Mathematical techniques in data science

Lecture 1: General information and Introductions

General information

- Classes:

  Tue-Thu 5:20pm–6:40pm, Ewing Hall 207
- Office hours (starting from the 2nd week):
  - Tue-Thu 3:30pm–4:30pm, Ewing Hall 312
  - By appointments
- Instructor: Vu Dinh
- Website:

  `https://vucdinh.github.io/m637s24`

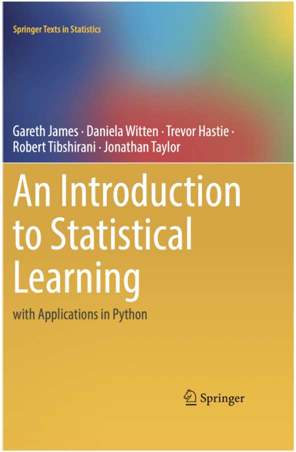
Data science

- is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from data, both structured and unstructured
- is a concept to unify statistics, AI, data analytics, machine learning and their related methods in order to understand and analyze data
- employs techniques and theories drawn from many fields: mathematics, statistics, information science, and computer science

Goals of the course

- Become familiar with the basic methods used to analyze modern datasets
- Be able to analyze datasets using Python
- Understand how to select a good model for data
- Understand the mathematical theory and the standard models used in data science

Textbook



Springer Texts in Statistics

Gareth James · Daniela Witten · Trevor Hastie · Robert Tibshirani · Jonathan Taylor

An Introduction to Statistical Learning

with Applications in Python

Springer

An Introduction to Statistical Learning. James, Witten, Hastie, Tibshirani, and Taylor.

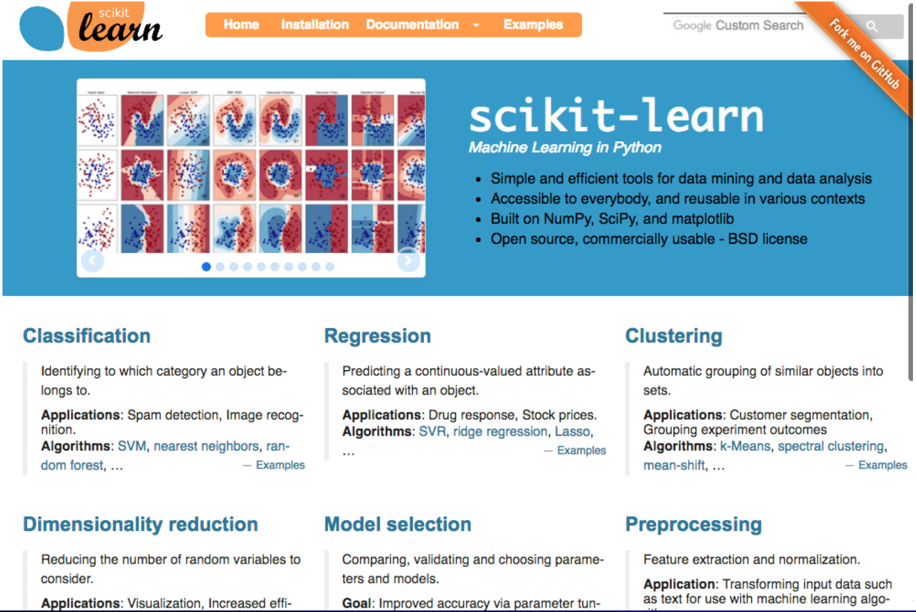
The pdf of the book is available at `https://www.statlearning.com`

Topics

The materials of the course can be organized

- By problems:
  - Classification
  - Regression
  - Clustering
  - Manifold learning
- By methods:
  - Regression-based methods
  - Tree-based methods
  - Network-based methods
- By meta-level techniques:
  - Regularization
  - Kernel trick
  - Boosting
  - Bootstrapping

scikit-learn



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, … — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, … — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, … — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased effi-

Model selection

Comparing, validating and choosing parameters and models.

