

# Introduction to classification

**Ilya Narsky**

# Copyright

© COPYRIGHT 2017 by The MathWorks, Inc.

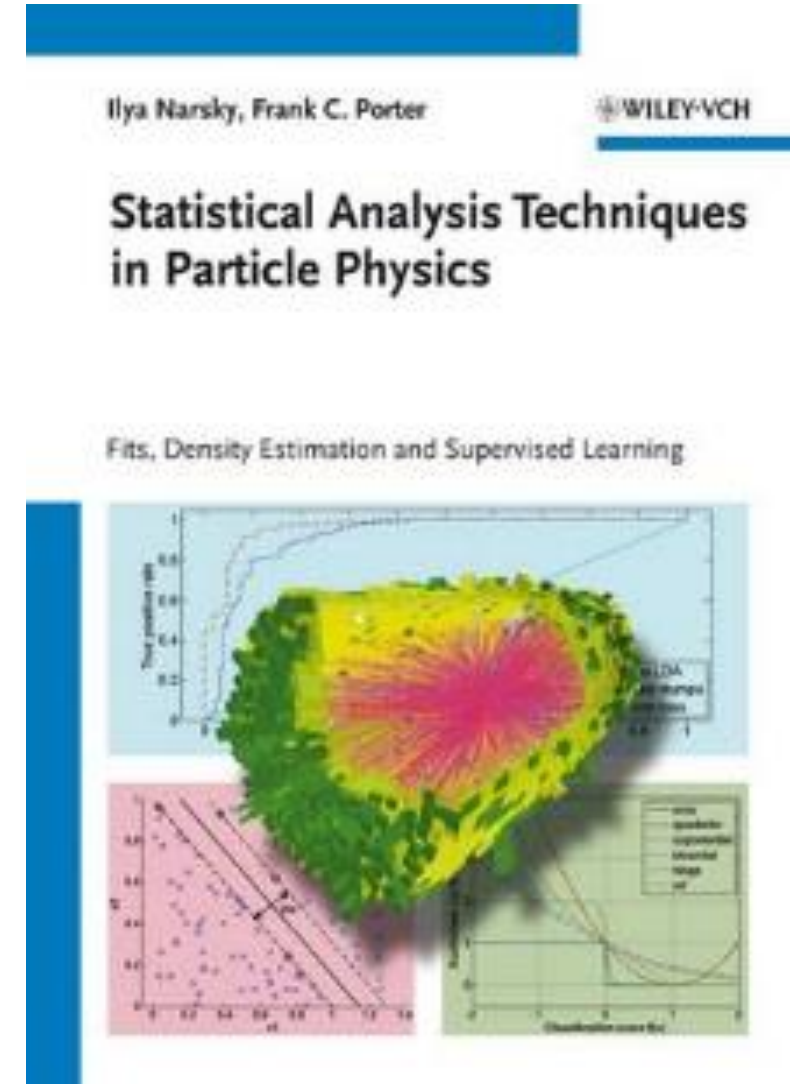
The materials of this training course shall at all times remain the intellectual property of The MathWorks, Inc. The MathWorks, Inc. reserves all rights in these materials. No part of these materials may be photocopied, reproduced in any form, or distributed without prior written consent from The MathWorks, Inc. The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement.

# About the lecturer

- Experimental high energy physics
  - 2001, PhD from Southern Methodist U
  - 2001-2008, researcher at Caltech
  - CLEO, BaBar, and CMS
- Software development at MathWorks since 2008
  - Implement various machine learning algorithms
- About MathWorks
  - The leading developer of mathematical computing software for engineers and scientists.
  - Headquarters near Boston with offices in 15 countries.
  - MATLAB and Simulink

# Statistical Analysis Techniques in Particle Physics (with Frank Porter)

- Most of the material covered in this course is discussed in the book.
- Table of content and ~10% of the book are available online.
- The book has exercises. Some solutions are posted at the Caltech site.
- The book has many MATLAB examples. All MATLAB examples can be downloaded from the publisher site (link at the Caltech site).
  - A bit outdated but all run.
- <http://www.hep.caltech.edu/~NarskyPorter/>

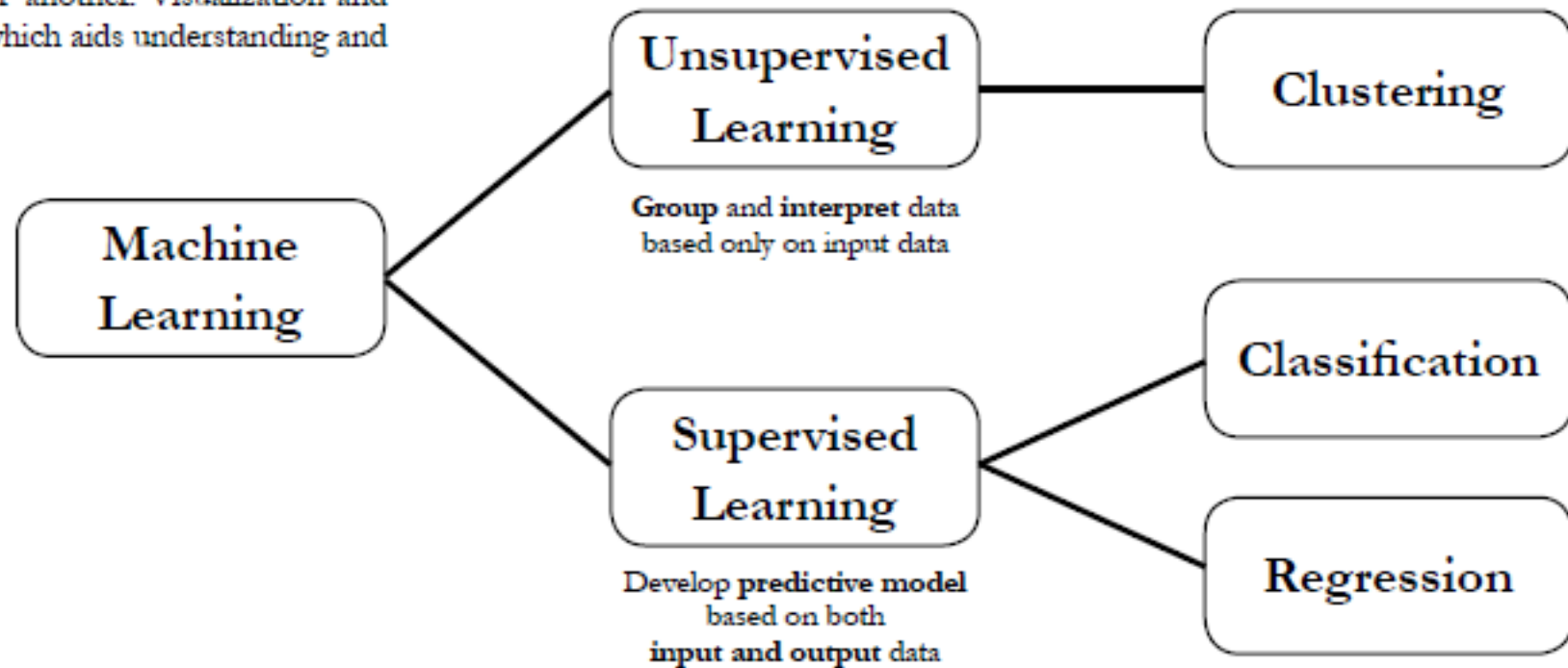


Give feedback in person or email  
to [inarsky@mathworks.com](mailto:inarsky@mathworks.com)

# Types of learning

- Supervised
- Unsupervised
- Feature learning and feature generation

uns use similar syntax, making of another. Visualization and which aids understanding and



# Jargon (aka conventions) and notation

- |  |  |
|--|--|
| <ul style="list-style-type: none"><li>▪ Observations</li><li>▪ Cases</li><li>▪ Samples</li><li>▪ Examples</li><li>▪ Events</li></ul> | <ul style="list-style-type: none"><li>▪ Variables</li><li>▪ Predictors<br/>(supervised learning)</li><li>▪ Features</li><li>▪ Attributes</li></ul> |
|--|--|

$$X_{N \times D}$$

N observations and D variables

$$x_{D \times 1}$$

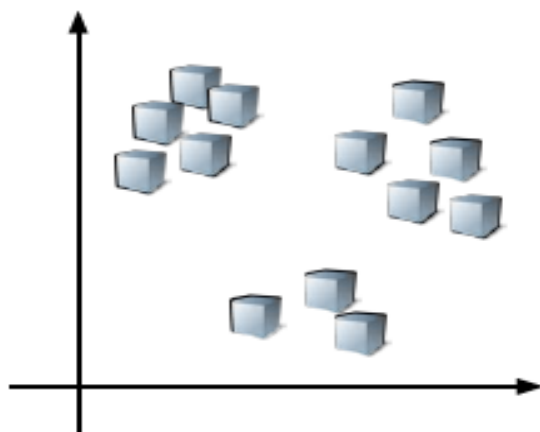
one multivariate observation

$$X_{D \times 1}$$

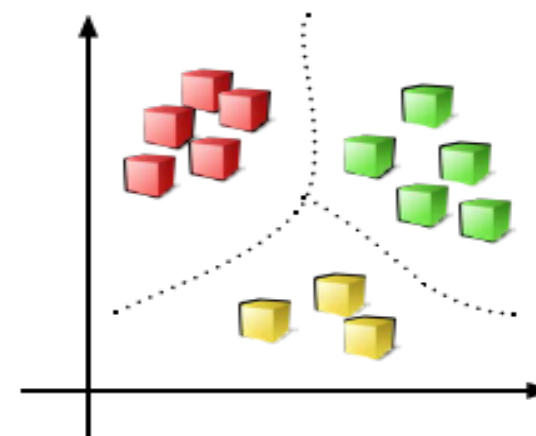
random variable

# What Is Machine Learning?

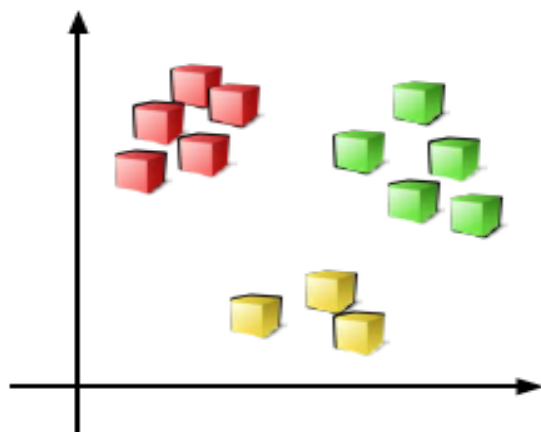
## Unsupervised




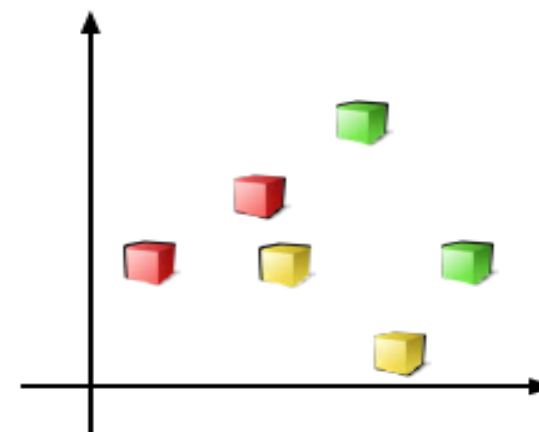
**Clustering**  
  
**Patterns in the data**



## Supervised



**Prediction**  
  
**Learn from examples**





# Classification and regression

**Statistical learning:**  
learn  $P(y|x)$

response  
(class label)



vector of  
observables

Response	Example	Type of learning
nominal $y$ (categorical unordered)	e, K, $\pi$ , p	classification $Y x \sim \text{Cat}(\{p_k(x)\}_{k=1}^K)$
ordinal $y$ (categorical ordered)	short, medium, long	classification for ordinal labels
continuous $y$	particle momentum	regression

