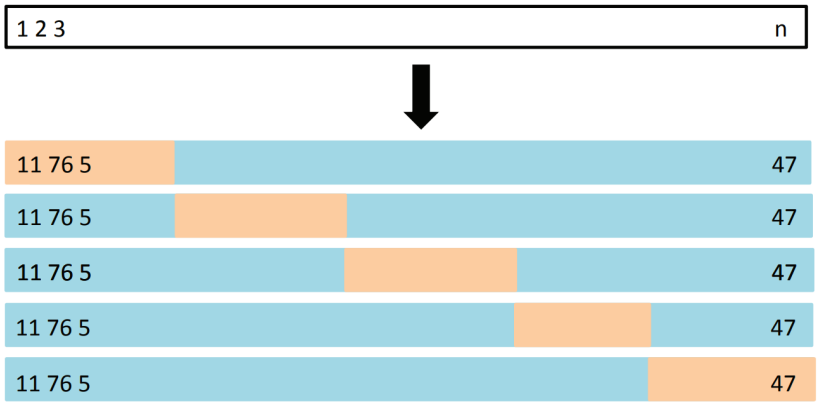


# K-fold Cross-Validation

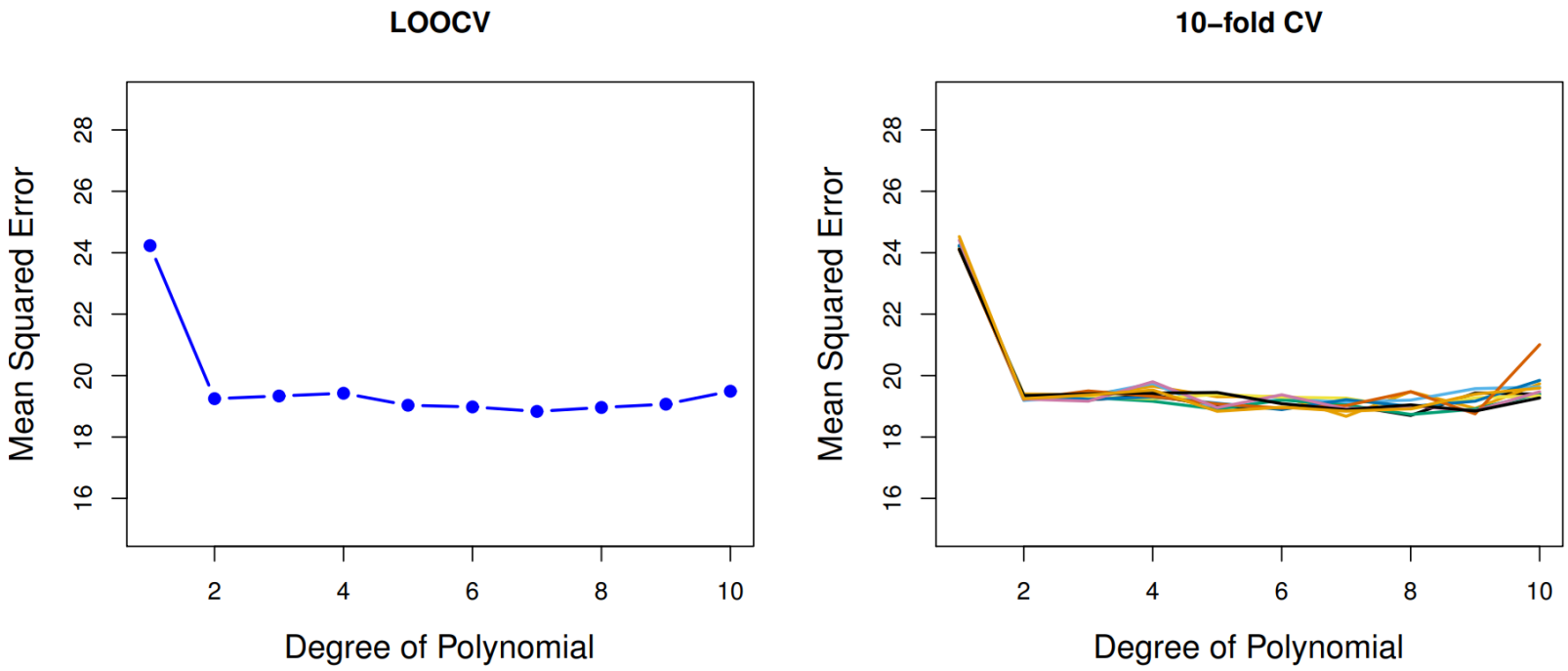
1. Randomly split the data set with $n$ observations into $k$ groups with approximatelyt $\frac{n}{k}$samples in each group.
2. Observations in the first group is used as a validation set, and the model is fit on the observations in the remaining $k-1$ groups. The fitted model is then applied to observation in the validation set. The error between the predicted response and the observed response is then used as an estimate of the test error $Err_i$.
3. The process 2. is repeated $k$ times, each time holding out a new group for validation. The test error is then estimated by averaging across the individual estimates