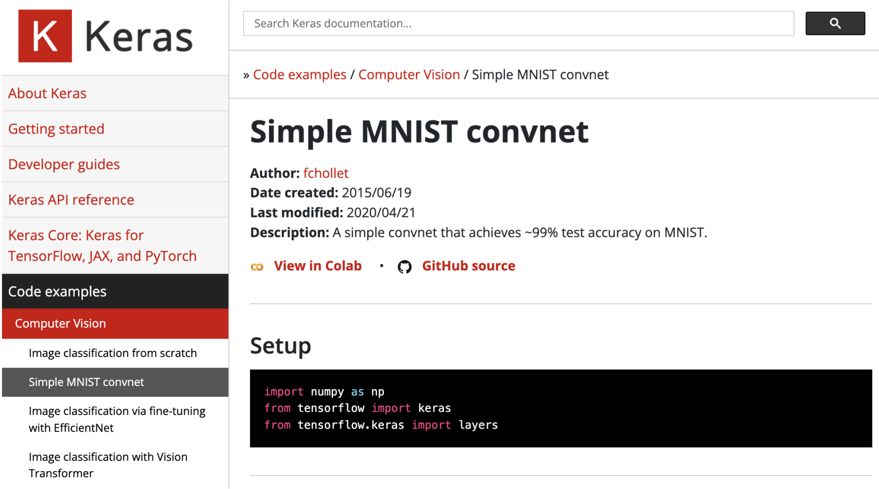
Goal: Improved accuracy via parameter tun-

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algo-

Keras



Keras
Search Keras documentation...
About Keras
Getting started
Developer guides
Keras API reference
Keras Core: Keras for TensorFlow, JAX, and PyTorch
Code examples
Computer Vision
Image classification from scratch
Simple MNIST convnet
Image classification via fine-tuning with EfficientNet
Image classification with Vision Transformer
» Code examples / Computer Vision / Simple MNIST convnet

Simple MNIST convnet

Author: fchollet
Date created: 2015/06/19
Last modified: 2020/04/21
Description: A simple convnet that achieves ~99% test accuracy on MNIST.

View in Colab • **GitHub source**

Setup

```
import numpy as np
from tensorflow import keras
from tensorflow.keras import layers
```

Prerequisites

- Probability theory and basic statistics (e.g. MATH 350 and MATH 450)
- Knowledge of algorithmic concepts. Comfortable programming in a high-level language
- Multivariable calculus (e.g. MATH 243)
- Linear algebra (e.g. MATH 349)

Evaluation

- Homework (theoretical + programming problems): 50%
- In-class quizzes and demos: 10%
- Final project: 40% (10% presentation, 30% final report)
- Grading system:

$\geq 94\%$	At least A
$\geq 90\%$	At least A-
$\geq 80\%$	At least B-
$\geq 70\%$	At least C-
$\geq 60\%$	At least D-
$< 60\%$	F

Platforms

- We will use Python during the course (there will be sessions to review the language). Specifically, we will use Google Colab for coding and programming assignments:

  https://colab.research.google.com

- We will use LaTeX to write the final report. The easiest way to use it collaboratively is to register an Overleaf account:

  https://www.overleaf.com

Homework policy

- Copying solutions in whole or in part from other students or **any other source** without acknowledgement constitutes cheating.
- Any student found cheating risks automatically failing the class and will be referred to the Office of Student Conduct
- You can discuss with other students, but must write up your own solutions/codes
- Please note your collaborators on your submissions

Final project

- Group project: 5-6 people (sign up on Canvas)
- The groups should be formed by the end of Week 4
- Data-oriented projects
  - Pick a practical learning problem with a dataset
  - Analyze the dataset
  - Write a report (in the form of a 4-page IEEE conference paper)
  - Present the project (last week of the semester)

Final project: some datasets

1. Fraud detection
2. Predict survival on the Titanic
3. Predict air pollution
4. Predict corporate credit rating
5. Predict next-day rain in Australia
6. Sign language MNIST
7. Job change prediction
8. House price prediction
9. Healthcare analytics
10. Predict water quality

Final project: scope

- your project should focus on only **one task**
- you will be graded based on:
  - how you apply the knowledge in the course to approach the problem
  - whether your experiment setups are reasonable to evaluate your methods
  - whether your conclusions are supported by your experiments
  - the clarity of your report.
- you are **not** graded based on
  - your model's accuracy
  - whether you can successfully solve the task

