Machine learning

Introduction

Mohammad-Reza A. Dehaqani

dehaqani@ut.ac.ir

Prerequisites

- Probability
  - Random Variables, Expectations, Distributions
- Statistical inference
  - Estimation Theory
  - Hypothesis Testing
- Linear Algebra & Multivariate/Matrix Calculus
  - Eigenvalue/vector
- Optimization
  - Gradient descent and Newton–Raphson algorithm
  - Lagrange multiplier
- Basic computer science principles
  - Complexity of algorithm
  - Comfortably write non-trivial code in Python/numpy

Meet the team

- Instructor:
  - Babak Nadjar Aarabi
  - Mohammadreza A. Dehaqani

- Head TAs:
  - Zahra Ebrahimian
  - Alireza hosseini

Hands On

- Active participation rather than theory
- Providing direct practical experience of course materials on real-world data

Homework (~7 score)

- Approximately **5 homework** assignments
- Homework includes:
  - **Analytical questions**
  - **Computing exercises** (using Python)
- Each homework assignment can have up to **10 extra points**
- **Delay Penalty**:
  - **Hard Delay Policy**: 5% penalty per day
  - **Soft Delay Policy**: Up to 50%
- You can use just Python(Anaconda Jupyter Notebook)

Collaboration

- You can discuss homework problems with your classmates, but you must submit your own individual homework write-up, **in your own words and using your own code for the programming exercises.** Please indicate at the top of your write-up the names of the students with whom you worked

Final Project (~4 score)

- Give you a chance to exercise what you learned in the course in some real-world problem and data.
- Students get involved in
  - Data Gathering
  - Problem Solving
  - Implementation
  - Documentation
- Competition on optional project (max 1 score)

Deadlines

- All deadlines Would be handled by your chief TA
  - Z.ebrahimian@ut.ac.ir
  - arhosseini77@ut.ac.ir

Exams

- Midterm (~4 score)
- Final (~5 score)

Course Preview

Lecture 6: Neural Network

- Neural networks (basic concepts)
- Neural networks (Learning)
- Convolutional Neural Networks

Lecture 7: Tree-Based Methods

- Decision Tree
- Ensemble learning
- Bagging and Boosting

Lecture 8: Support Vector Machines:

- SVM (hard, soft)
- SVM (dual problem)

Lecture 9: Unsupervised Learning

- Clustering:
- Dimensionality Reduction Methods
    - PCA
    - LDA

Course Preview

Lecture 10: Model and Feature Selection

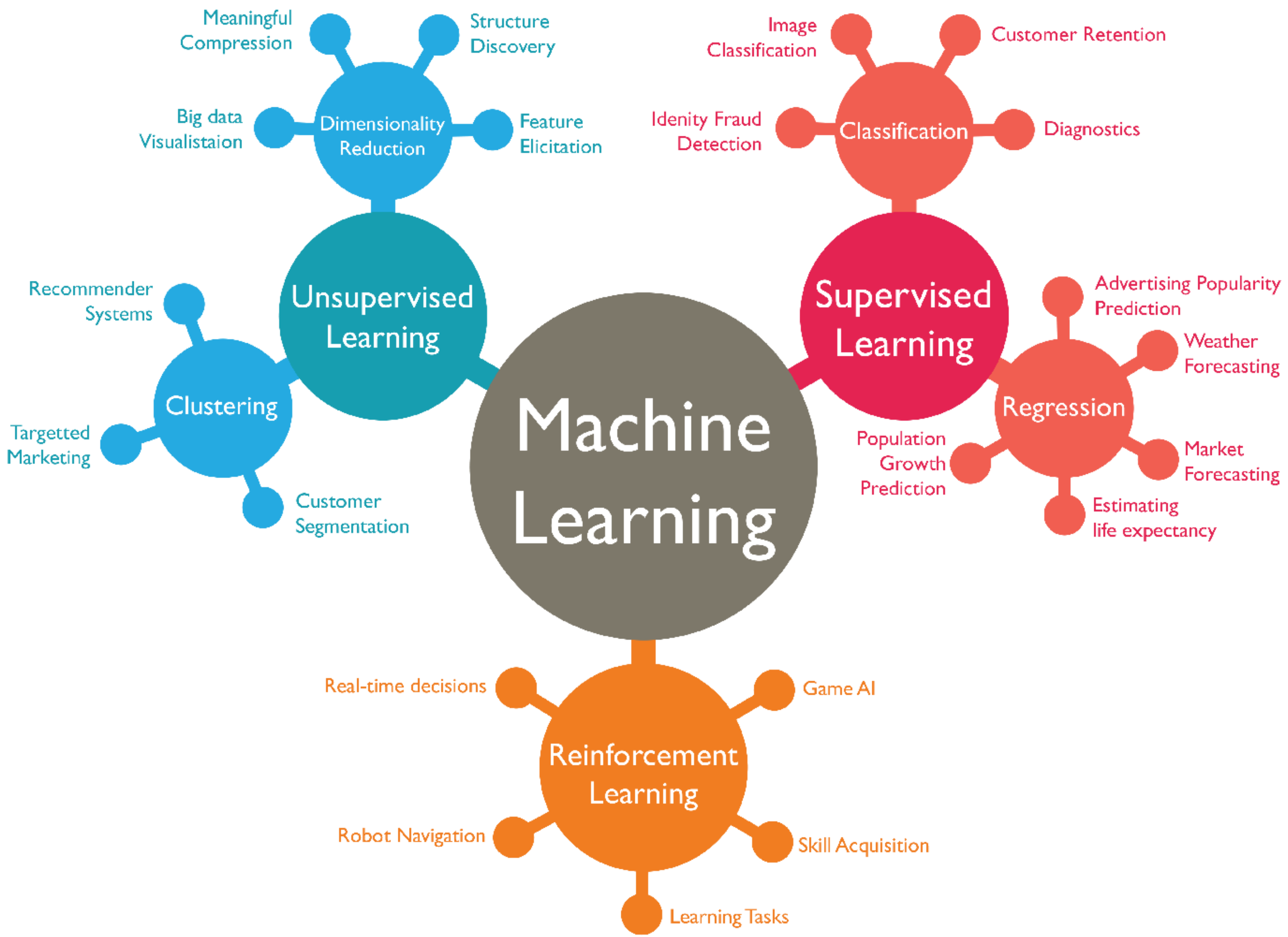
Lecture 11: Reinforcement Learning

- MDPs & Value Functions
- Value & Policy Iteration
- Q-learning & Deep RL

Lecture 12: weakly supervised Learning (extra session)

- Self-Supervised Learning
- Semi-Supervised Learning
- Active Learning
- Contrastive learning

Data driven machine learning


Meaningful Compression
Structure Discovery
Big data Visualistaion
Dimensionality Reduction
Feature Elicitation
Image Classification
Customer Retention
Idenity Fraud Detection
Classification
Diagnostics
Recommender Systems
Unsupervised Learning
Supervised Learning
Advertising Popularity Prediction
Weather Forecasting
Clustering
Regression
Targetted Marketing
Machine Learning
Population Growth Prediction
Market Forecasting
Customer Segmentation
Estimating life expectancy
Real-time decisions
Game AI
Reinforcement Learning
Robot Navigation
Skill Acquisition
Learning Tasks

Machine Learning **Algorithms and Principles**

- **Classification**: Naïve Bayes, Logistic Regression, Neural Networks,
- **Support Vector Machines**, k-NN, Decision Trees, Boosting
- **Regression**: Linear regression, Kernel regression, Nonparametric regression
- **Unsupervised methods**: Kernel density estimation, k-means and hierarchical clustering, PCA
- **Core concepts**: Probability, Optimization, Theory, Model selection, overfitting, bias-variance tradeoffs

Recommended textbooks (not required)

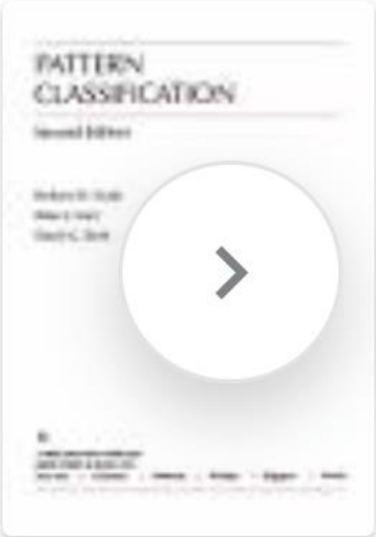
Pattern Classification

Book by David G. Stork, Peter E. Hart, and Richard O. Duda

Book preview

71/679 pages available

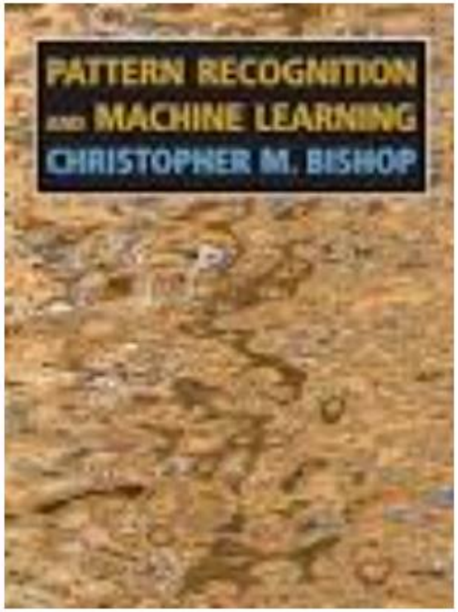
PREVIEW



Recommended textbooks (not required)

Pattern Recognition and Machine Learning

Book by Christopher Bishop



Recommended textbooks (not required)

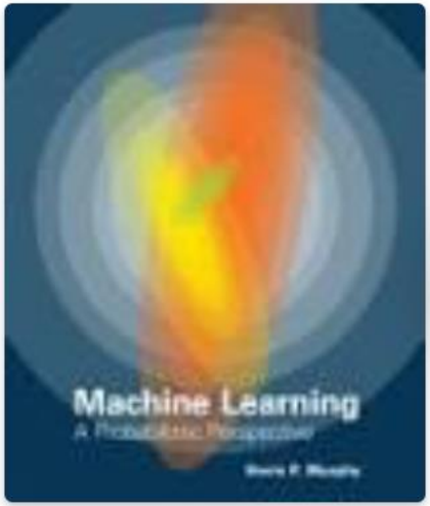
Machine Learning: A Probabilistic Perspective

Book by Kevin P. Murphy

Book preview

113/1102 pages available

PREVIEW



Recommended textbooks (not required)

Machine Learning: A multistrategy approach

Book by Tom M. Mitchell

Recommended textbooks (not required)