## Bayes Rule in classification

$$P(x, y) = P(y|x)P(x) = P(x|y)P(y)$$

Assumed or specified by prior knowledge (such as, e.g., estimate of a branching fraction for a decay)

Learned by the model

$$P(y|x) = \frac{P(x|y)P(y)}{\sum_k P(x|k)P(k)}$$

### Models in this class

- Naïve Bayes:  $P(x|y)$  factorizable into a product of marginal pdf's
- Discriminant analysis:  $P(x|y)$  is multivariate normal

# Practical application of the Bayes Rule

Being killed by lightning  $Y \in \{0,1\}$

Being outside during lightning  $X \in \{0,1\}$

$$P(y = 1) = \frac{1}{7,000,000}$$

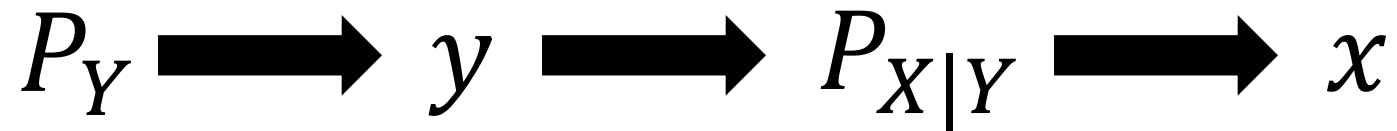
$$P(x = 1|y = 1) = 1$$

$$P(y = 1|x = 1) = \frac{P(x = 1|y = 1)P(y = 1)}{P(x = 1)}$$

$$P(y = 1|x = 1) = \frac{1}{7,000,000 \cdot P(x = 1)}$$



**Generative models**  $P_{X,Y}(x, y) = P_{X|Y}(x|y)P_Y(y)$



Generate a point in space and its class label

Examples:

- (Fisher) discriminant
- Naïve Bayes (aka “projection” method)

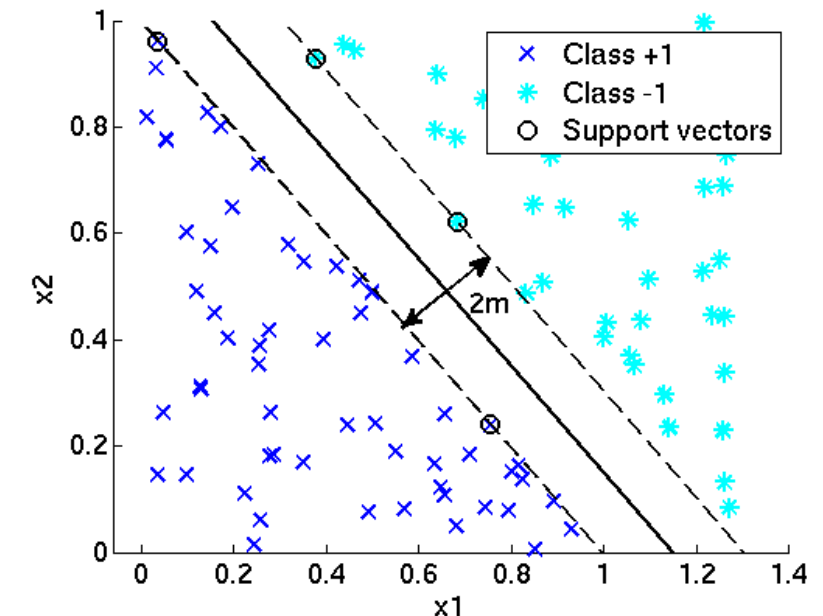
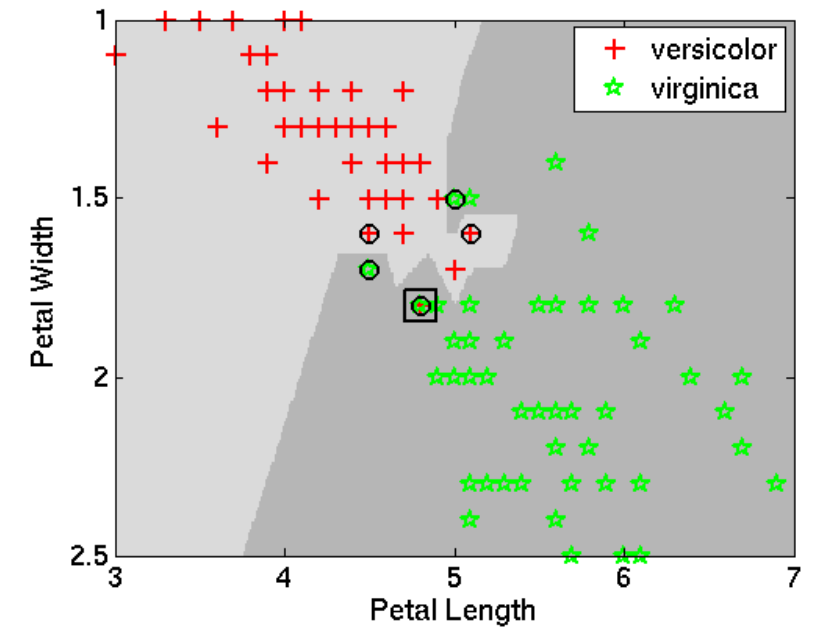
**Benefits:**

- If  $P_{X|Y}$  is analytically known, all distributions can be analytically computed
- $P_X(x)$  can be used to find outliers

# Sometimes we do not need probability

## (A bit less than) statistical learning

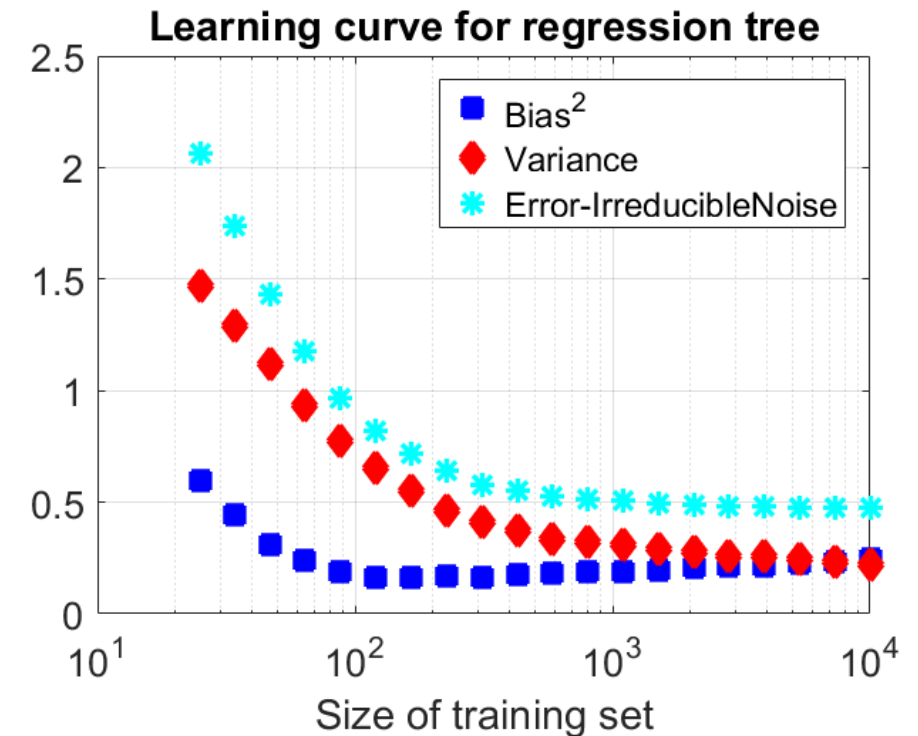
- Find a decision boundary between two classes (or decision boundaries for multiple classes)
- Examples:
  - Nearest neighbor rules
  - Support Vector Machines
- Estimates of classification confidence (scores) are usually available
- Classification scores can be often transformed to posterior probabilities  $P(y|x)$  by some rule



# The learning curve (regression)

For continuous response, at fixed  $x$

- Observed  $Y$  and predicted  $F$
- Mean squared error  $\epsilon^2 = E[(Y - F)^2]$
- Irreducible noise  $\text{Var } Y = E[(Y - EY)^2]$
- Prediction bias (squared)  $B^2 = (EF - EY)^2$
- Prediction variance  $\text{Var } F = E[(F - EF)^2]$
- $\epsilon^2 = \text{Var } Y + B^2 + \text{Var } F$





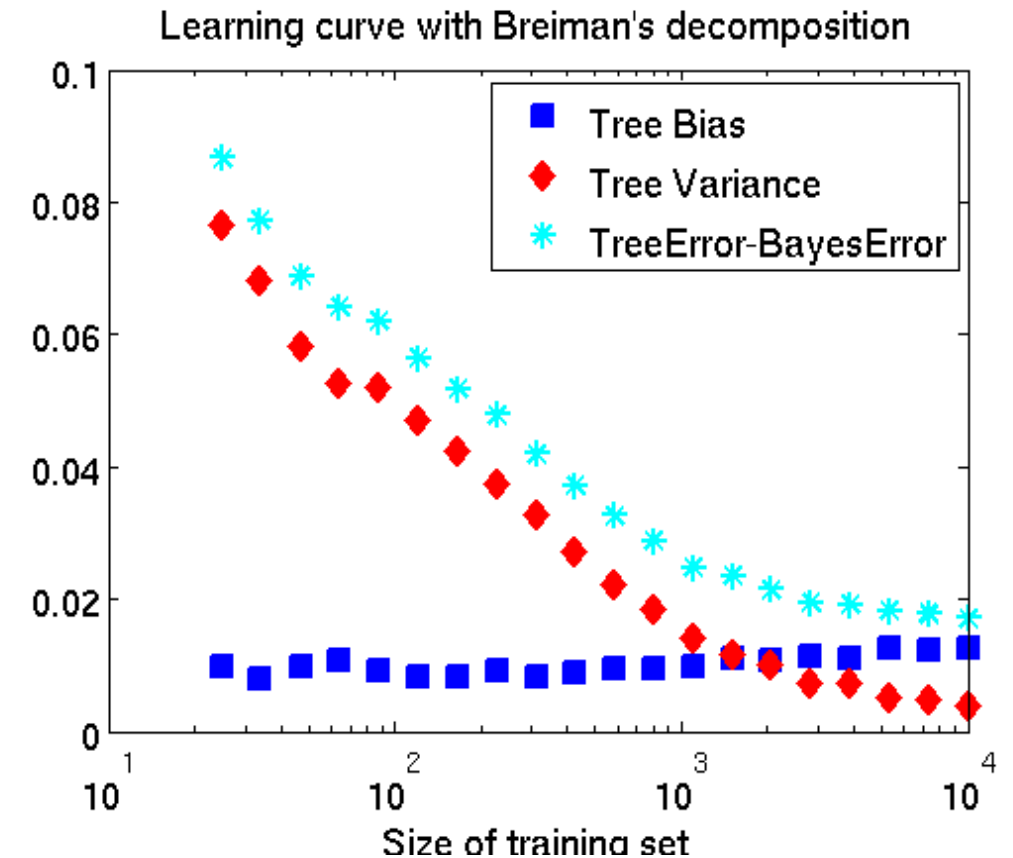
# The learning curve (classification)

For binary response, at fixed  $x$

- Error  $P(\hat{Y} \neq Y)$ 
  - that is, fraction of misclassified observations
- Bayes error  $1 - P(y^* | x)$
- Prediction bias ...
- Prediction variance ...