Final project: report

- in IEEE conference format
- maximum length: 4 pages + 1 additional page for the references
- should include:
  - An abstract (short paragraph summarizing your work)
  - An introduction (giving an overview of your work)
  - Related work (discussing briefly previous work on the problem)
  - Data and Methods (discussing the problem, the dataset, and your approaches to the problem, etc.)
  - Experiments (detailing your experiment setups to evaluate your methods, presenting and discussing the results of your experiments; include any tables or figures to show your results)
  - Discussions and conclusions (any discussions and conclusions that you can draw from your work)

Tentative schedule

- Introduction to (supervised) machine learning (4 weeks)
- Mathematical techniques in data science (8 weeks)
- Final project presentations (1 week)

Introduction to supervised learning (4 weeks)

- Week 1: Intros and reviews. Working with Python and sklearn.
- Week 2-3: Basic methods (Nearest neighbors, Logistic regression, LDA, SVM, Decision tree, Feed-forward neural nets). Formulations and demos.
- Week 4: Deep learning

Mathematical techniques in data science (8 weeks)

Meta-level techniques

- Week 5: Intro to statistical learning theory
- Week 6: SVM and the Kernel trick
- Week 7 and 8: Model selection and regularization
- Week 9: Boosting, bagging, bootstrapping

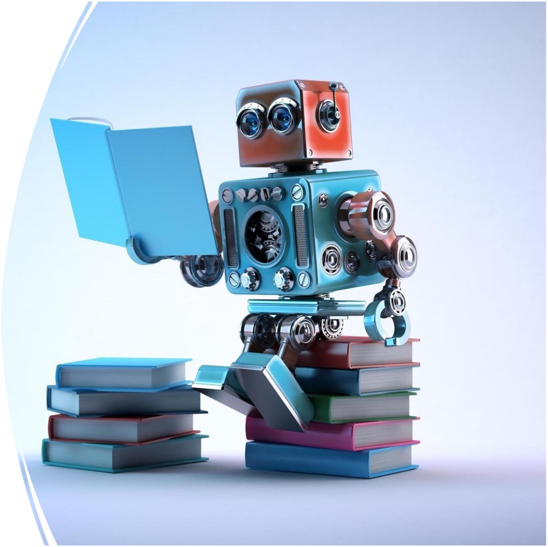
Other learning contexts

- Week 10: PCA and Manifold learning
- Week 11: Clustering
- Week 12: Selected topics

Questions?

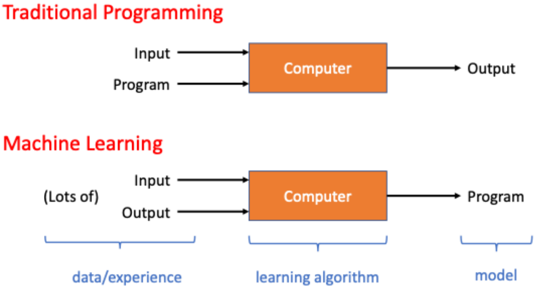
An introduction to machine learning

What is Machine Learning?

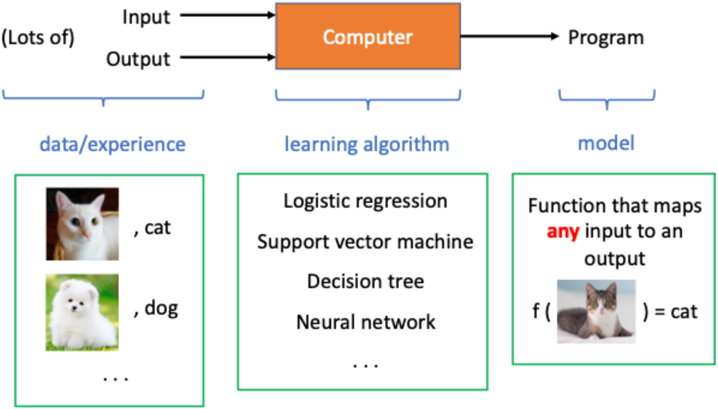


What is Machine Learning?