The Elements of Statistical Learning

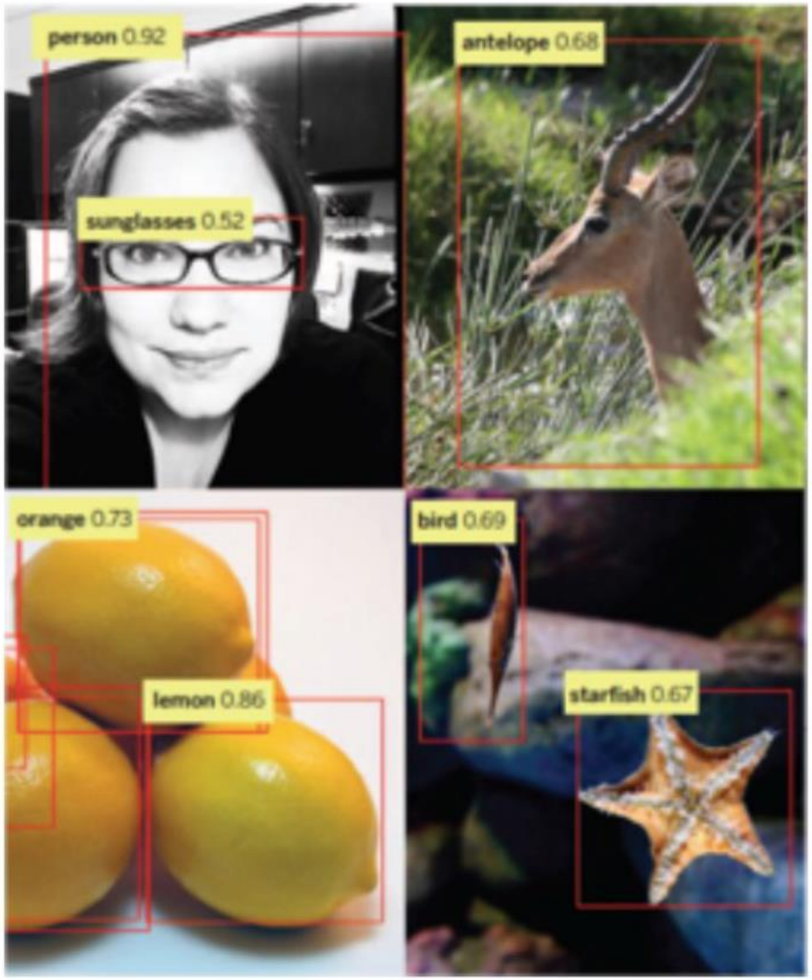
Book by Jerome H. Friedman, Robert Tibshirani, and Trevor Hastie

Book preview

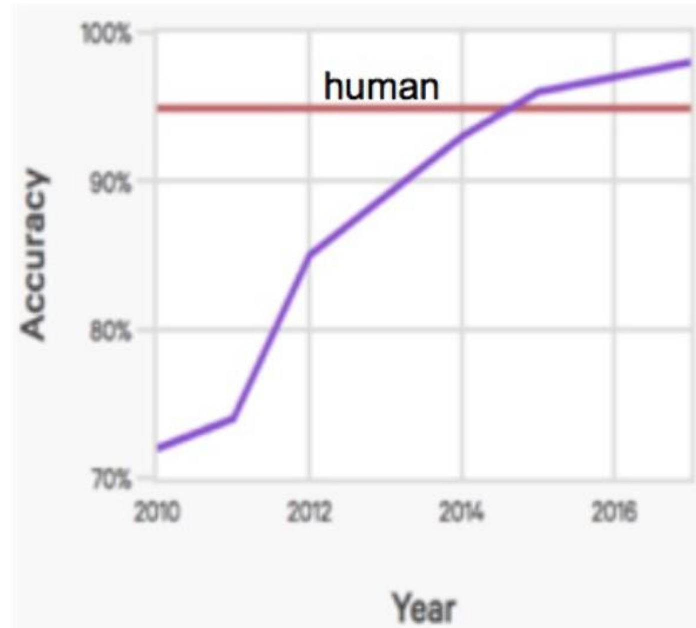
60/560 pages available

PREVIEW

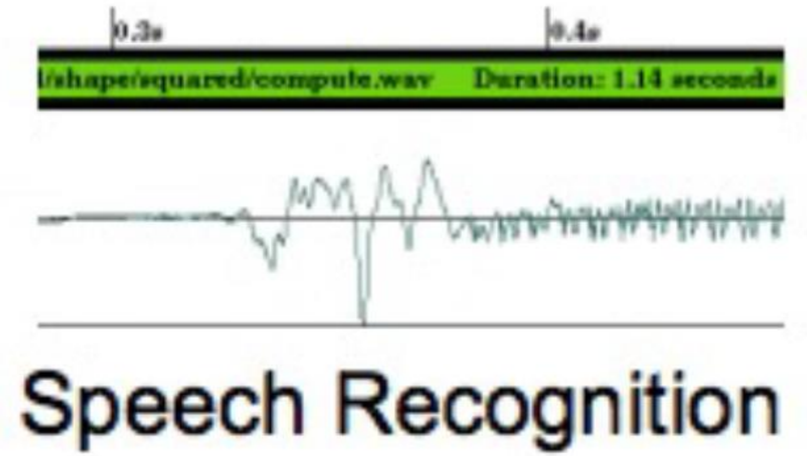
Machine Learning in Action

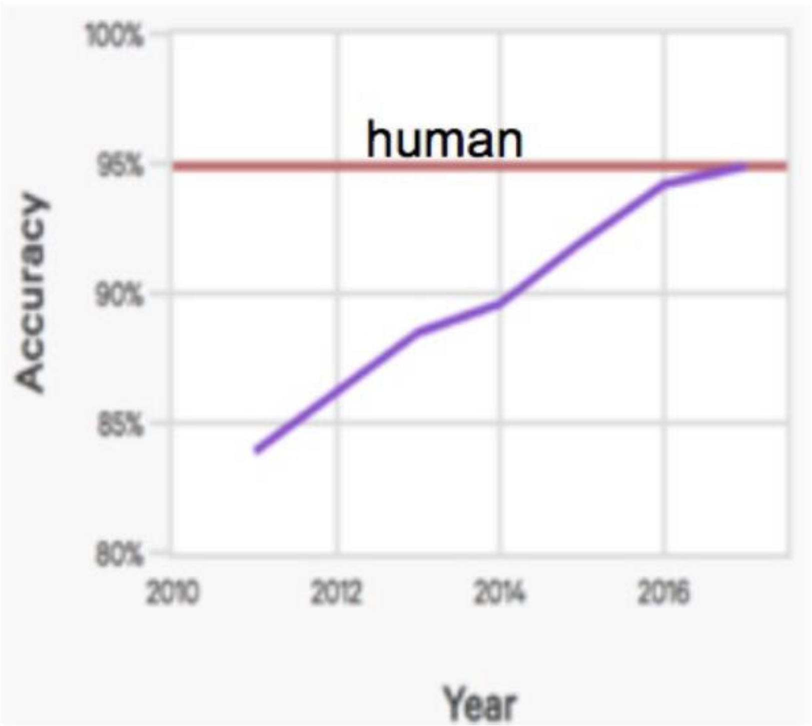


Computer vision

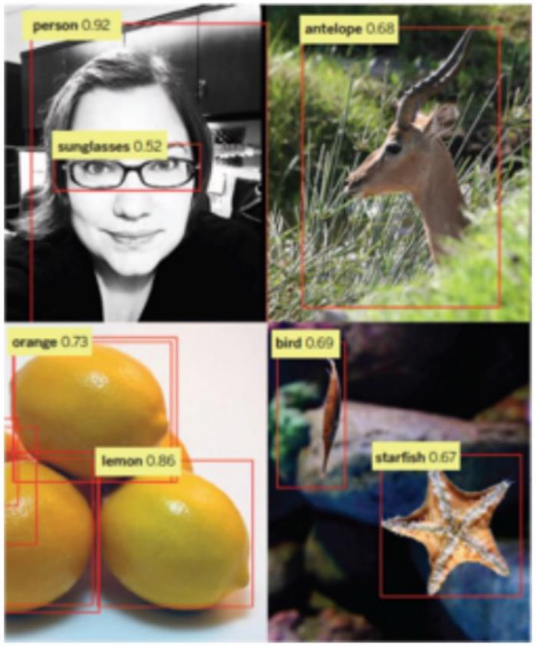


Machine Learning in Action

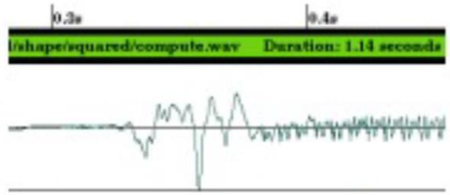




Machine Learning in Action



Computer vision

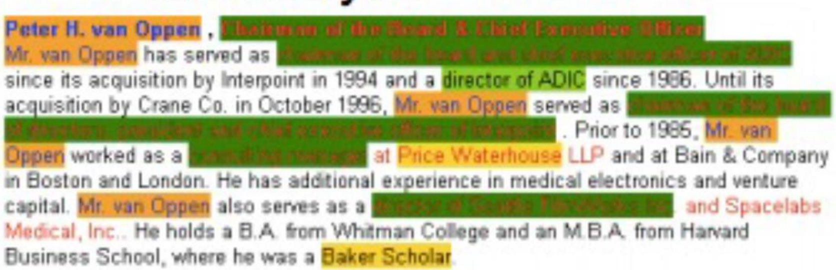


Speech Recognition

Robotic control

Games & Reasoning

Text analysis

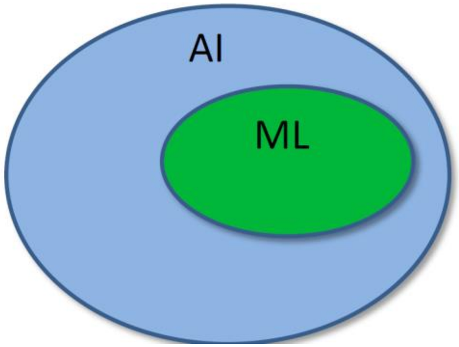

Peter H. van Oppen , Chairman of the Board & Chief Executive Officer
Mr. van Oppen has served as [illegible]
since its acquisition by Interpoint in 1994 and a director of ADIC since 1986. Until its
acquisition by Crane Co. in October 1996, Mr. van Oppen served as [illegible]
. Prior to 1985, Mr. van
Oppen worked as a [illegible] at Price Waterhouse LLP and at Bain & Company
in Boston and London. He has additional experience in medical electronics and venture
capital. Mr. van Oppen also serves as a [illegible] and Spacelabs
Medical, Inc.. He holds a B.A. from Whitman College and an M.B.A. from Harvard
Business School, where he was a Baker Scholar.

Protein folding

ML is ubiquitous

- Wide applicability
- Software too complex to write by hand
- Improved machine learning algorithms
- Improved data capture, networking, faster computers
- Demand for self-customization to user, environment

- **AI: develop intelligent agents**
- **ML: learn to generalize using data**



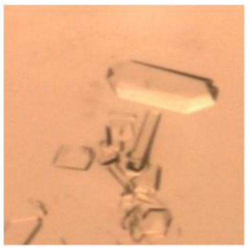
What is Machine Learning?

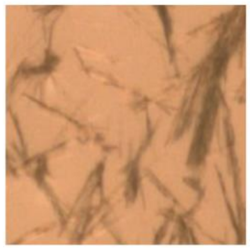
- Design and Analysis of algorithms that
  - improve their performance
  - at some task
  - with experience

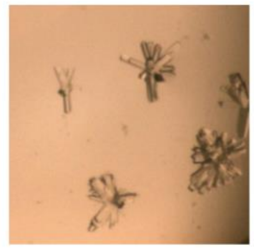
(experience) **(performance on task)**

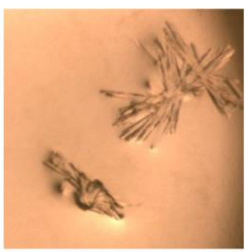
Example

- **Task:** Learning stage of protein crystallization

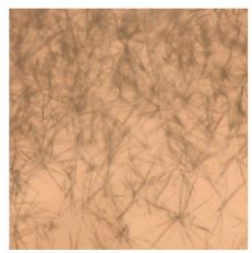

Crystal


Needle


Tree

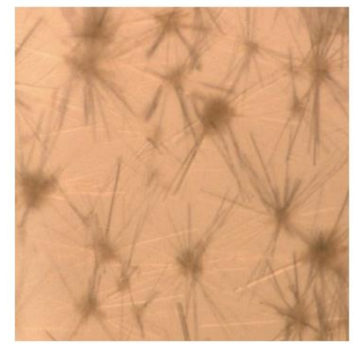

Tree


Empty


Needle

Experience

Predict the label of the test image?