In classification, there are no agreed-upon definitions. I am using Breiman's definitions for classification tree.

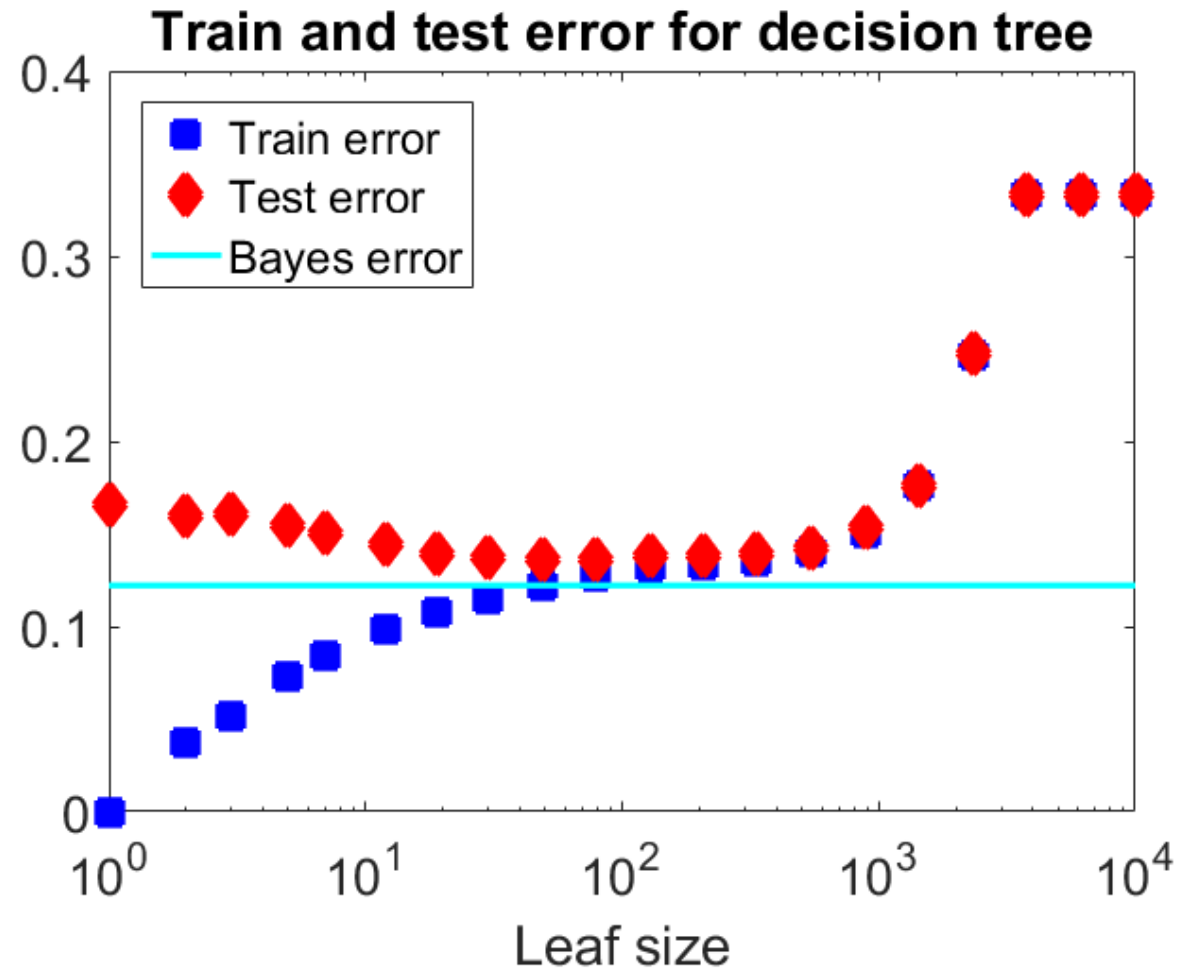
- $Y^*$  = most probable class at  $x$
- $Y$  = observed class at  $x$
- $\hat{Y}$  = predicted class at  $x$

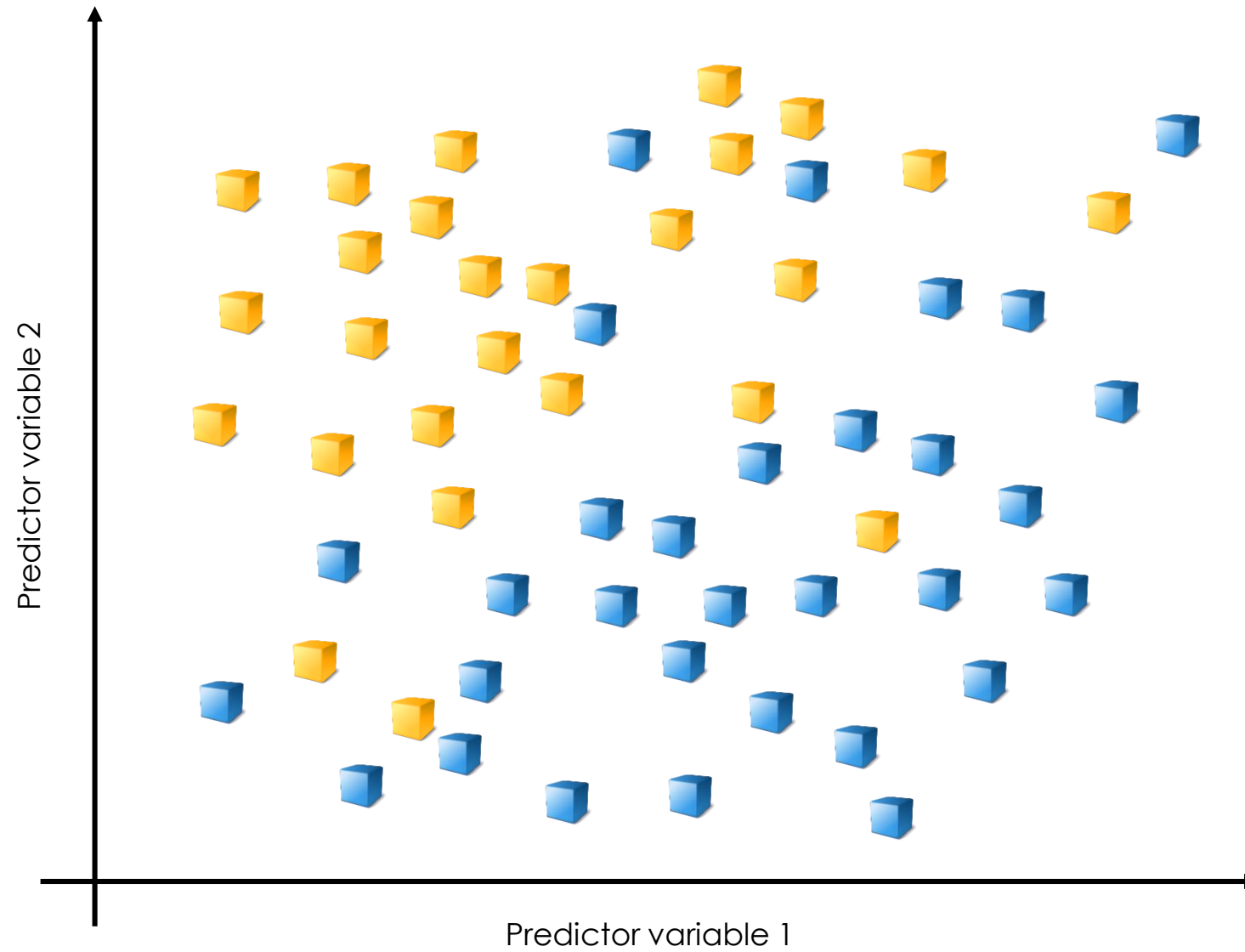


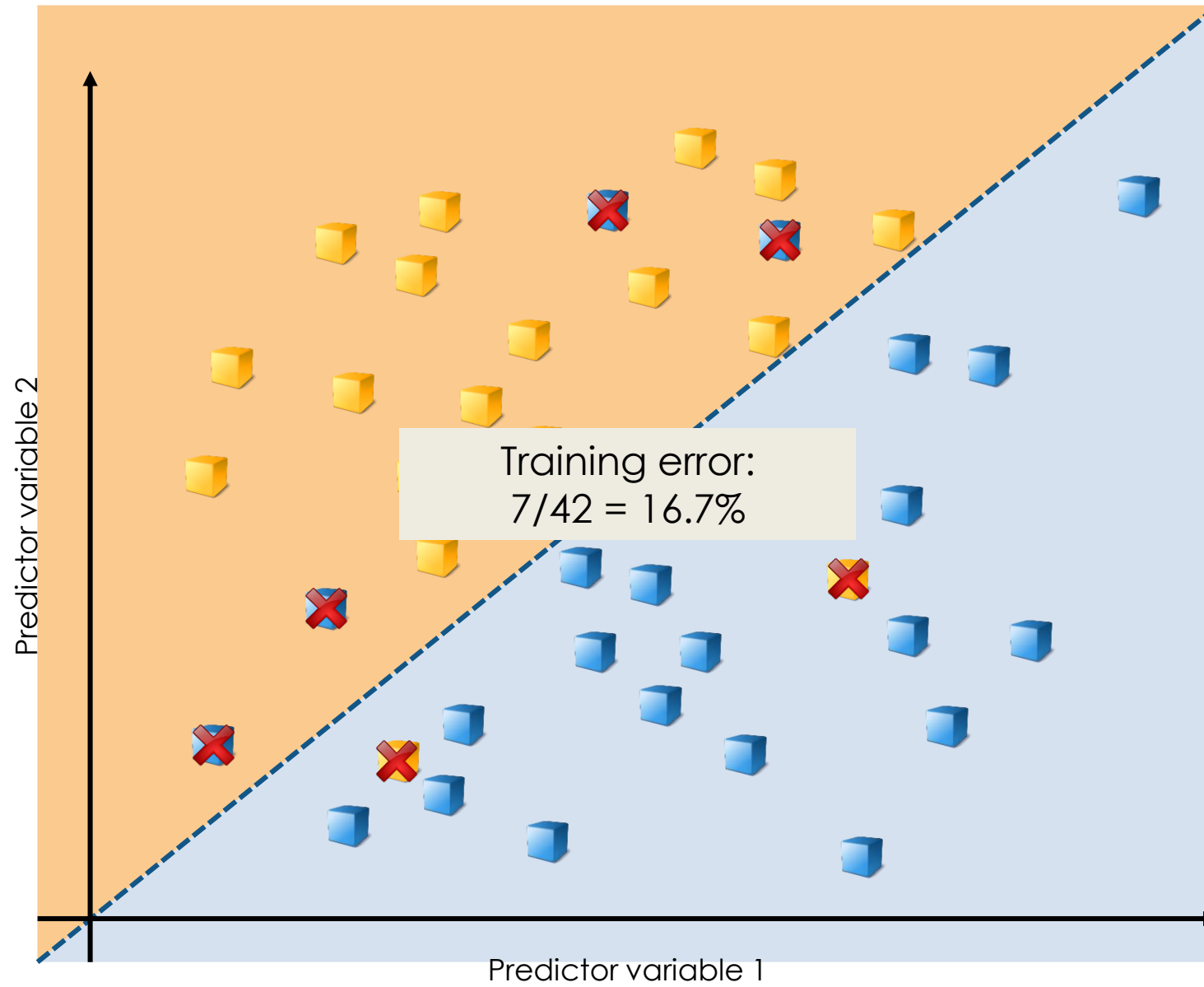
**In classification, Bayes error = irreducible noise**