- A field that studies "algorithms that allow computer programs to automatically improve through experience." Tom Mitchell (1997)



Machine learning components



(Lots of) Input Output

Computer

Program

data/experience

learning algorithm

model

, cat

, dog

. . .

Logistic regression

Support vector machine

Decision tree

Neural network

. . .

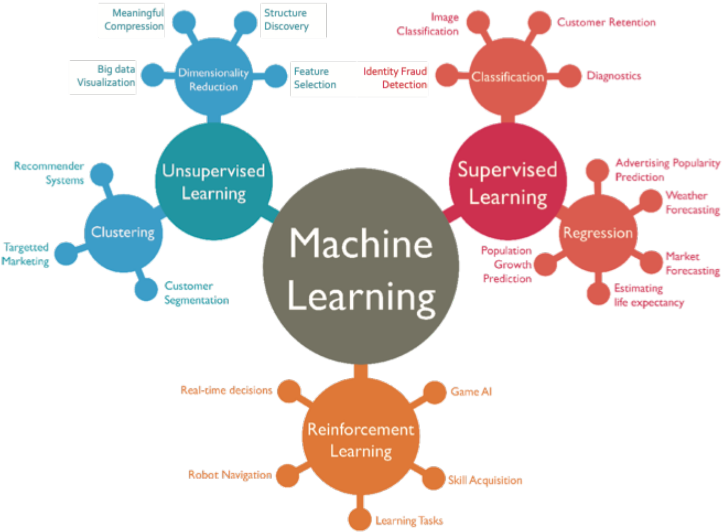
Function that maps **any** input to an output

f ( ) = cat

ML Paradigms

- Supervised learning
  Learn a function that maps an input to an output
  (input, output) pairs are given as examples.
- Unsupervised learning
  Learn patterns from inputs only (no outputs).
- Reinforcement learning
  Learn to take actions to maximize some reward

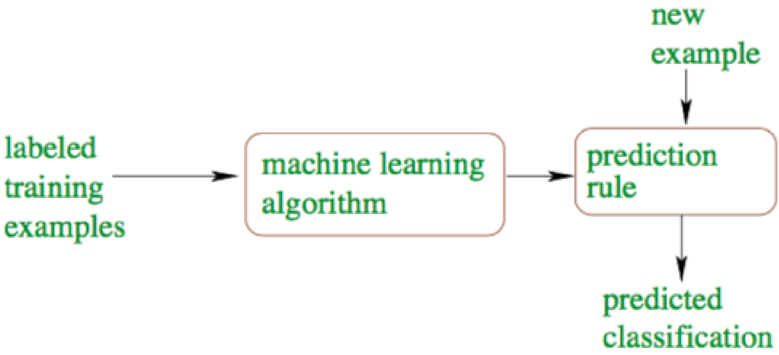
ML Paradigms



(Source: Abdul Rahid)

Supervised learning

In the first 4 weeks, we will focus on supervised learning



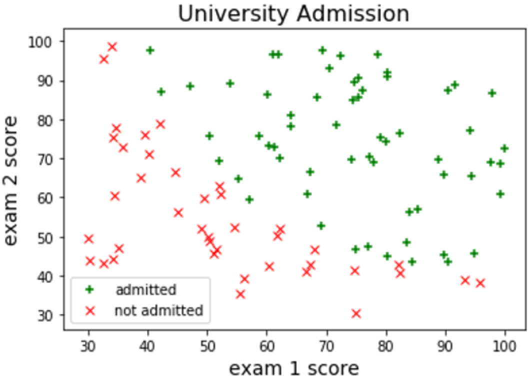
Supervised learning: learning a function that maps an input to an output based on example input-output pairs

Supervised learning

- One example contains both input (X) and output (Y)
- Two most common tasks:
  - Classification: discrete output Y
  - Regression: continuous output Y

Classification or regression?