?

Performance

Machine Learning Tasks

- Broad categories -
- **Supervised learning**
  - Classification, Regression
- **Unsupervised learning**
  - Density estimation, Clustering, Dimensionality reduction
- **Graphical models:** a probabilistic model for which a graph expresses the conditional dependence structure between random variables
- **Semi-supervised learning:** combination of a small amount of human-labeled data followed by a large amount of unlabeled data
- **Active learning:** a learning algorithm can interactively query a human user (or some other information source), to label new data points
- **Bayesian optimization**
- **Reinforcement learning**
- **Many more** …

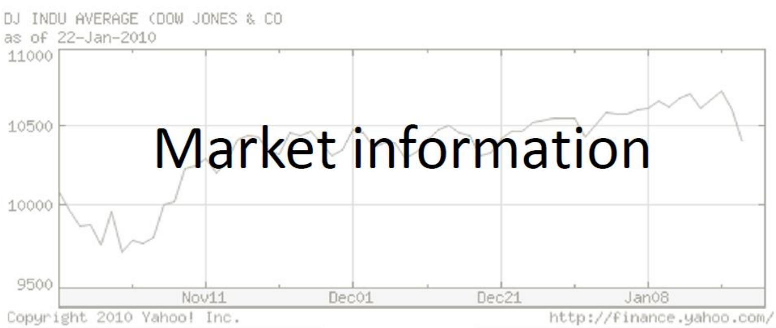
Supervised Learning

Input $X \in \mathcal{X}$

Label $Y \in \mathcal{Y}$

“Sports”
“News”
“Science”
…

Discrete Labels
Classification



Share Price
“$ 24.50”

Continuous Labels
Regression

Task: Given $X \in \mathcal{X}$, predict $Y \in \mathcal{Y}$.

$\equiv$ Construct **prediction rule** $f : \mathcal{X} \to \mathcal{Y}$

Unsupervised Learning

Aka "learning without a teacher"

Input $X \in \mathcal{X}$

Document/Article

Word distribution
(Probability of a word)

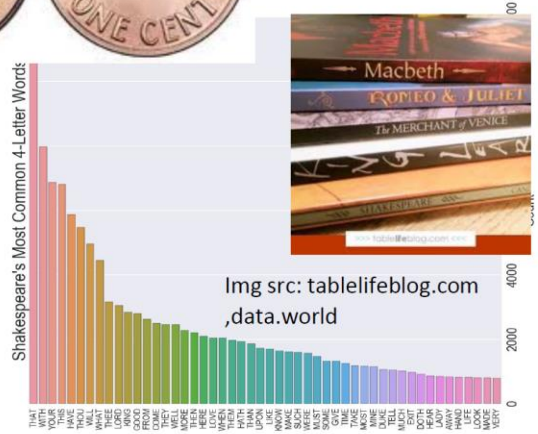
Task: Given $X \in \mathcal{X}$, learn $f(X)$.

Unsupervised Learning

Learning a Distribution

Bias of a coin

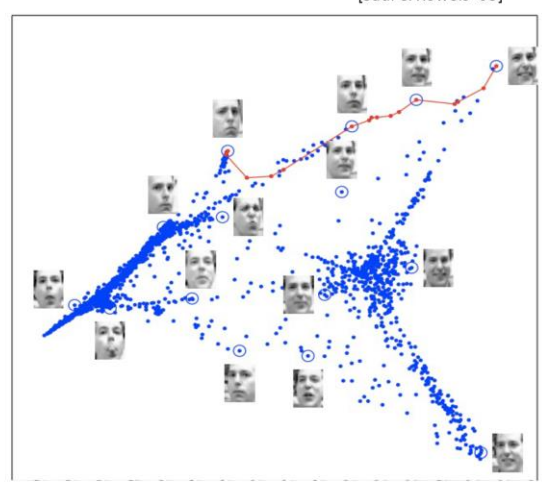
Distribution of words in text



Img src: tablelifeblog.com ,data.world

Clustering

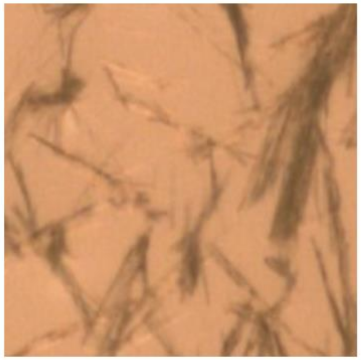
[Saul & Roweis '03]



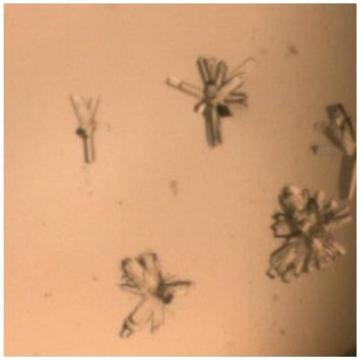
Dimensionality Reduction/Embedding

Experience = Training Data

Crystal



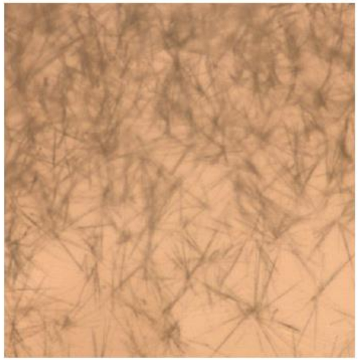
Needle



Tree

Tree

Empty

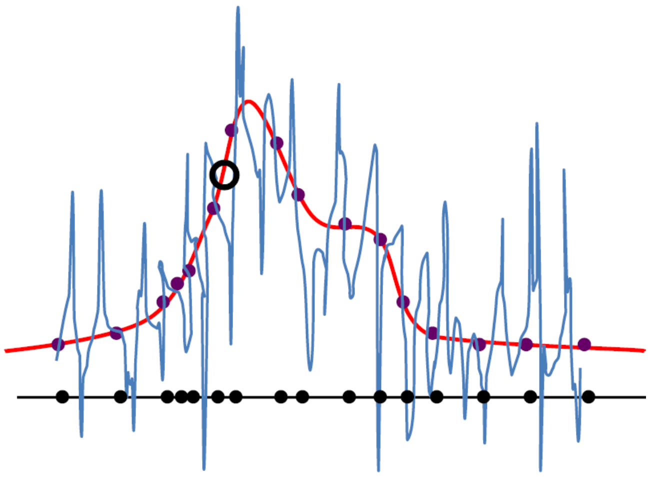


Needle

Experience

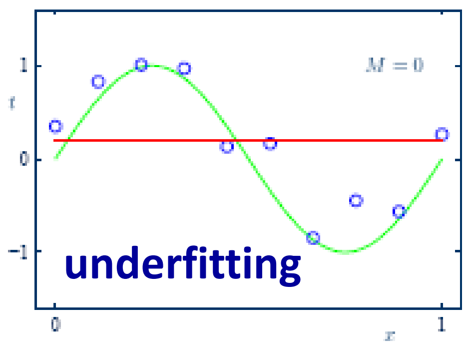
Training Data vs. Test Data

- **Generalization & Overfitting**
  - A good ML algorithm:
    - **should generalize** aka perform well on **test** data
    - should not: **overfit** the **training** data

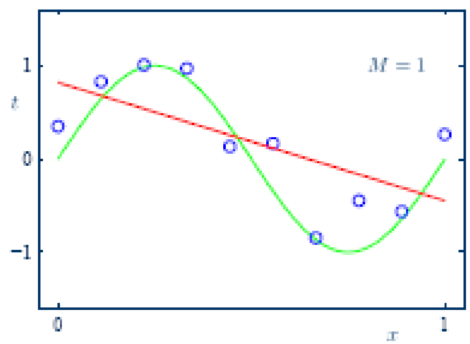


    - Critical to report test and NOT training data accuracy

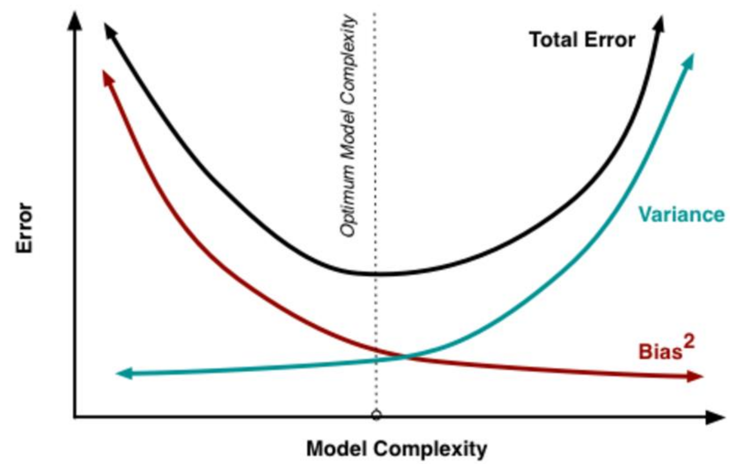
Understanding model complexity (theory of learning): The bias–variance tradeoff

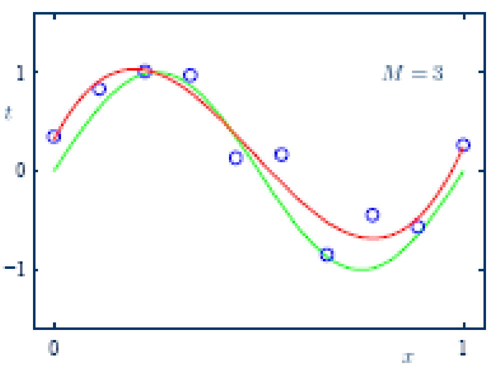


(a) 0'th order polynomial

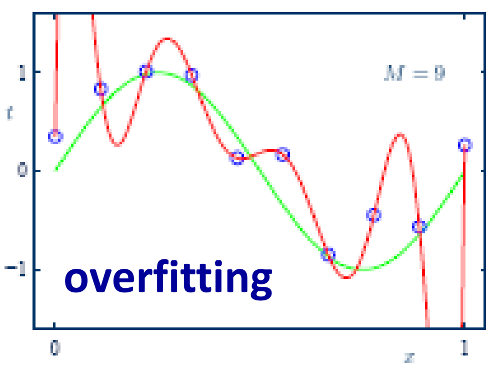


(b) 1'st order polynomial





(c) 3'rd order polynomial



(d) 9'th order polynomial

bias–variance tradeoff is the **property** of a set of **predictive models** whereby models with a lower bias in **parameter estimation** have a higher variance of the parameter estimates across samples, and vice versa]

Performance Measure

- $\text{loss}(Y, f(X))$ asure of **closeness** between label *Y* and prediction *f*(*X*) for test data X