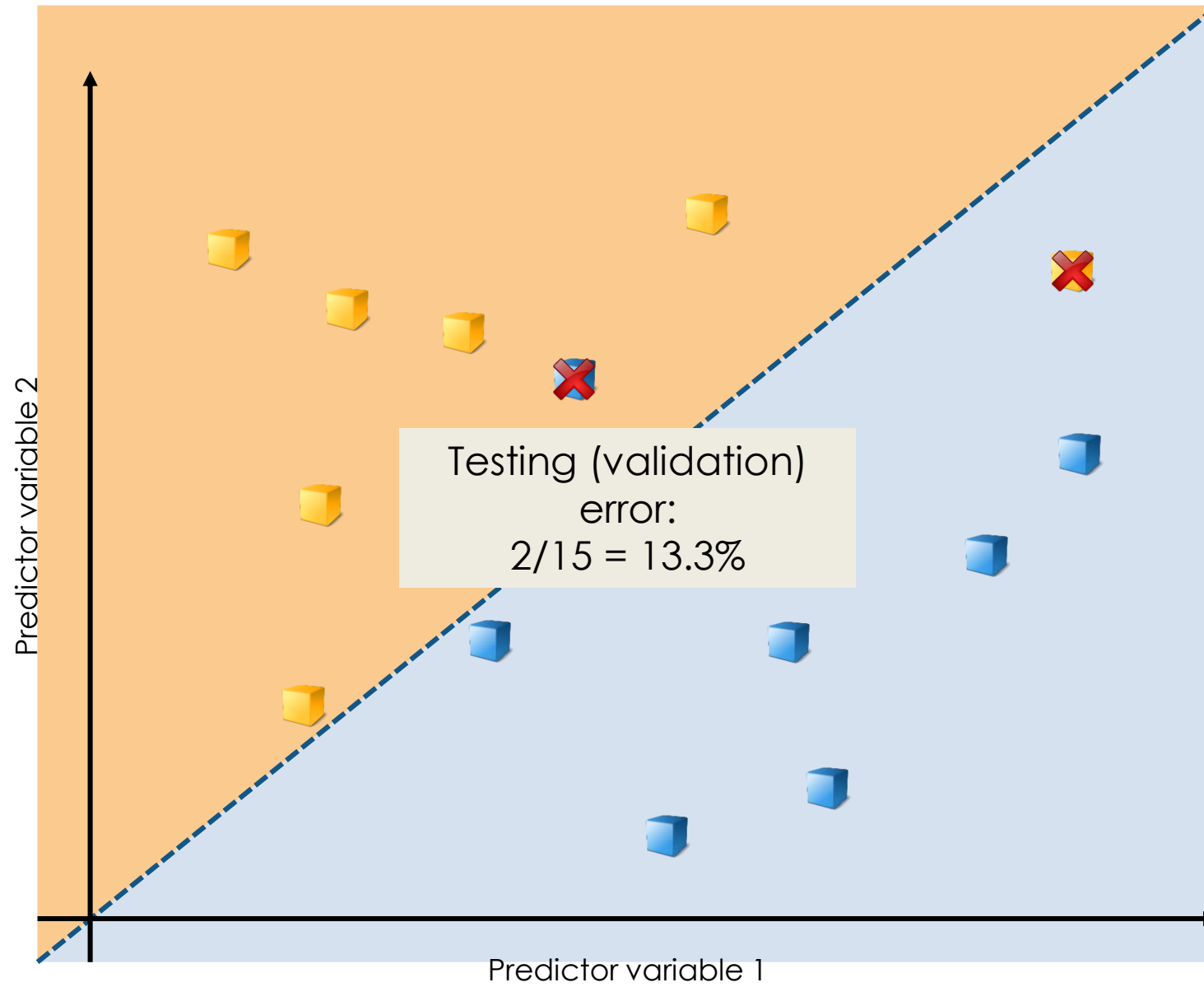
# Overtraining

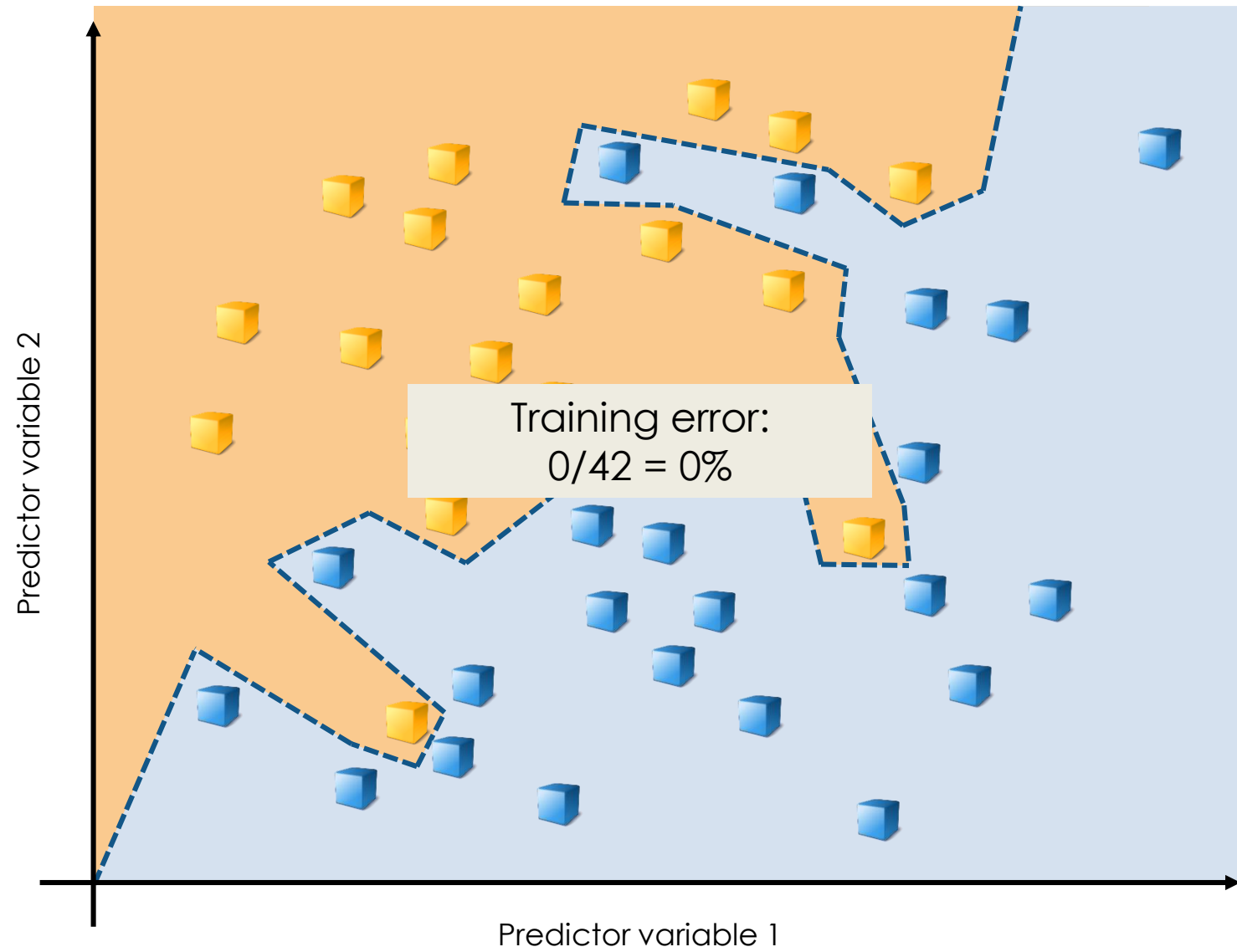
- Training, or resubstitution error
  - measured on the training set
- Generalization error
  - defined on the true distribution  $P(x, y)$
  - needs to be estimated
- It is not unusual for training error to be much less than generalization error
- Example for a single decision tree