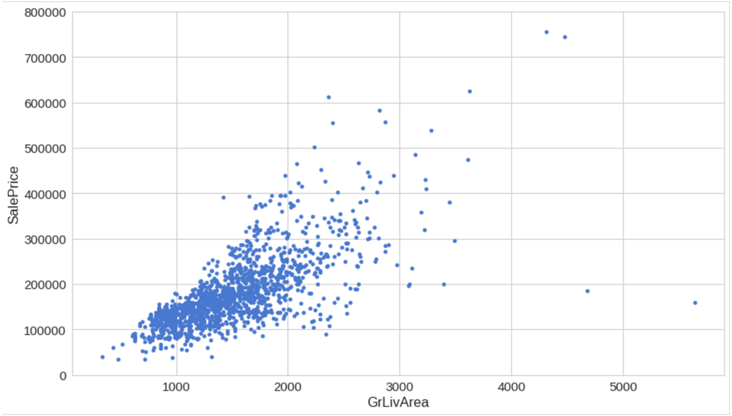
Example: Predict university admission based on exam scores



→ Two classes: admitted/not admitted → Binary classification

Classification or regression?

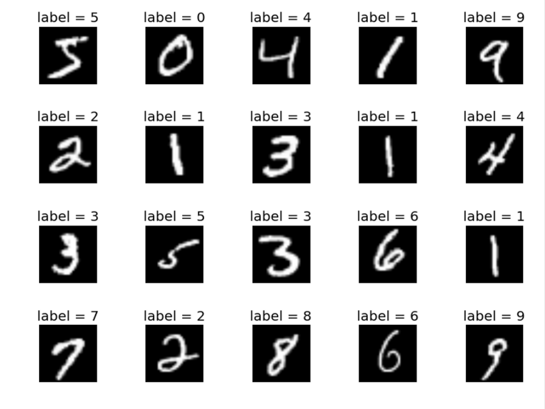
Example: Predict house price by living area



→ Regression

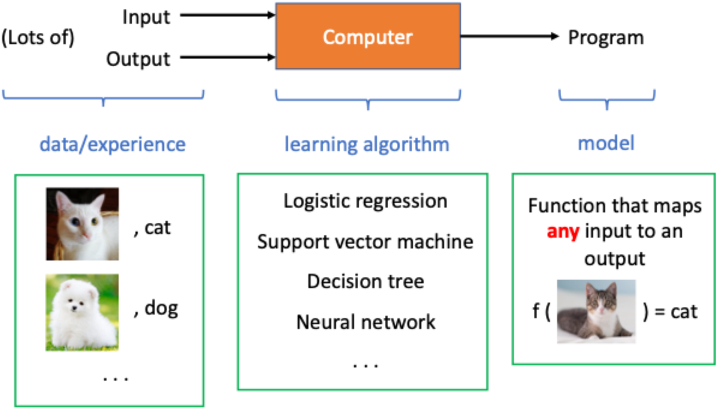
Classification or regression?

Example: Handwritten digit recognition



→ Multiple classes (labels) → Multi-class classification

Machine learning components



(Lots of) Input Output → Computer → Program

data/experience

learning algorithm

model

, cat

, dog

. . .

Logistic regression

Support vector machine

Decision tree

Neural network

. . .

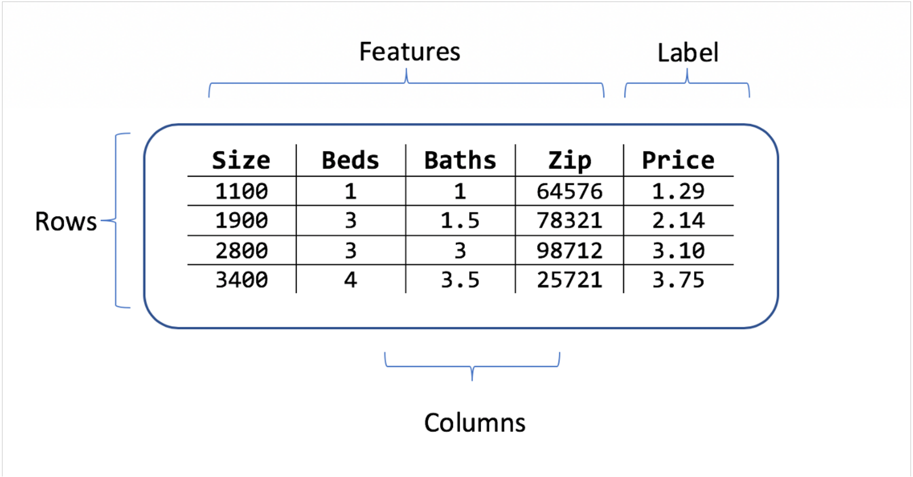
Function that maps **any** input to an output

f ( ) = cat

Supervised learning: data

- In principle, data can be in any form
- Raw and complex data are hard to use → may need to manually (by humans) extract features before usage
- You may also need to pre-process certain features
    - Handle missing values
    - Normalize data