Binary Classification

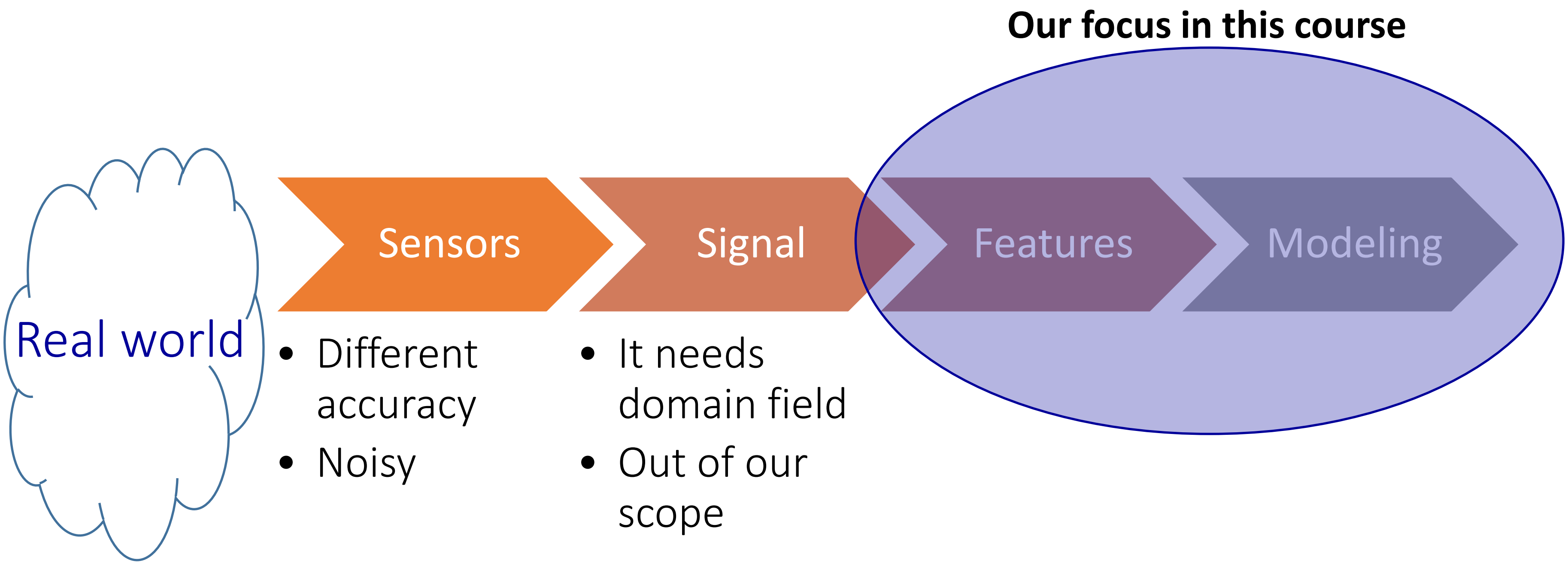
$$\boxed{\text{loss}(Y, f(X)) = 1_{\{f(X) \neq Y\}} \quad \textbf{0/1 loss}}$$

Regression

$$\boxed{\text{loss}(Y, f(X)) = (f(X) - Y)^2 \quad \textit{squared loss}}$$

- We will talk about more performance measures including for unsupervised learning later in course

Signal vs. feature



Core material

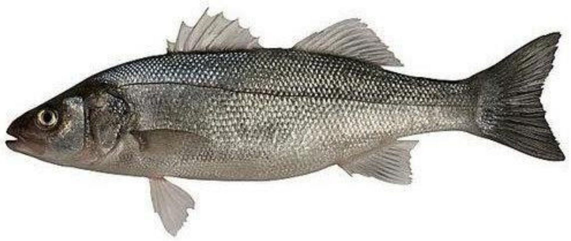
- **Finding patterns** in data; using them to make predictions.
- **Models** and statistics help us understand patterns.
- **Optimization** algorithms "learn" the patterns.
- The most important part of this is the **data**. Data drives everything else.
    - You cannot learn much if you don't have enough data.
- **Machine learning** has changed a lot in the last decade because the internet has made truly vast quantities of **data available**.

An Example

- “Sorting incoming fish on a conveyor according to species using **optical sensing**”

Species

Sea bass



Salmon

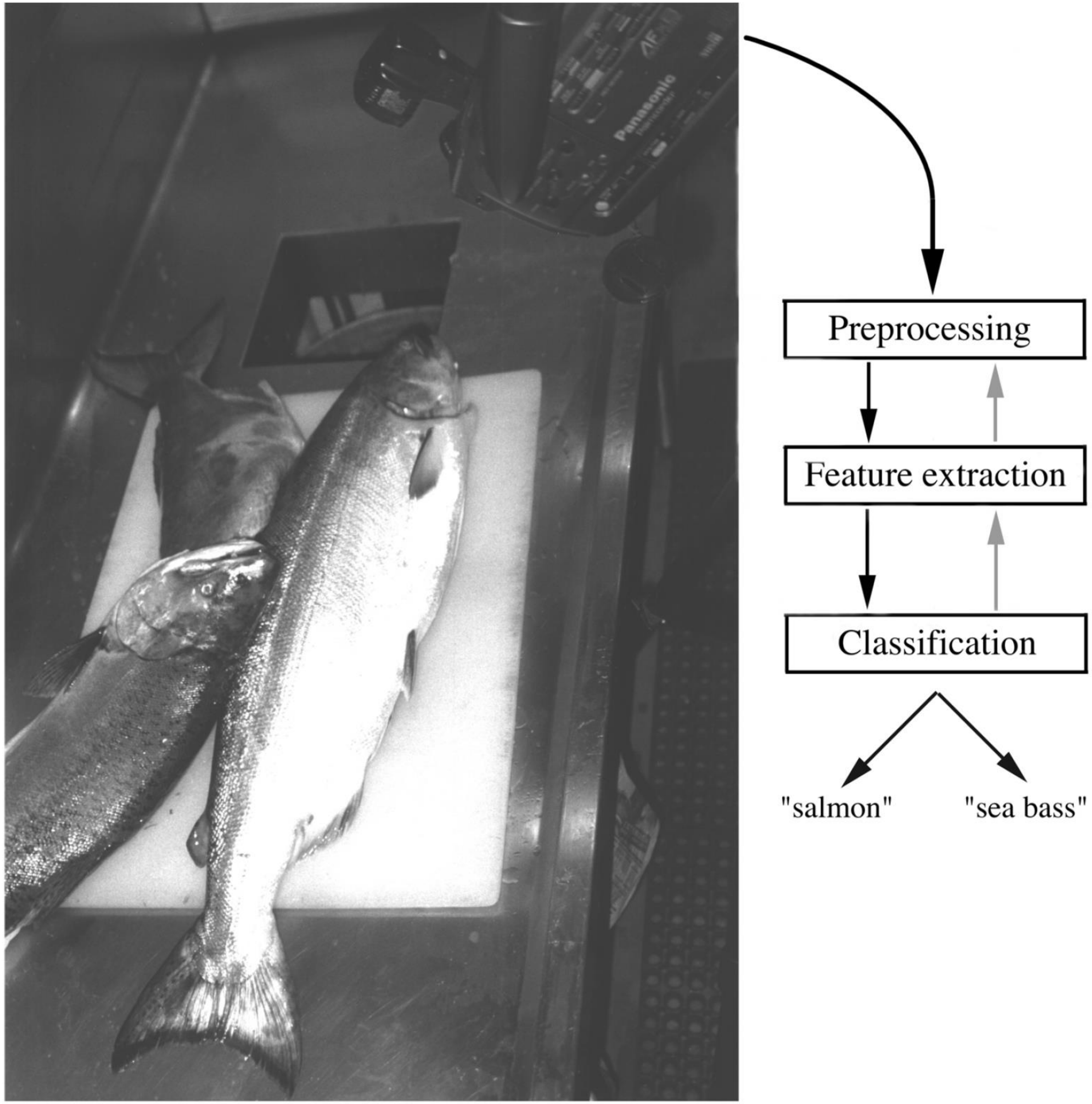
Problem Analysis

- Set up a camera **(sensors)** and take some sample images to extract features
  - Length
  - Lightness
  - Width
  - Number and shape of fins
  - Position of the mouth, etc…

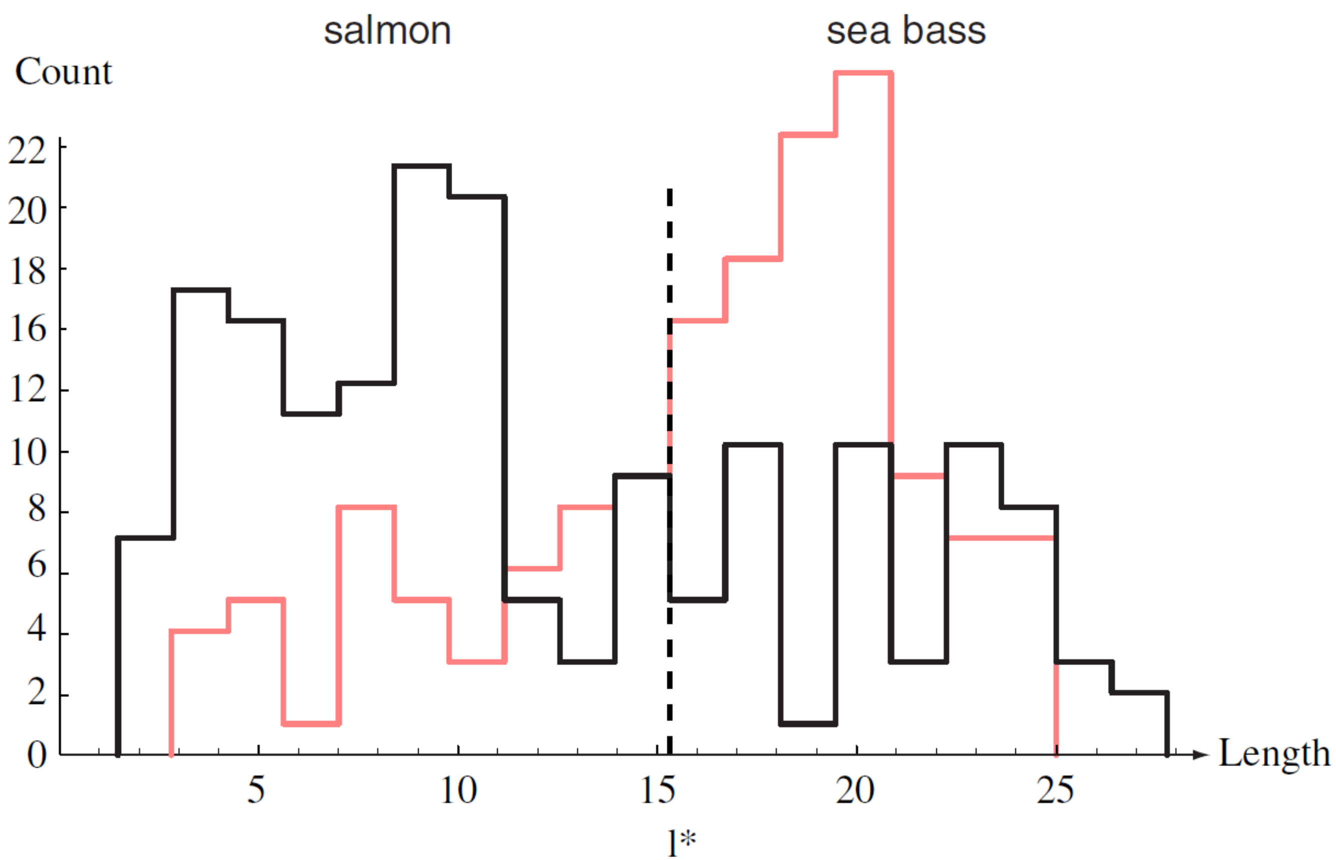
- This is the set of all suggested **features** to explore for use in our classifier!