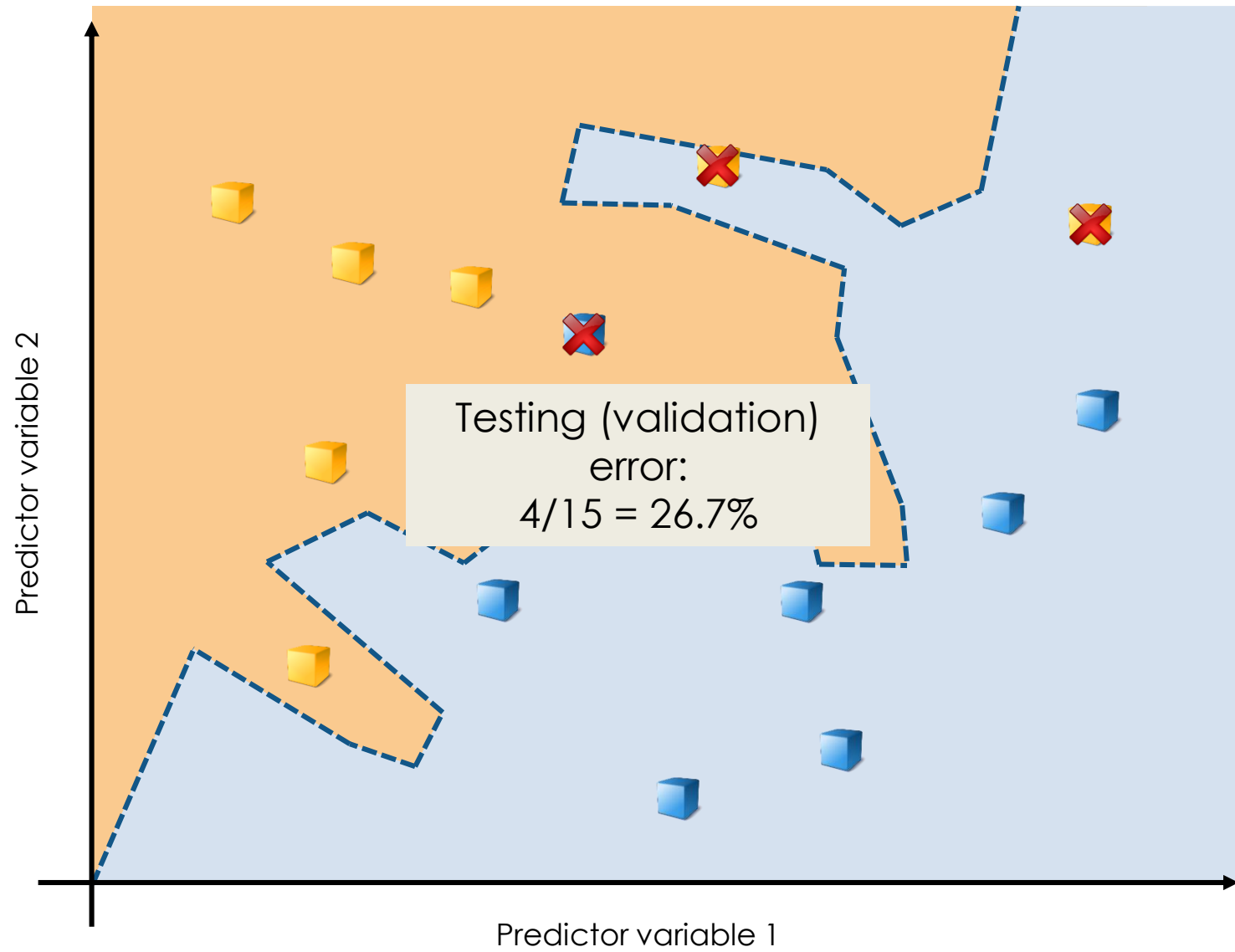






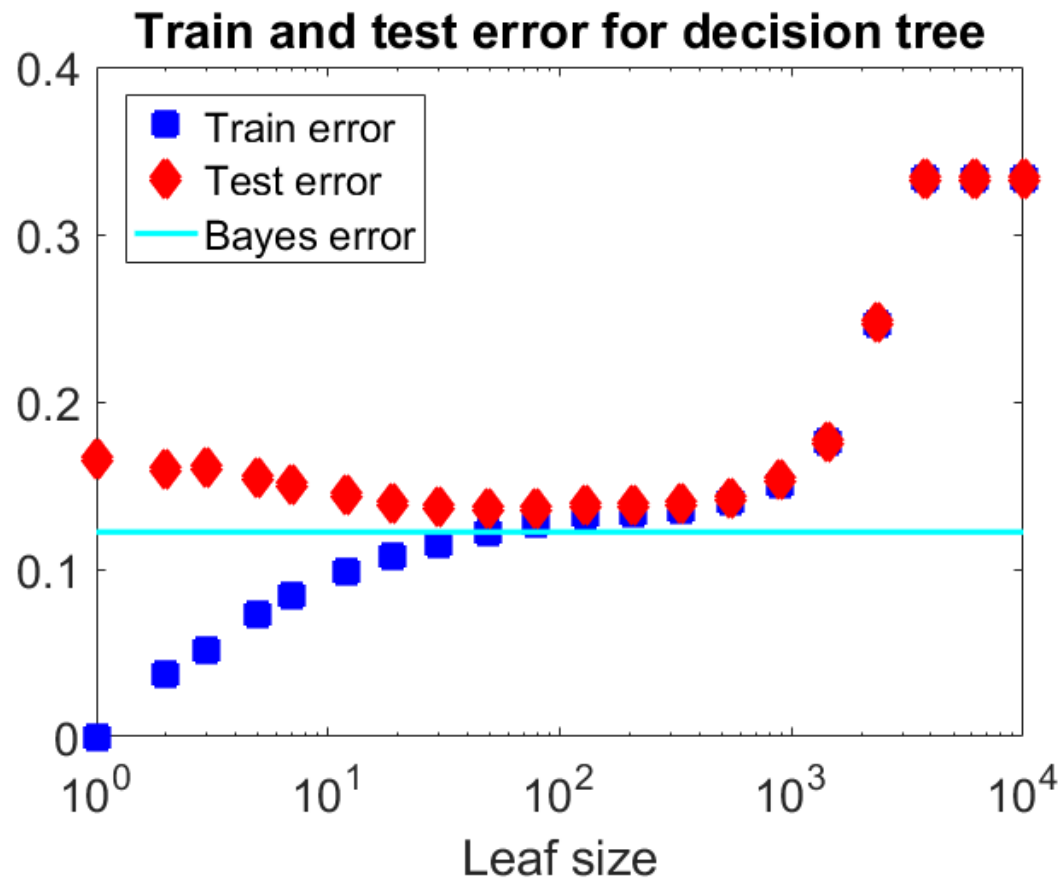




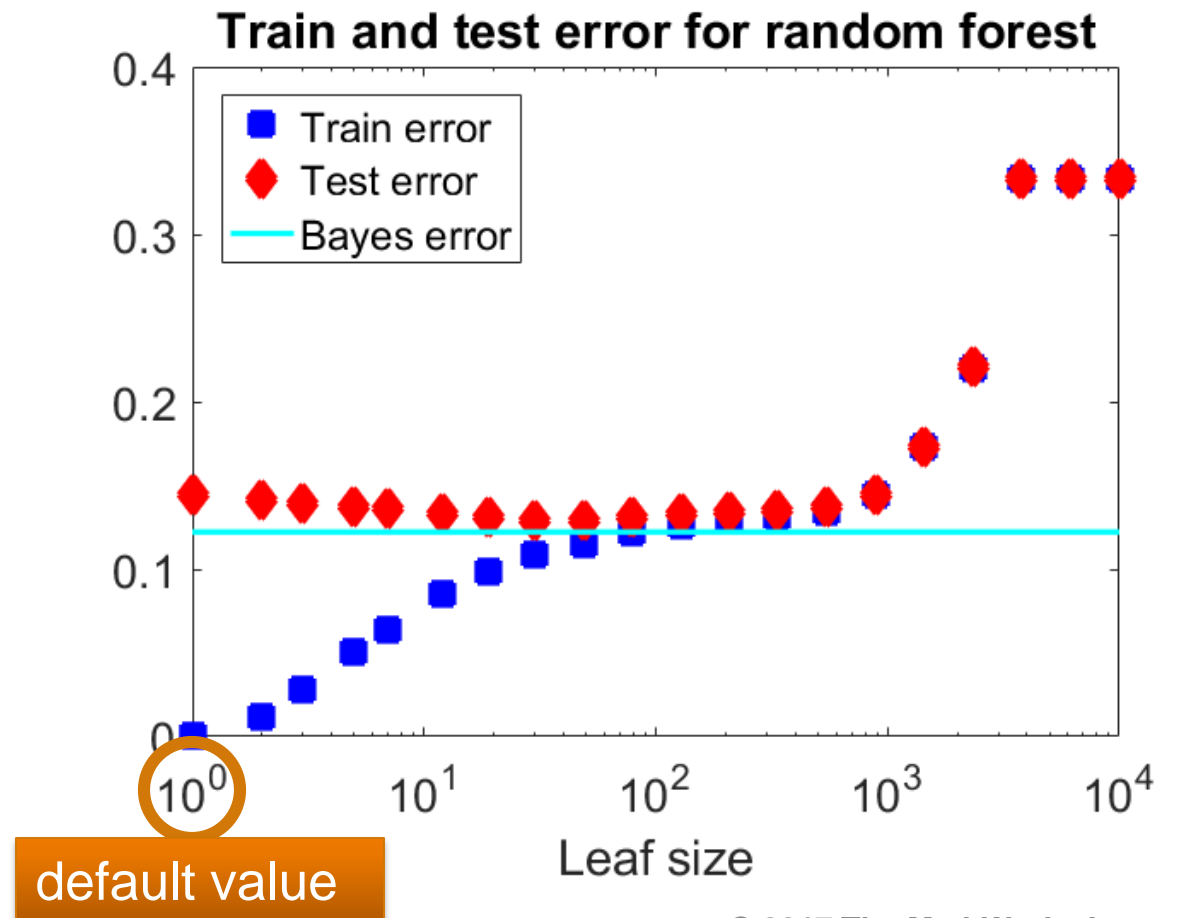


# Overtraining for simple and complex models

**Simple models: Optimal accuracy attained when training and test errors are equal.**



**Complex models: Various patterns emerge.**





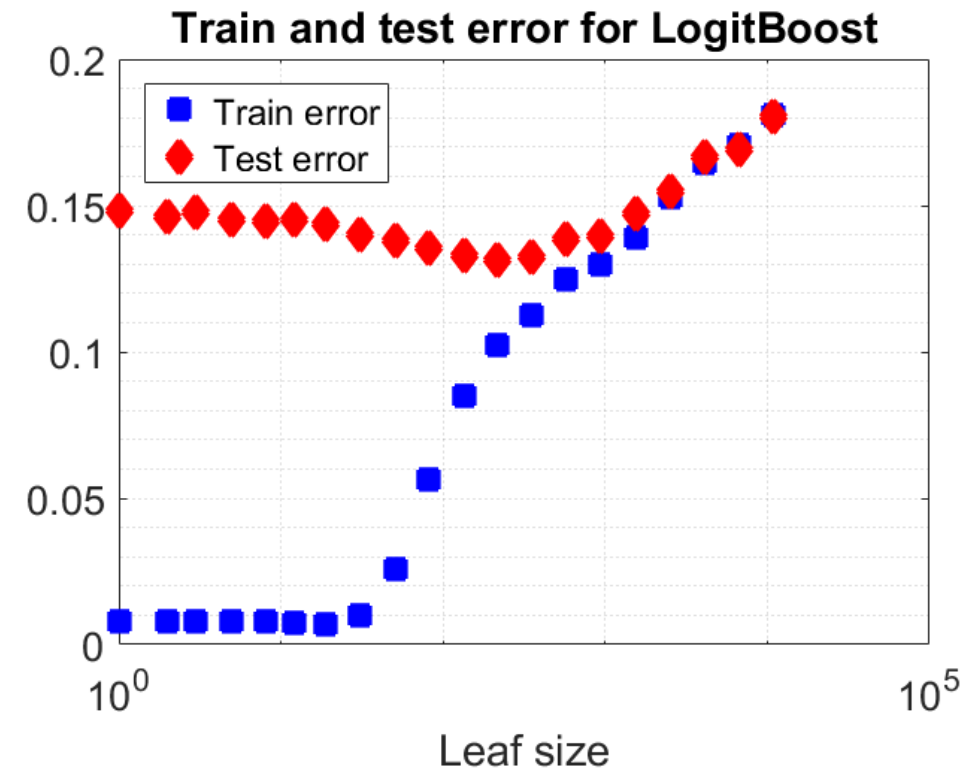
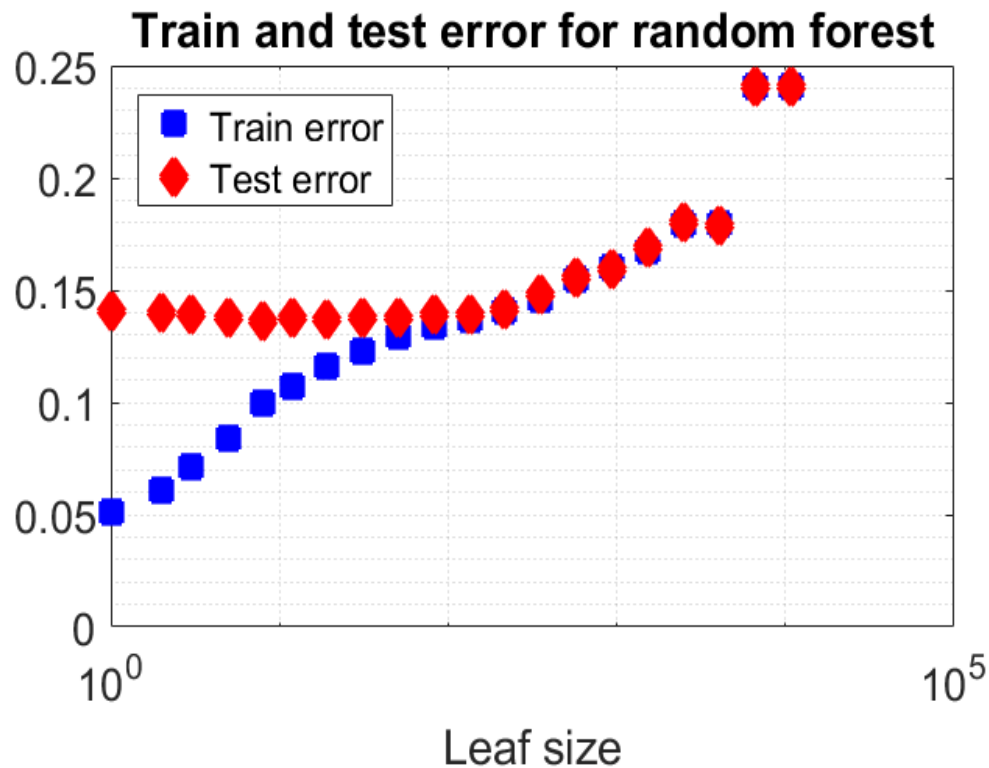
# Overtraining for complex models