Sample data

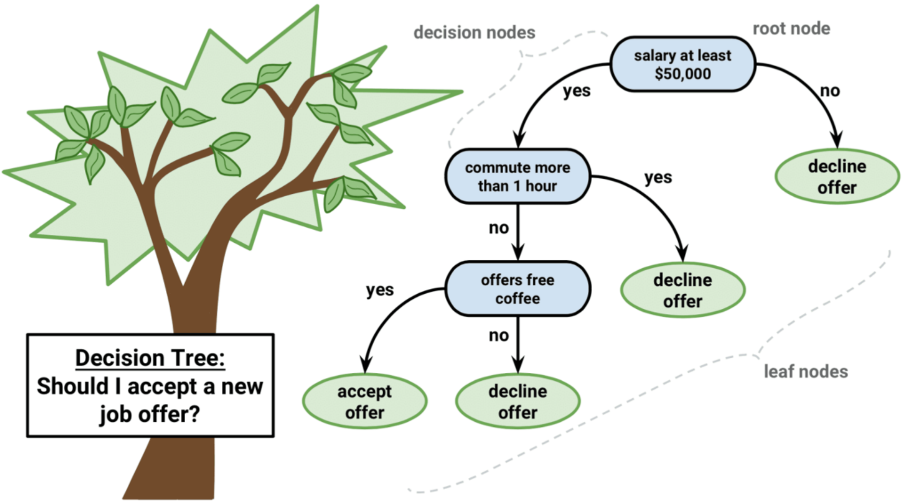

Features
Label
Rows
Columns

Size	Beds	Baths	Zip	Price
1100	1	1	64576	1.29
1900	3	1.5	78321	2.14
2800	3	3	98712	3.10
3400	4	3.5	25721	3.75

(Source: Microsoft)

Model

Function that takes a pre-processed feature vector and predicts its label



(Source: packtpub.com)

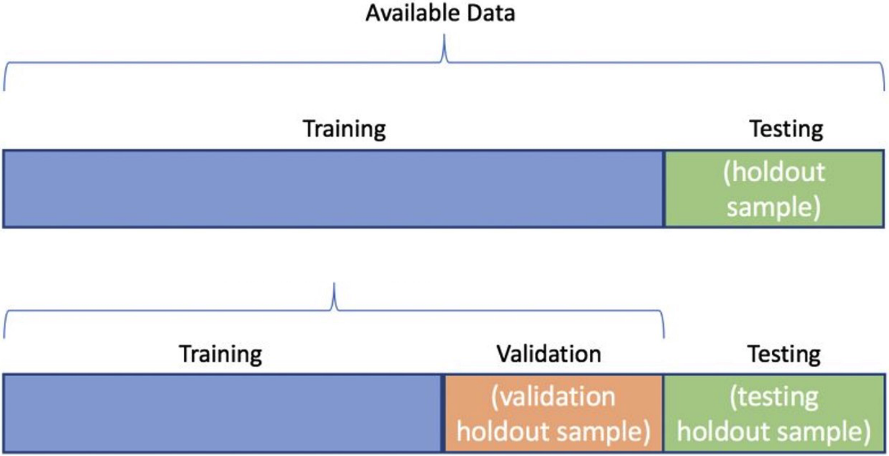
Learning algorithm

- An algorithm that returns the **parameters** or **configurations** of the model from data
- Note: learning algorithm = training algorithm
- May need to optimize an objective or loss function

Evaluate a learned model

- How effective the model makes predictions on new (unseen) data
- Classification: accuracy or error rate
- Regression: average (squared) distance between predicted and true values (mean squared error)

Data splitting practices


Available Data
Training
Testing
(holdout sample)
Training
Validation
Testing
(validation holdout sample)
(testing holdout sample)