Preprocessing to obtain features

- Use a **segmentation** operation to isolate fishes from one another and from the background
- Information from a single fish is sent to a feature extractor whose purpose is to **reduce** the data by measuring certain features
- The features are passed to a **classifier**

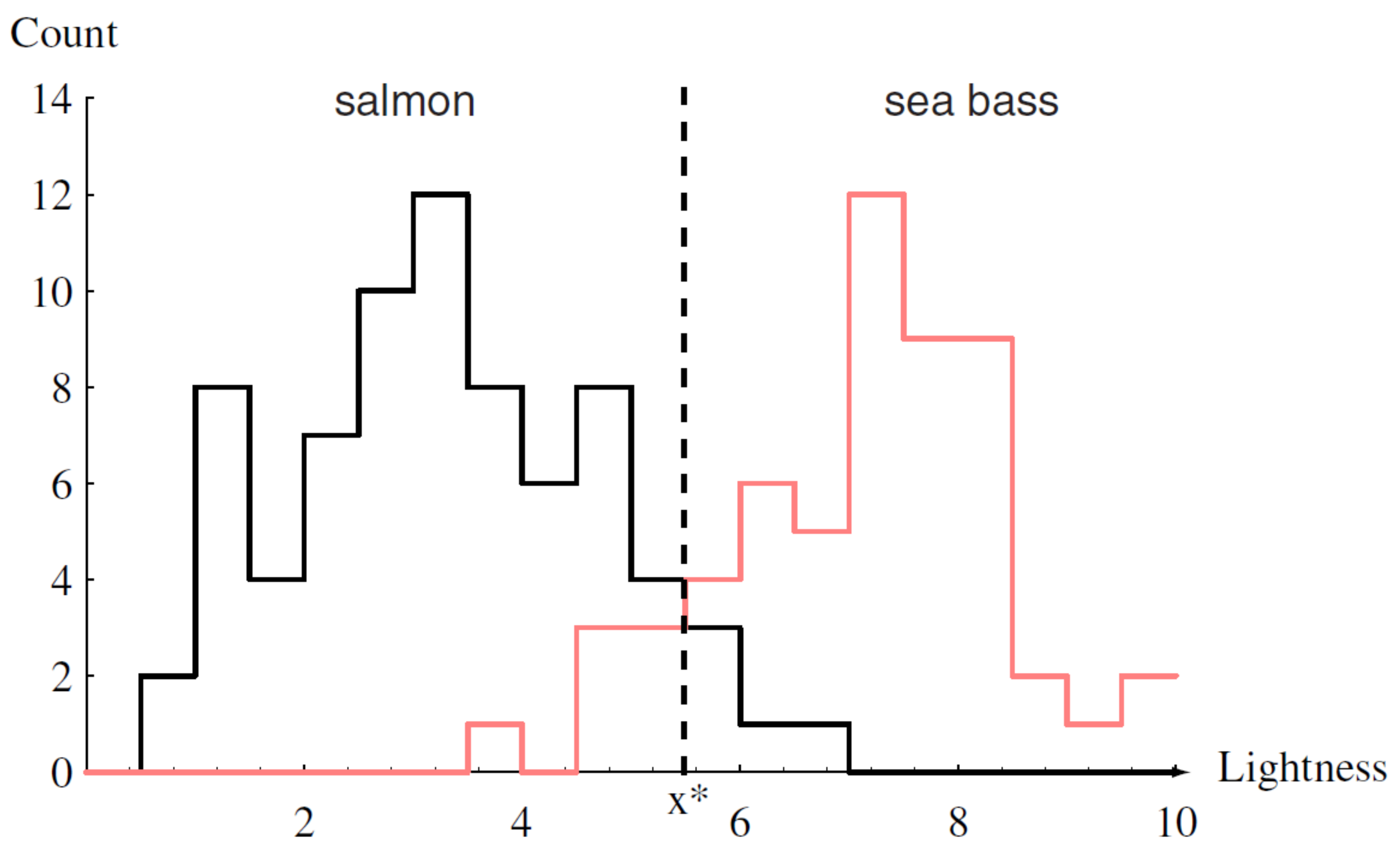

Preprocessing
Feature extraction
Classification
"salmon"
"sea bass"

Length feature for discrimination



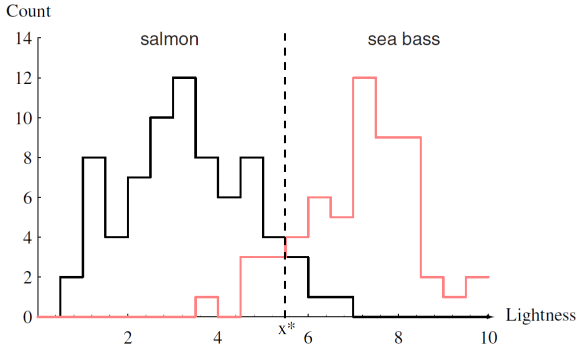
No single **threshold value l*** (**decision boundary**) will serve to unambiguously discriminate between the two categories; **using length alone**, we will have some errors. The value l* marked will lead to the **smallest number of errors**, on average.

Select the lightness as a possible feature



Decision theory;
Threshold decision boundary and cost relationship

Move our **decision boundary** toward smaller values of lightness in order to minimize the **cost** (reduce the number of sea bass that are classified salmon!)



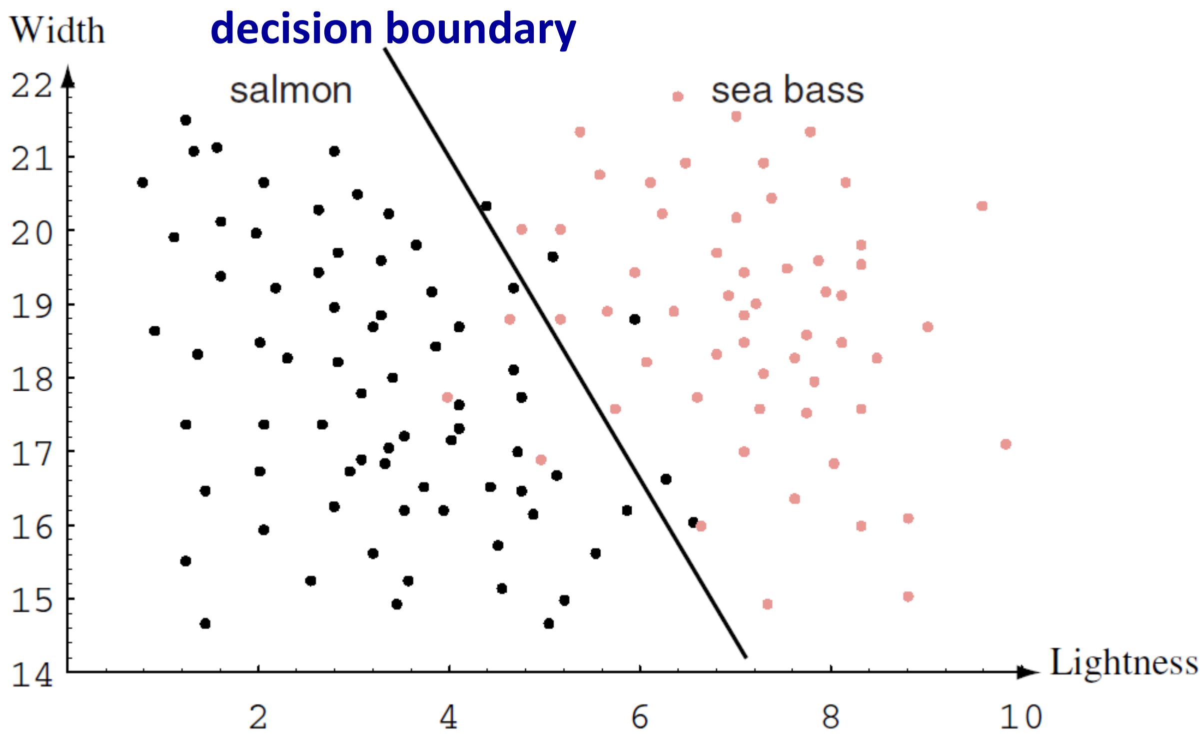
Task of decision theory

Multiple Features

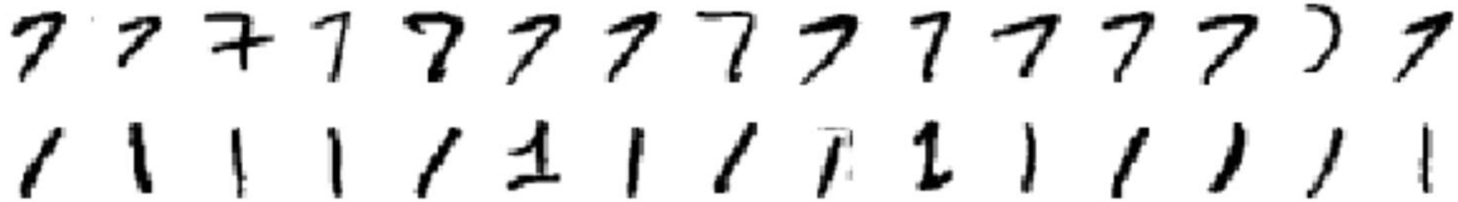
Adopt the lightness and add the width of the fish

Fish $\longrightarrow$ $x^T = [x_1, x_2]$

Lightness **Width**



Representation of digits



$$\begin{array}{|c|c|c|c|}\hline 3 & 3 & 3 & 3 \\ \hline 0 & 0 & 2 & 3 \\ \hline 0 & 0 & 1 & 3 \\ \hline 3 & 3 & 3 & 3 \\ \hline \end{array} \rightarrow \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 0 \\ 0 \\ 2 \\ 3 \\ 0 \\ 0 \\ 1 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}$$

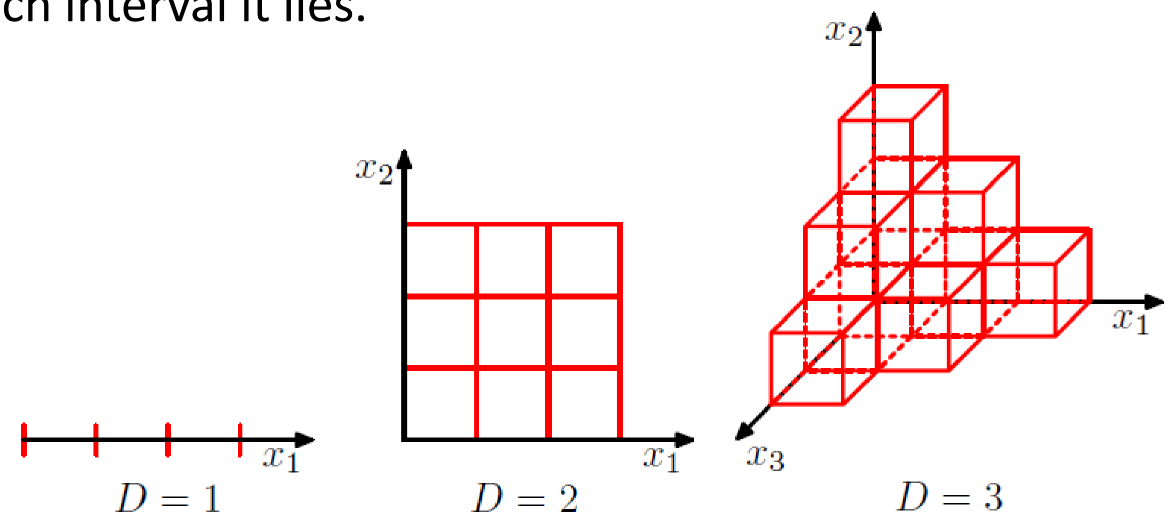
Images are points in **16-dimensional space**. Linear decision boundary is a **hyperplane**

How Many Features?

- Does adding more features always improve performance?
  - It might be difficult and computationally expensive to extract certain features.
  - Correlated features might not improve performance (i.e. redundancy).
  - “Curse” of dimensionality.

Curse of Dimensionality

- Adding too many features can, **paradoxically**, lead to a **worsening** of performance.
  - Divide each of the input features into a number of intervals, so that the value of a feature can be specified approximately by saying in which interval it lies.