$$CV_k = \frac{1}{k}\sum_{i=1}^{k} Err_i$$

- 10-fold cross-validation is a very popular way of assessing model performance.

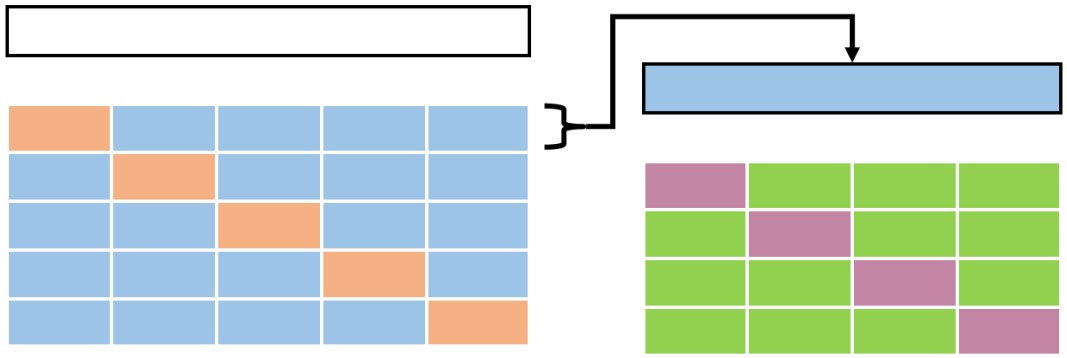


# K-fold Cross-Validation for model selection



# Model Selection & Model evaluation

- Often we are interested in both **a)** selecting a suitable model (and suitable model flexibility), and **b)** evaluate/assess the model's performeance.
- We should then use *nested* cross-validation – otherwise our estimate of generalization error may be too optimistic.
- Nested cross-validation: The data is split into $k$ parts for error estimation (lhs. below). For each training set, the training observations are again partitioned into e.g. $k-1$ parts (rhs. below) and cross-validation is used to select the best model (e.g. most optimal model complexity). The resulting model is then fitted to all training data in the trainng split and applied to observation in the validation split to estimate the generalization error.

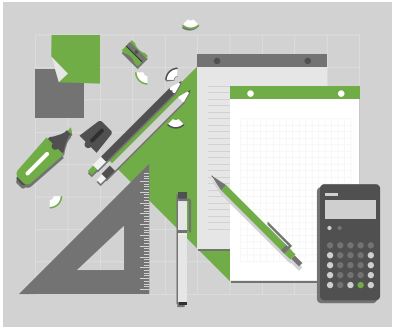


# Resampling, randomness, and reproducibility

- Initialize the random number generator in the beginning of your Matlab script so that you get the same results next time you run your analysis.

```
c = cvpartition(10,'KFold',5);
c.training(1)'
ans =

1×10 logical array
1 1 1 1 0 1 1 1 1 0

c = cvpartition(10,'KFold',5);
c.training(1)'
ans =

1×10 logical array
1 1 1 1 0 1 1 1 1 0
```

```
rng('default') % For reproducibility
c = cvpartition(10,'KFold',5);

c.training(1)'
ans =

1×10 logical array
1 1 1 1 0 1 1 1 1 0

rng('default') % For reproducibility
c = cvpartition(10,'KFold',5);

c.training(1)'
ans =

1×10 logical array
1 1 1 1 0 1 1 1 1 0
```



# In-class exercises

4 Model evaluation and resampling

# References

Figures from James et al. *An Introduction to Statistical Learning*, second edition, https://www.statlearning.com/resources-second-edition