**Adult dataset (1996) from UCI repository:**

- Predicting if annual income exceeds \$50k
- 6 continuous and 8 categorical variables
- 33k training and 16k test observations

**Algorithms**

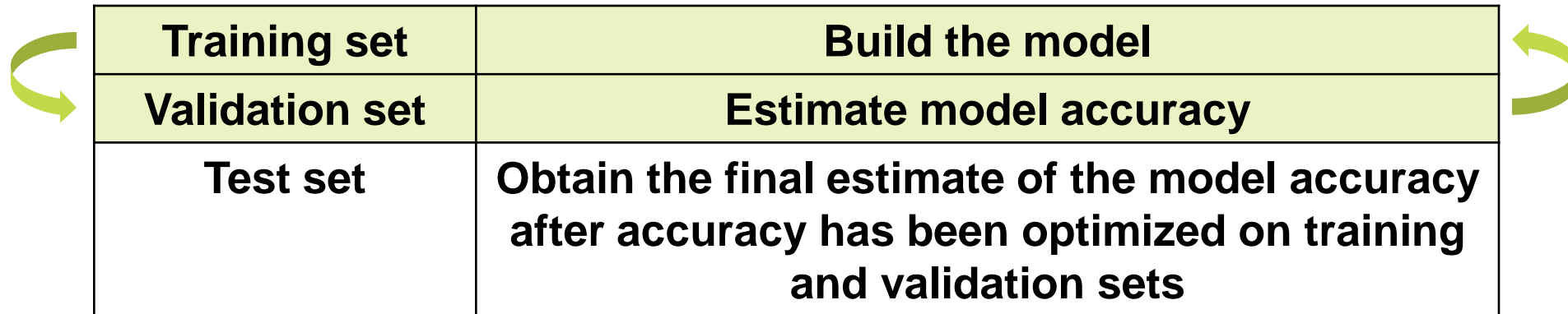
- Random forest with 100 trees
- LogitBoost with 200 trees and learning rate 0.1
- Vary the minimal tree leaf size



## Overtraining: takeaways

- For simple models, optimal accuracy typically is attained when training and test errors are equal.
- For complex models, there is no obvious relation between training and test errors. It is not unusual for the training error to be much lower than the test error at the **optimal** classification settings.
- **Training error does not matter.** Only test error is a reliable indicator of performance.

# Train, validate and test

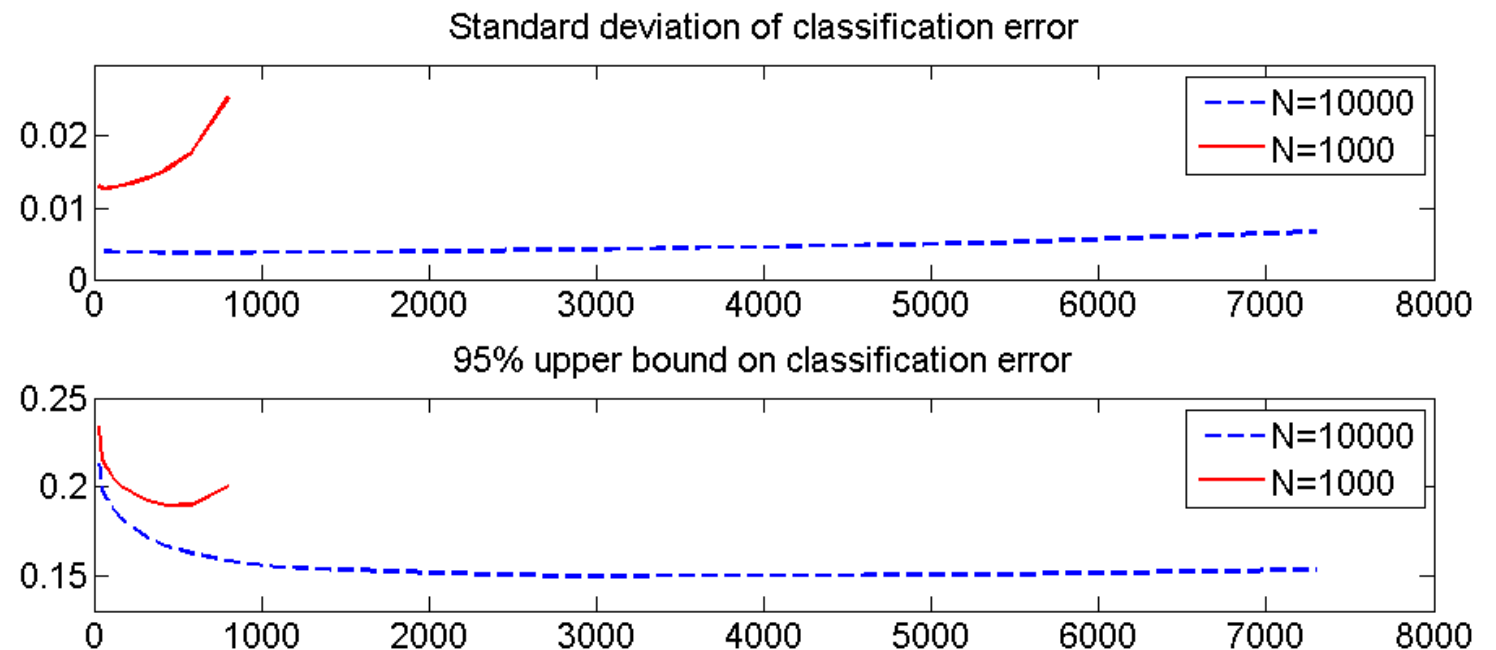


<b>Training set</b>	<b>Build the model</b>
<b>Validation set</b>	<b>Estimate model accuracy</b>
<b>Test set</b>	<b>Obtain the final estimate of the model accuracy after accuracy has been optimized on training and validation sets</b>

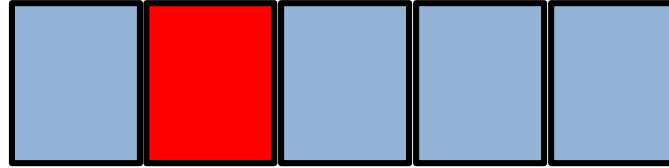
Can use cross-validation instead of a validation set

# Holdout validation

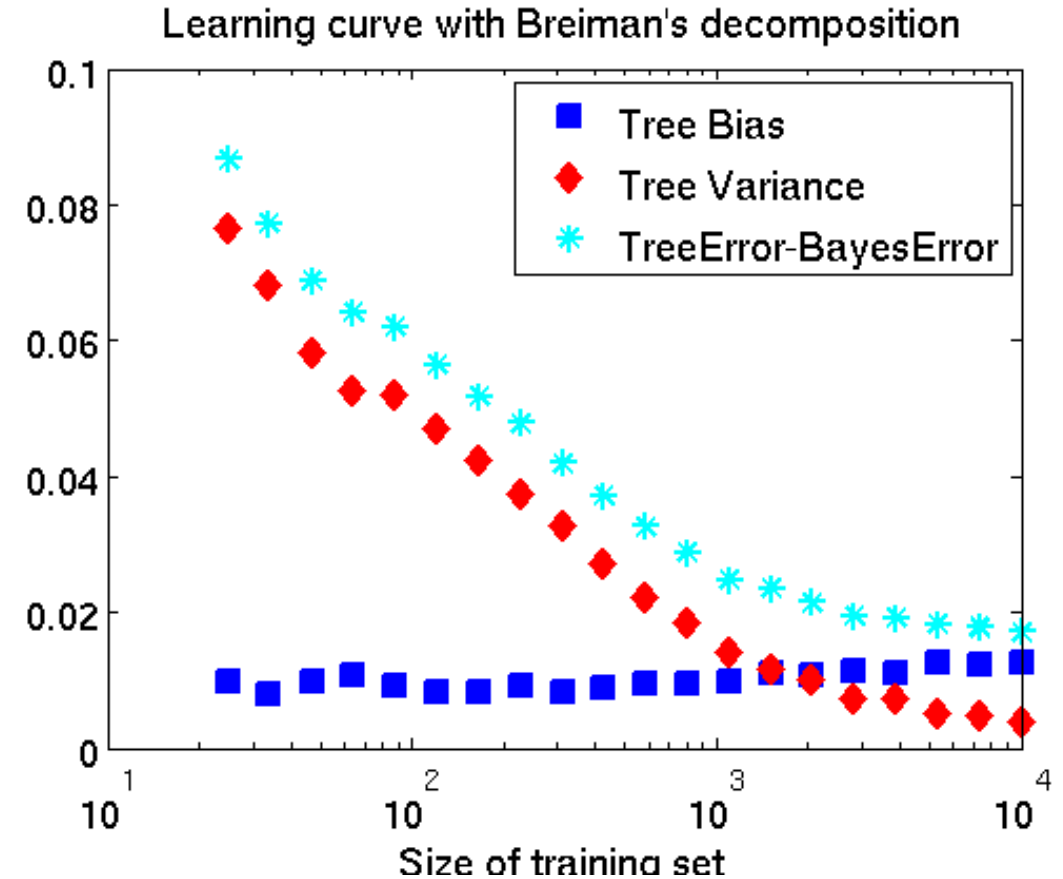
- Train on  $z\%$ , measure accuracy on  $(100-z)\%$
  - Not recommended for small datasets
  - For small datasets, use cross-validation
  - Optimal  $z$ ?
- Have a fixed number of observations  $N$
  - Need to split  $N = N_{\text{train}} + N_{\text{test}}$
  - Minimize  $\hat{\varepsilon}_{\text{test}} \Rightarrow$  need large  $N_{\text{train}}$
  - Minimize conf. interval on  $\hat{\varepsilon}_{\text{test}} \Rightarrow$  need large  $N_{\text{test}}$