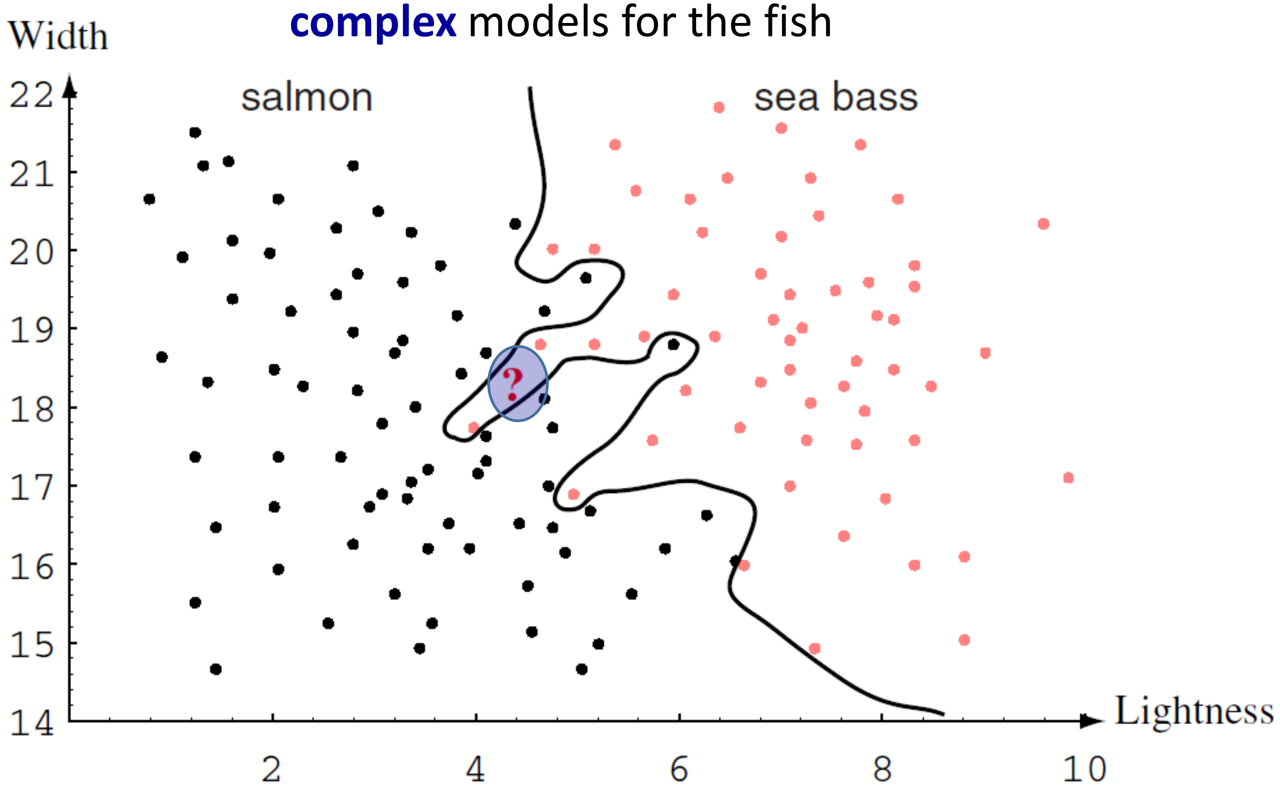


  - If each input feature is divided into **M** divisions, then the total number of cells is $\mathbf{M}^{d}$ (***d:*** # of features).
  - Since each cell must contain at least one point, the number of needed data grows exponentially with ***d***.

Issue of generalization

This **decision boundary** may lead to perfect classification of our **training samples**, it would lead to poor performance on future patterns (overfitting).

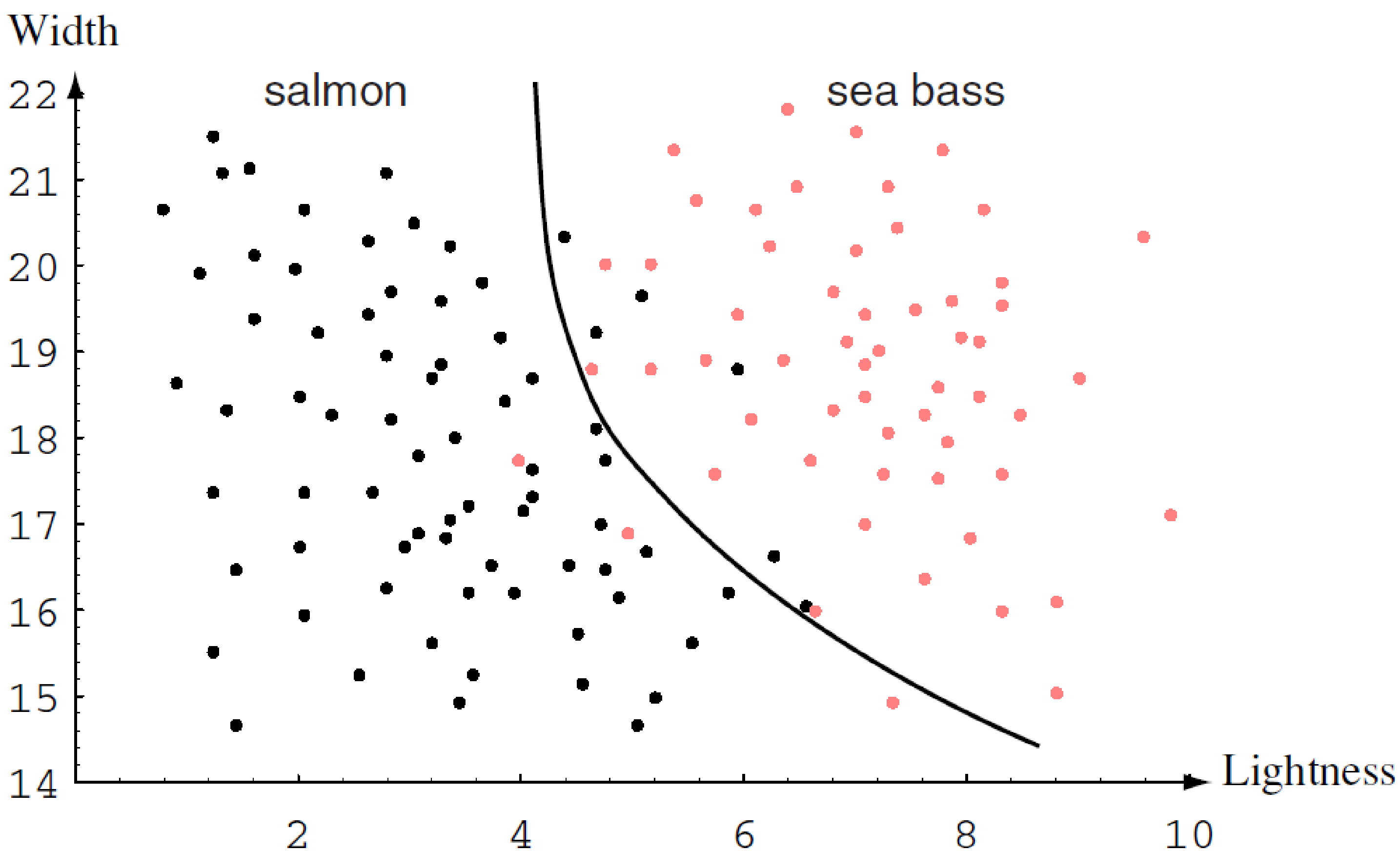
complex models for the fish



the **central aim** of designing a classifier is to correctly classify **novel input**

Optimal tradeoff

The decision boundary shown might represent the optimal tradeoff between **performance on the training set** and **simplicity of classifier**.

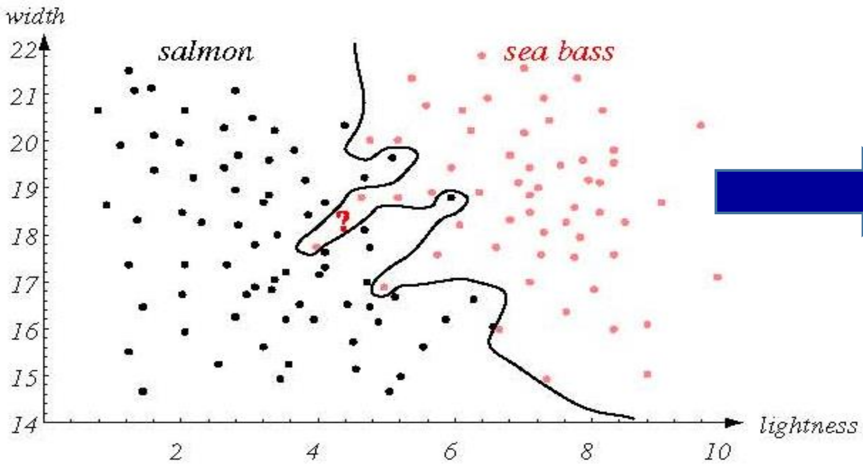


Generalization

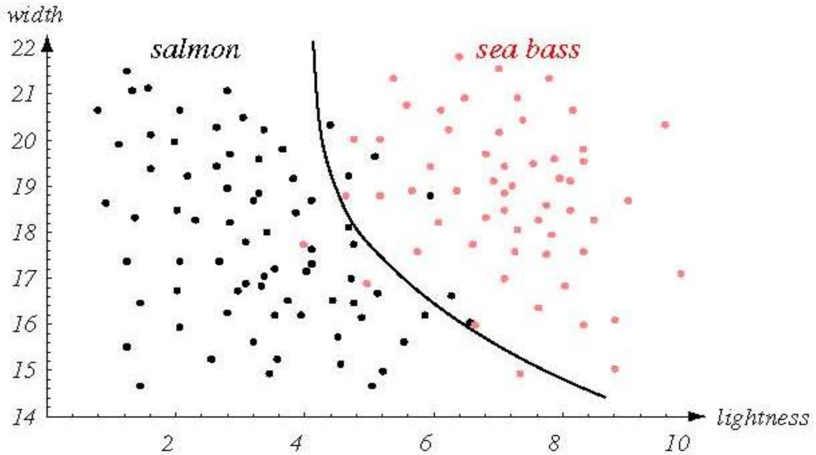
- Generalization is defined as the ability of a classifier to produce correct results on **novel** patterns.

- How can we improve generalization performance ?
  - More training examples (i.e., better model estimates).
  - Simpler models usually yield better performance.

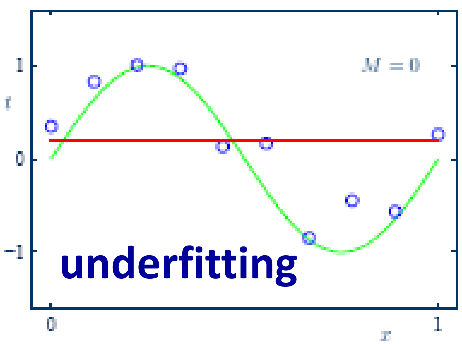
complex model



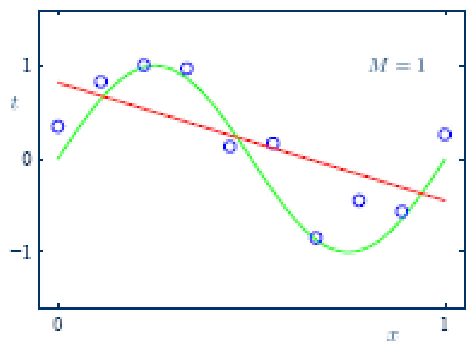
simpler model



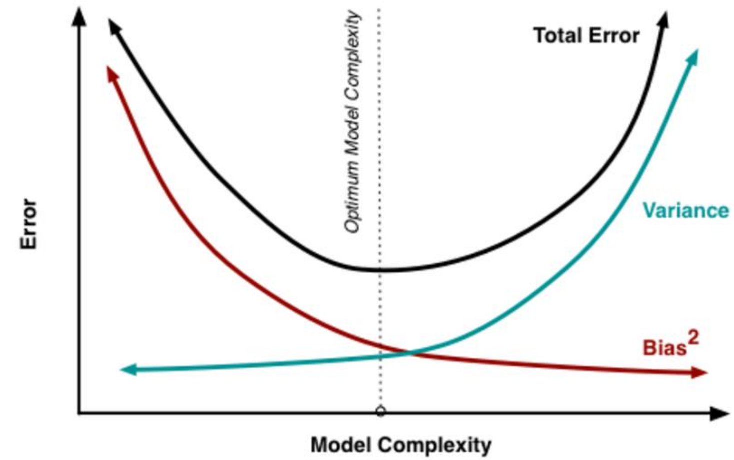
Understanding model complexity (theory of learning): The bias–variance tradeoff

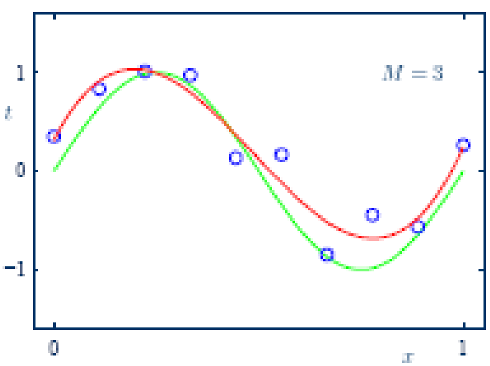


(a) 0'th order polynomial

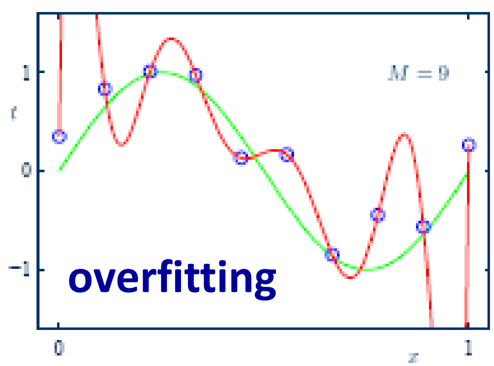


(b) 1'st order polynomial





(c) 3'rd order polynomial

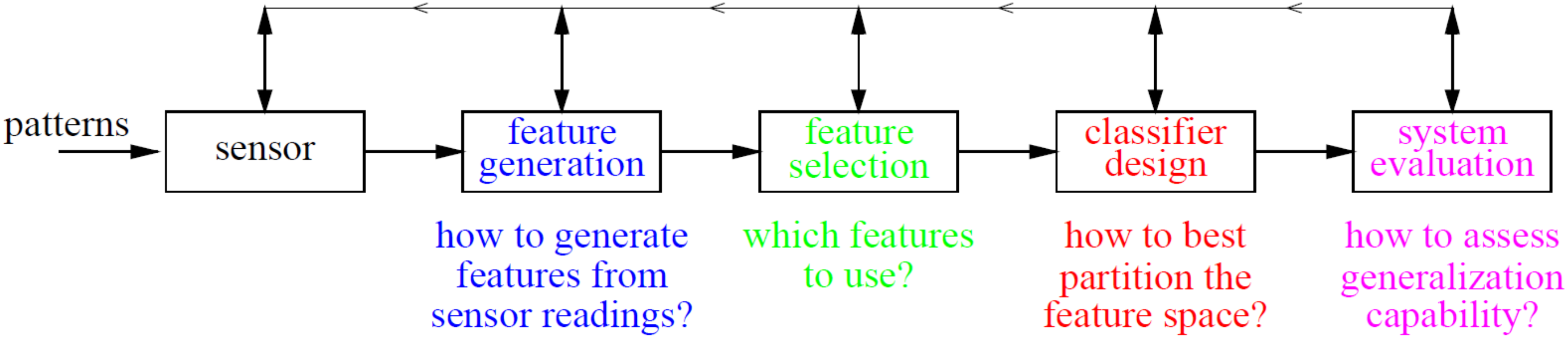


(d) 9'th order polynomial

bias–variance tradeoff is the **property** of a set of **predictive models** whereby models with a lower bias in **parameter estimation** have a higher variance of the parameter estimates across samples, and vice versa]

Underfitting vs. overfitting

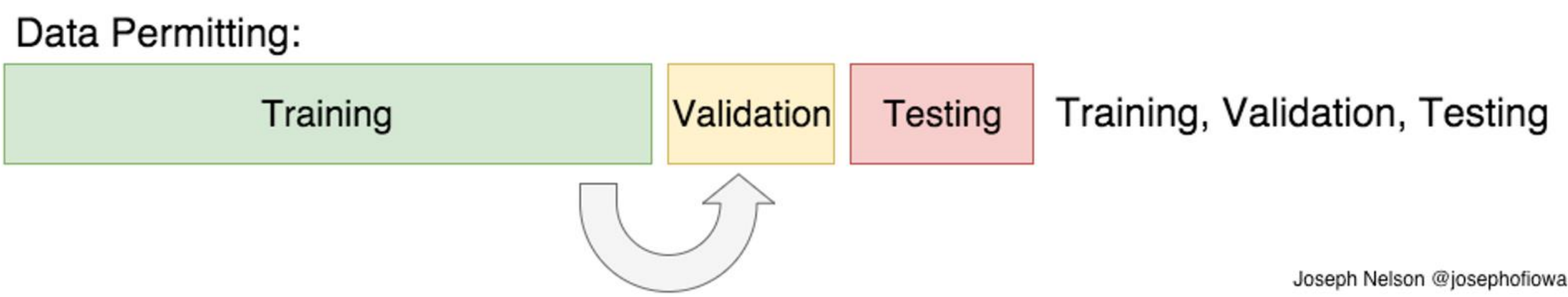
- The **bias** error: (**underfitting**)
  - From **erroneous assumptions** in the learning algorithm.
  - High bias can cause an algorithm to **miss the relevant relations** between features and target outputs.
- The **variance** error: (**overfitting**).
  - From **sensitivity** to **small fluctuations** in the training set.
  - High variance can cause an algorithm to **model the random noise** in the training data, rather than the intended outputs



- Feat. Gen.: Want to **reduce sensitivity to noise** and reduce complexity but **retain important Information**
  - Big sensor information into small number of features
- Feat. Sel.: Want to **reduce complexity** and reduce **redundancy** but retain important information
  - Select small set of features that **separates classes**
- Classif. Des.: Want **small generalization error** and fast training and classification (i.e. low complexity)
- Sys. Eval.: Want to accurately **estimate classier's generalization error**
- Some stages might be combined
- Feedback loops

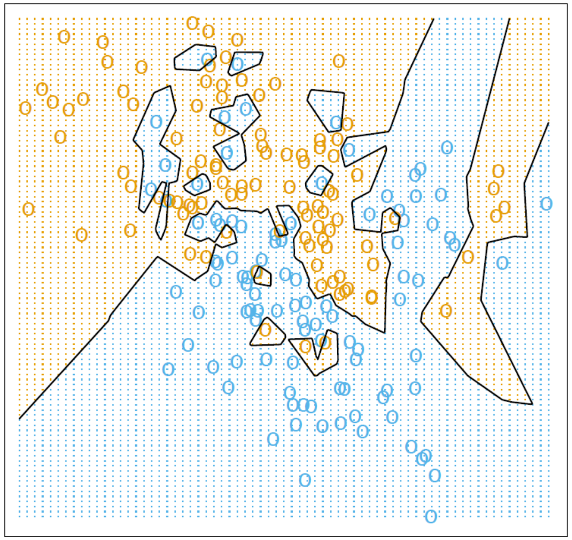
Now we have 3 sets:

- **training** set used to learn model weights
- **validation** set used to tune hyperparameters, choose among different models
- **test set** used as FINAL evaluation of model. Keep in a vault. Run ONCE, at the very end.