# Cross-validation

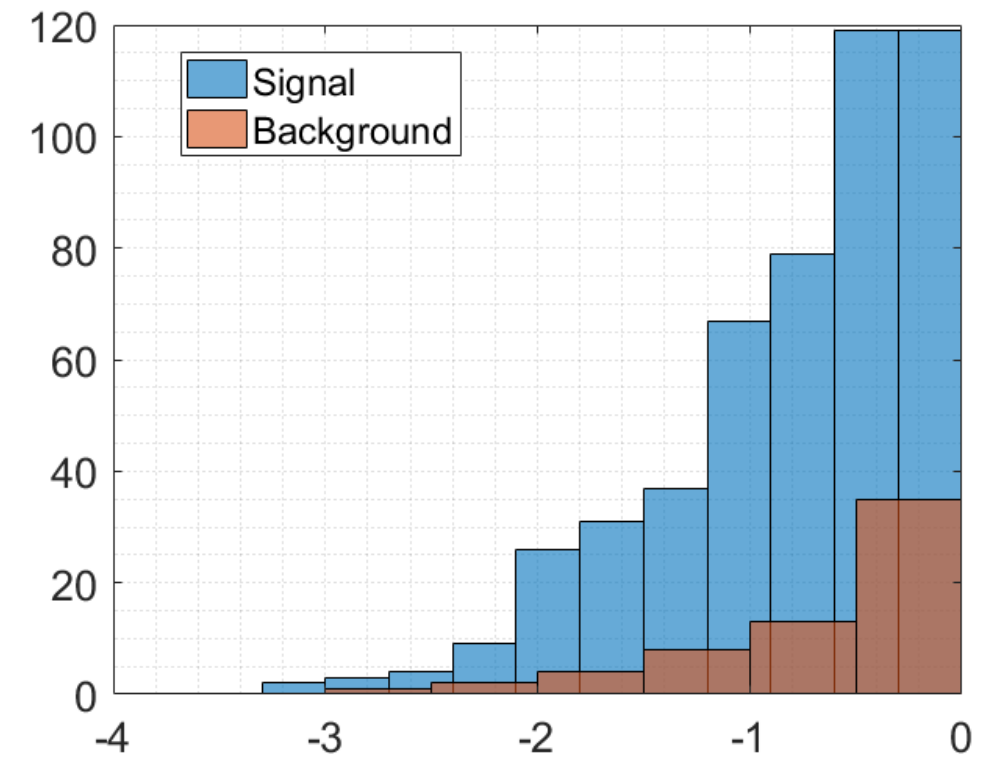
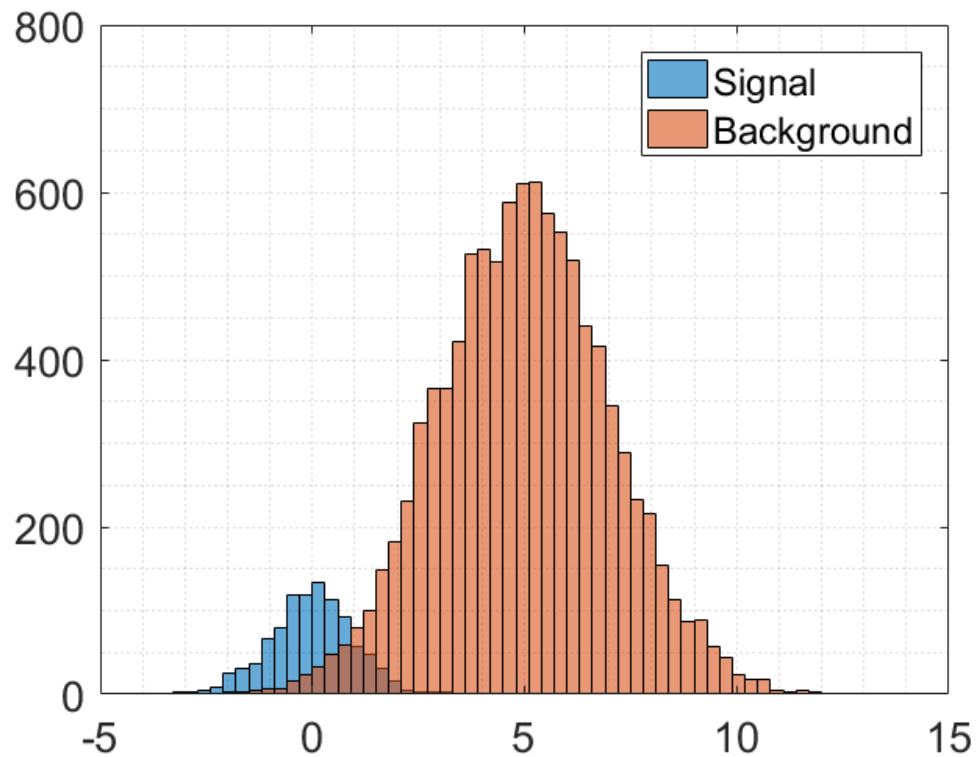


- CPU-intensive
- Generally not needed for large datasets
- **Two-fold CV**: Loss of accuracy due to halving the dataset
- **Leave-one-out (LOO) CV**: Can underestimate variance of prediction because models in folds are very similar: They are essentially trained on the same dataset!
- Use **5 or 10 folds** as a rule of thumb



## “Small” and “large” datasets

- Size is determined by the size of the interesting subset
  - For example, number of events in the signal region



# Stratified and non-stratified partitioning

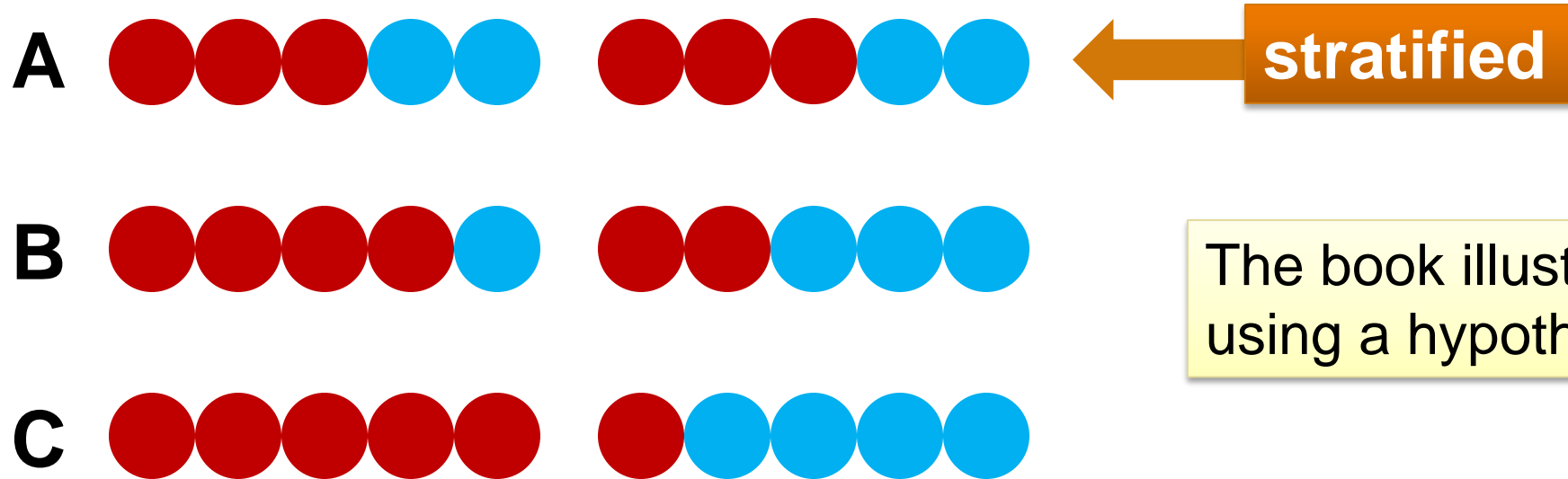


Partition in two sets (training and test)



**What partition is best?**

## Stratified and non-stratified partitioning



The book illustrates this problem using a hypothetical example.

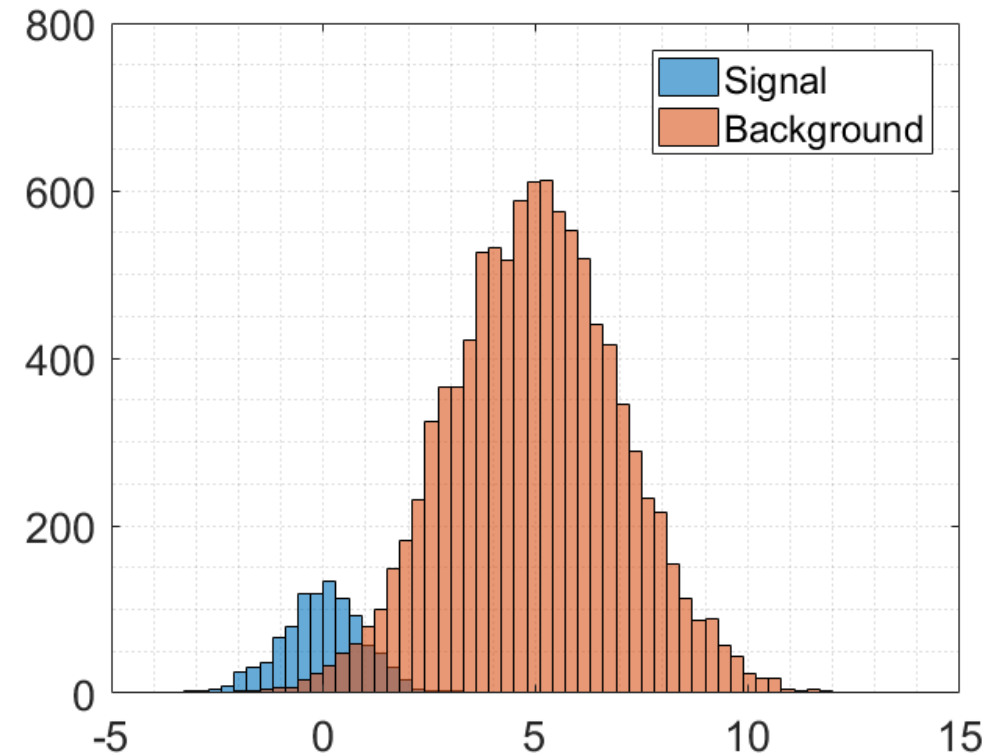
### For estimation of the classifier accuracy:

- Stratification reduces variance of the accuracy estimate.
- Good if you want to obtain an accurate estimate.
- Bad if you want to estimate uncertainty of the accuracy estimate.