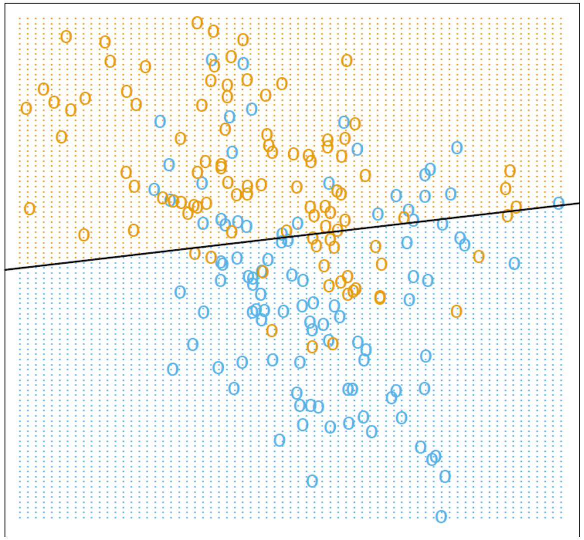


Classifier examples

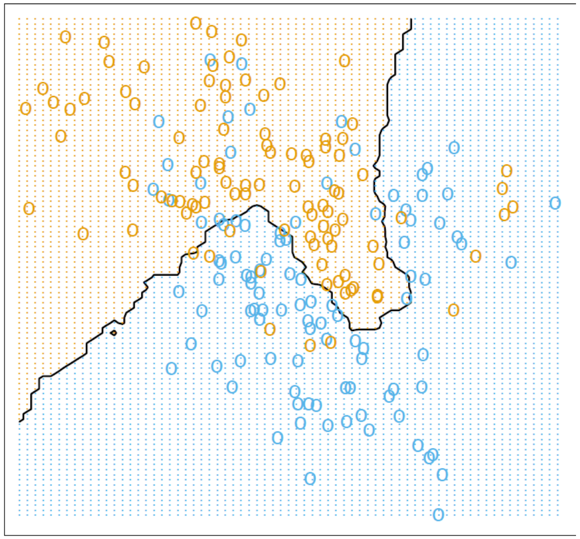
nearest neighbor classifier,



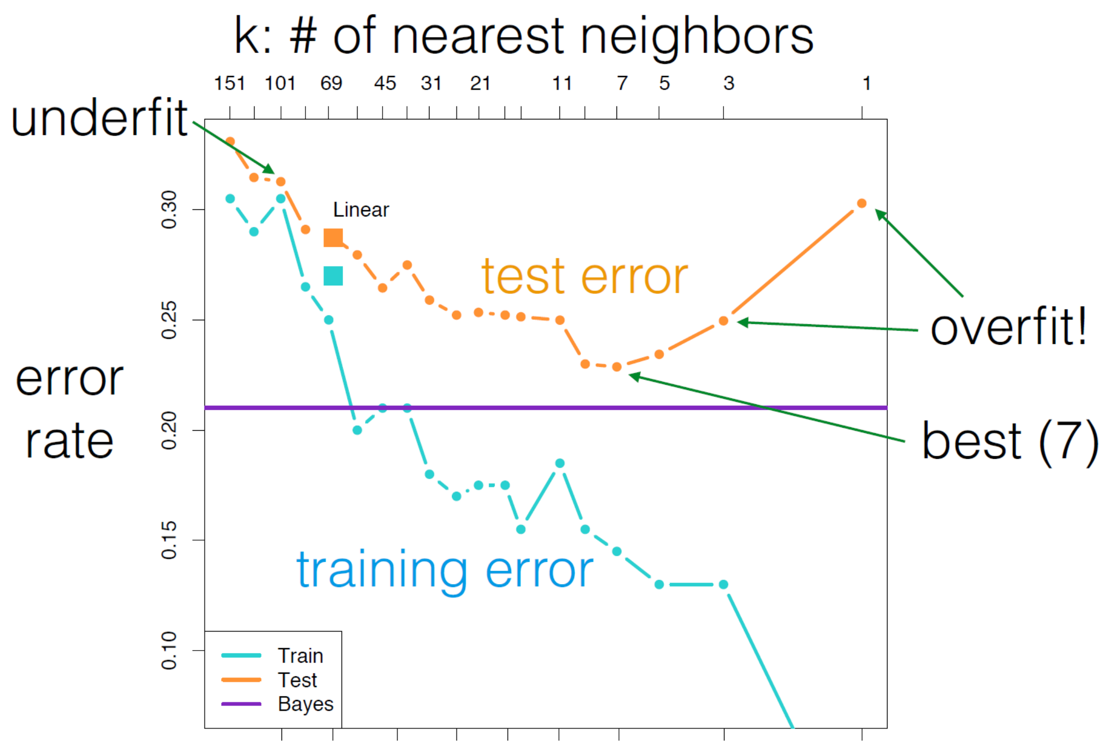
linear classifier



15-nearest neighbor classifier,

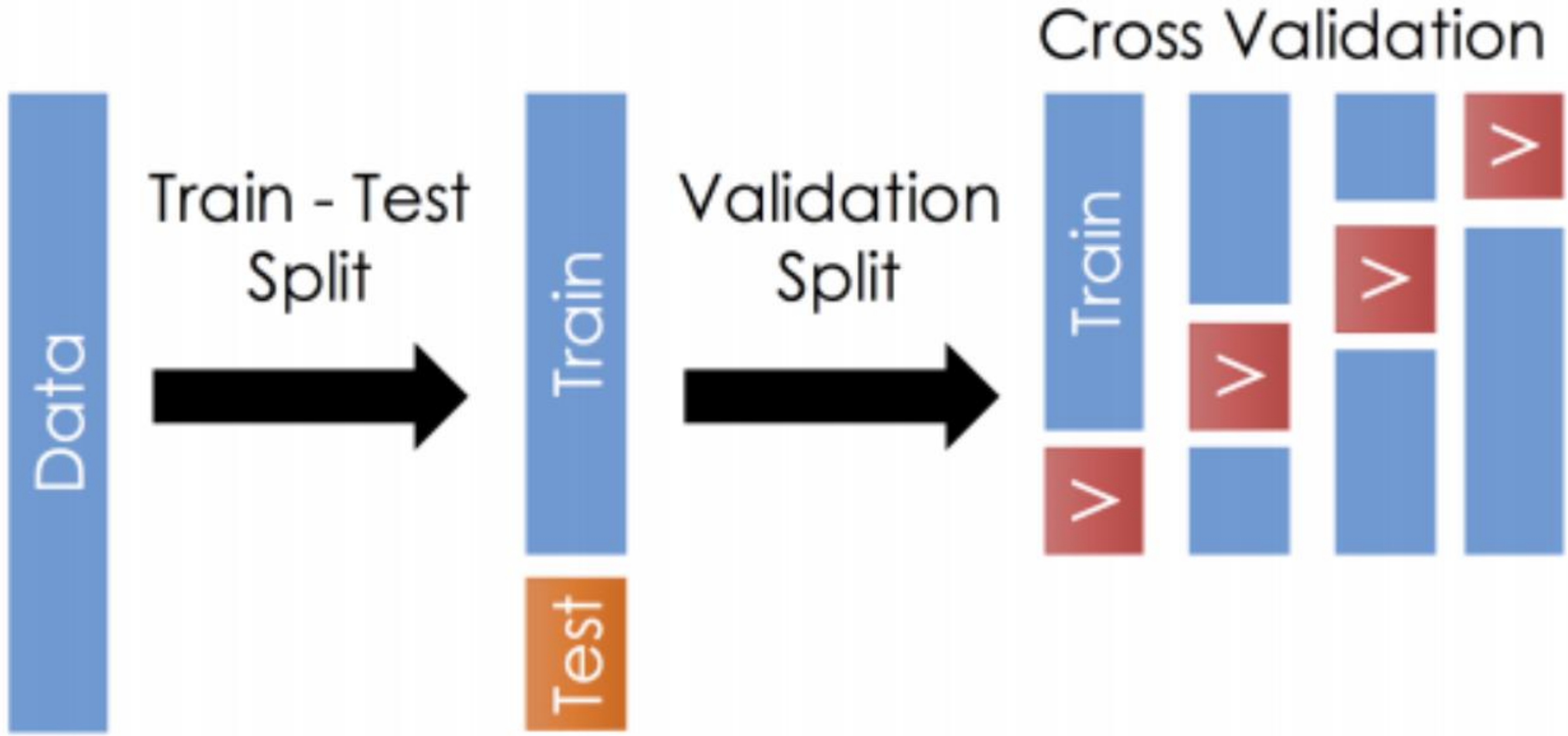


Hyperparameters



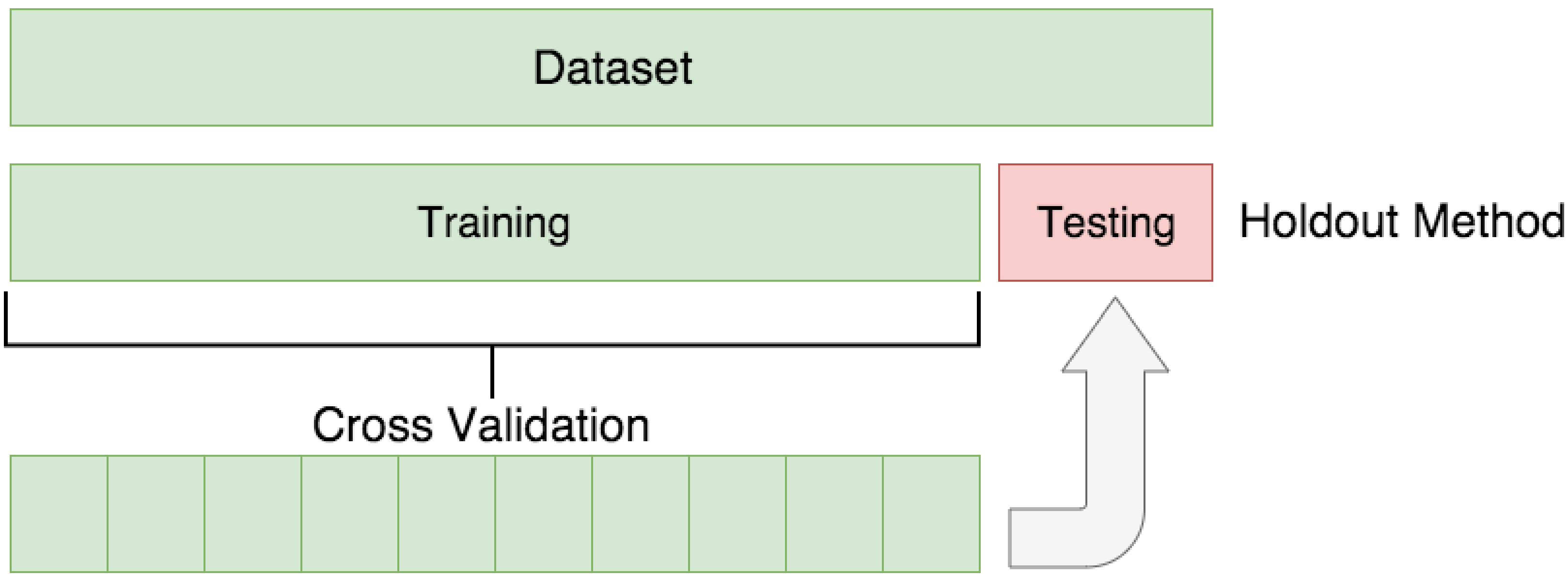
- Most ML algorithms have a few hyperparameters that control over/underfitting, e.g. k in k-nearest neighbors.
- We select them by validation

Cross validation



Holdout methods

- **K-Fold Cross Validation**
- **Leave P-out Cross Validation**
- **Leave One-out Cross Validation**



Post Processing

- Feature extraction
  - Discriminative features
  - Invariant features with respect to translation, rotation and scale.
- Classification
  - Use a feature vector provided by a feature extractor to assign the object to a category
- Post Processing
  - Exploit context input dependent information other than from the target pattern itself to improve performance

Feature Choice

Depends on the characteristics of the problem domain.

Simple to extract

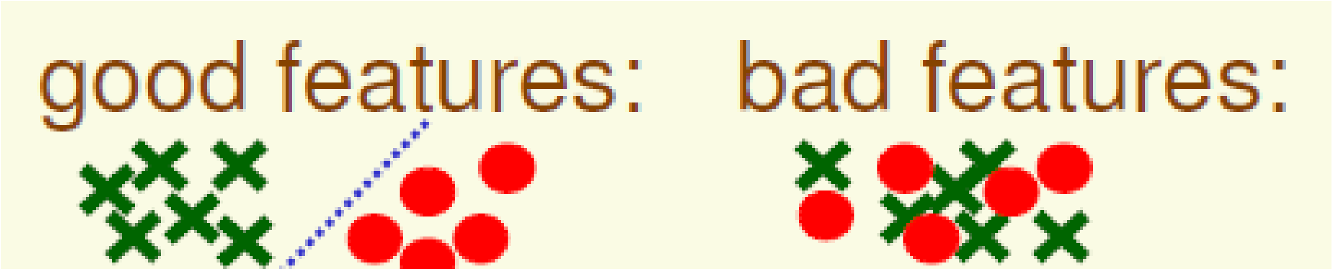
Invariant to irrelevant transformation

Insensitive to noise.

Missing Features problem

“Quality” of Features

- How to choose a good set of features?
  - **Discriminative** features



  - **Invariant** features (e.g., invariant to geometric transformations such as translation, rotation and scale)

Missing Features

- Certain features might be missing (e.g., due to occlusion).
- How should we train the classifier with missing features ?
- How should the classifier make the best decision with missing features ?

Cost of miss-classifications

- Fish classification: two possible classification errors:

(1) Deciding the fish was a sea bass when it was a salmon.

(2) Deciding the fish was a salmon when it was a sea bass.

- Are both errors equally important ?

Cost of miss-classifications

- Suppose that:
    - Customers who buy salmon will object vigorously if they see sea bass in their cans. (false alarm; type I error)
    - Customers who buy sea bass will not be unhappy if they occasionally see some expensive salmon in their cans. (missed called; type II error)

- How does this knowledge affect our decision?

Computational Complexity

- What is the trade-off between **computational ease (or complexity)** and **performance**?

- (How an algorithm **scales** as a function of the number of features, patterns or categories?)

Time Complexity of Algorithms

- Big-Theta
  - The function $g(n)$ is $\Theta(f(n))$ iff there exist two real positive constants $c_1 > 0$ and $c_2 > 0$ and a positive integer n_0 such that:

    $c_1 f(n) \geq g(n) \geq c_2 f(n)$ for all $n \geq n_0$
- Big-Oh
  - Upper bounds of complexity
- Big-Omega
  - Lower bound ($g(n) \geq c f(n)$ for all $n \geq n_0$)

Ascending order of complexity

$1 \leftarrow \log\log n \leftarrow \log n \leftarrow n \leftarrow n\log n \leftarrow n^{\alpha};\ 1<\alpha<2 \leftarrow n^2 \leftarrow n^3$

$\leftarrow n^m;\ m>3 \leftarrow 2^n \ldots$

Running time T(*n*)	**Complexity O(*n*)**
$n^2 + 100\,n + 1$	$O(n^2)$
$0.001n^3 + n^2 + 1$	$O(n^3)$
$23\,n$	$O(n)$
$100000\,n^2 + 10000\,n$	$O(n^2)$
2^{3+n}	$O(2^n)$ as $2^{3+n} \equiv 2^3 \cdot 2^n$
$2 \cdot 3^n$	$O(3^n)$