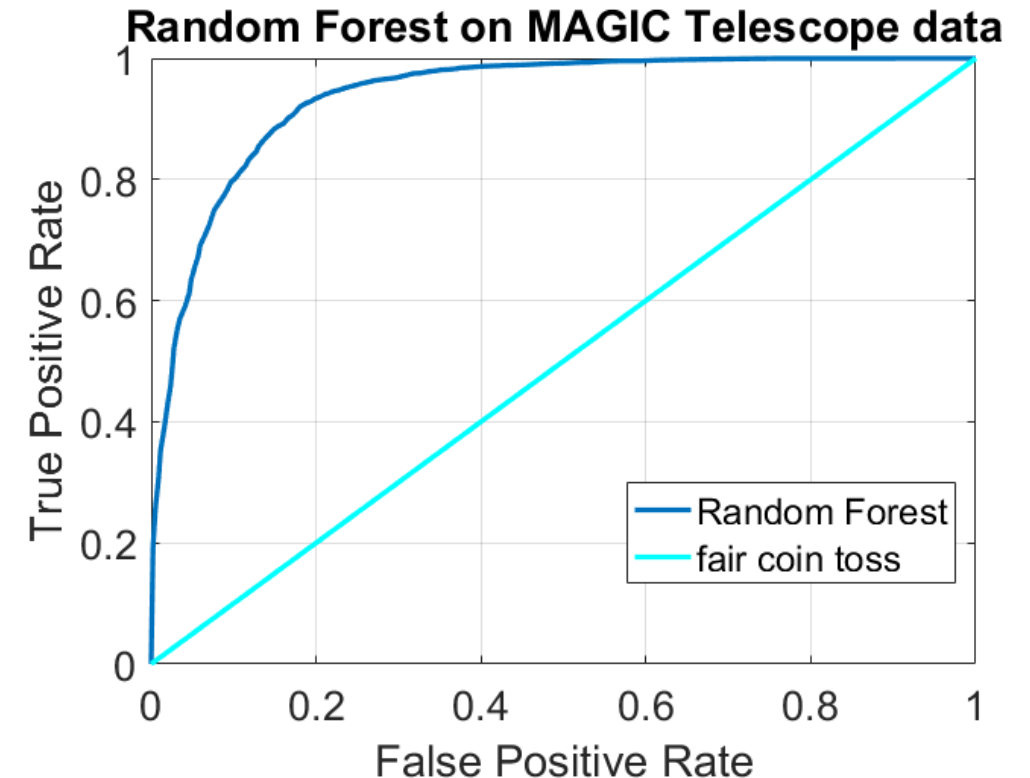
# Learning on imbalanced data

- Class imbalance
  - Training set has many more observations of one class than of another class
  - Common situation in physics analysis
- Popular approaches
  - Set class prior probabilities (class weights) to uniform
  - Undersample the majority class: RUSBoost
  - Oversample the minority class: SMOTE and SMOTEBoost
- More techniques in the book



# Learning on imbalanced data

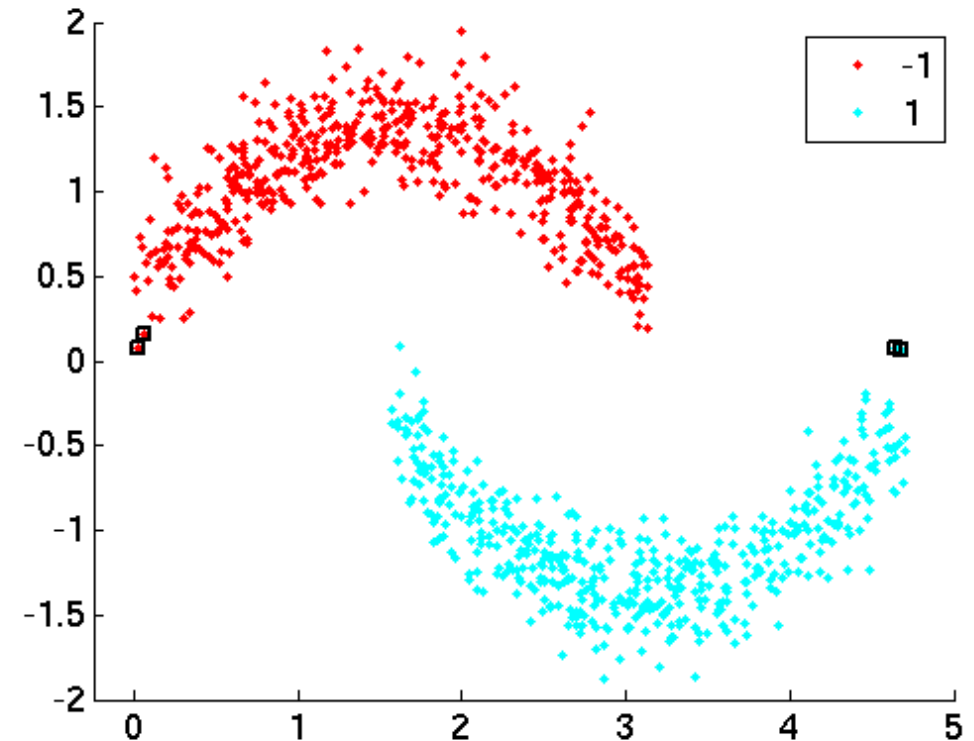
- Choose the technique based on how the classification model works and what the software supports
- Detailed discussion in the book
- Practical advice:
  - Do something special only if the class imbalance is large (an order of magnitude or more).
  - If the class imbalance is small (factor of 2 or 3), likely no need to worry. Just find the optimal point on the Receiver Operating Characteristic curve, the way you usually do.



**More types of learning (something you might want to hear about and quickly forget)**

# Semisupervised learning

- Some observations in the data are labeled (true class is known)
- Need to label the rest
- Supervised approach:
  - Learn on labeled data and classify unlabeled observations.
  - What if the labeled set is small?
- Unsupervised approach
  - Cluster and then assign each cluster into the same class
  - May be inaccurate
- Specialized techniques



## Active learning

- Subclass of semisupervised learning
- Querying class label is expensive and can only be done for a few observations
- Deciding which observations should be queried is part of the algorithm

# Online learning

- Data become available bit by bit
- The model needs to be updated on the next bit of data
- Can be used to process big data
- Learning on i.i.d. data (identical, independently distributed)
  - May have to account for a drift
- Learning with adversaries
  - Play a game with an opponent

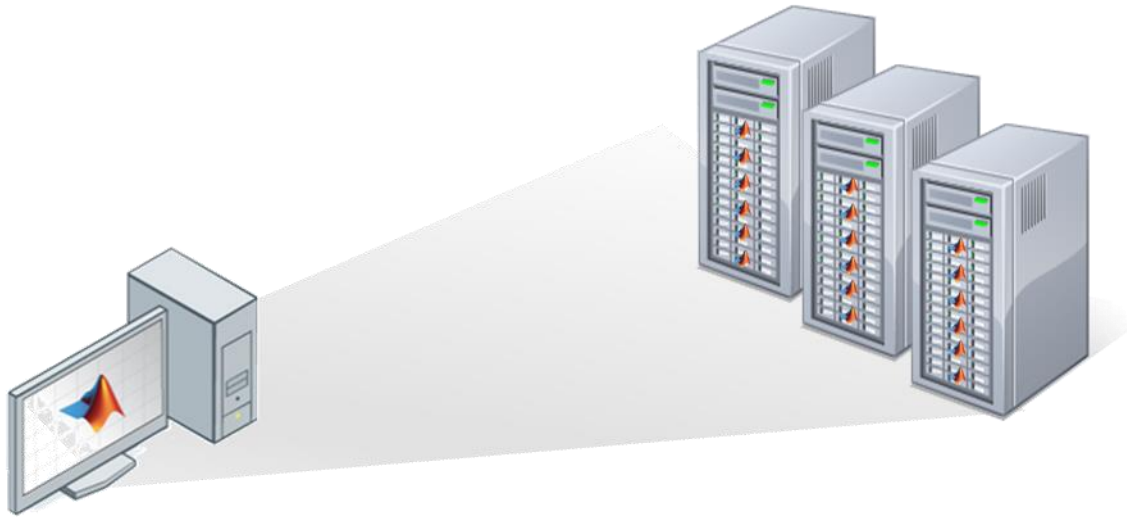
## Steepest Descent

- Minimize  $L(X, \theta)$  wrt  $\theta$  over set of  $N$  observations  $X$
- Descend in the direction of  $-\nabla_{\theta} L(X, \theta)$

## Stochastic Gradient Descent

- Minimize  $L(X, \theta) = \sum_{n=1}^N \ell(x_n, \theta)$
- Select observation  $n$  at random or get the next observation from the source
- Descend in the direction of  $-\nabla_{\theta} \ell(x_n, \theta)$

# Learning on big data

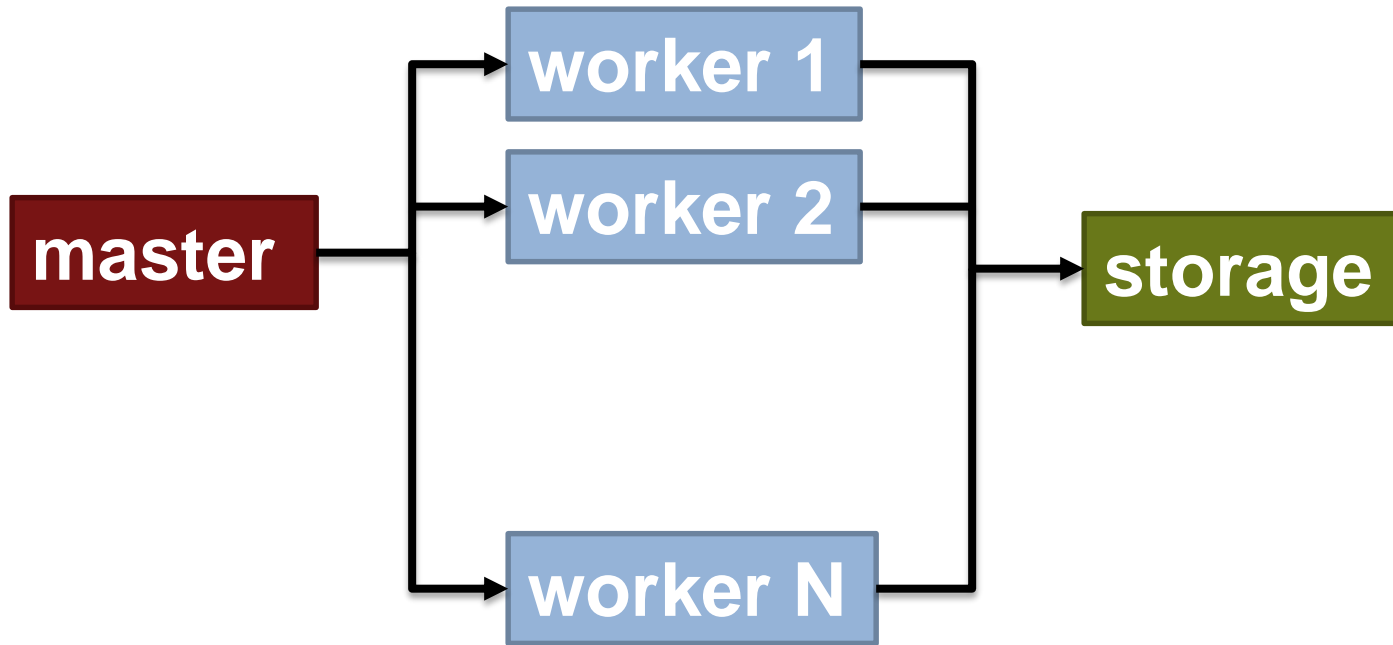


- Consider minimization of an objective function
- “Embarrassingly parallel” algorithms: Objective is a sum of many independently evaluated contributions,  $L(\mathbf{X}, \theta) = \sum_{n=1}^N \ell(x_n, \theta)$

## Example:

- Distribute computation of the objective  $L(\mathbf{X}, \theta)$  and gradient  $\nabla_{\theta} L(\mathbf{X}, \theta)$  across worker nodes
- Each node handles a chunk of data  $\mathbf{X}$

# Learning on big data: Computational issues



- Local data (available to workers) resides in storage
  - limited by network bandwidth AND disk I/O
- Local data can be kept in worker memory between passes
  - limited by network bandwidth

- One-pass algorithms

- $\hat{\mu} = \frac{1}{N} \sum_{n=1}^N x_n$
- $\widehat{\sigma^2} = \frac{1}{N} \sum_{n=1}^N x_n^2 - \hat{\mu}^2$

- Two-pass algorithms:

- $\widehat{\sigma^2} = \frac{1}{N} \sum_{n=1}^N (x_n - \hat{\mu})^2$

- Multi-pass algorithms:
  - Most ML algorithms in this course

# End of lecture 1