Predictive Learning from Data

LECTURE SET 2

Problem Setting, Basic Learning Problems and Inductive Principles

OUTLINE

2.0 Objectives + Background

- formalization of inductive learning
- classical statistics vs predictive approach

2.1 Terminology and Learning Problems

2.2 Basic Learning Methods and Complexity Control

2.3 Inductive Principles

2.4 Alternative Learning Formulations

2.5 Summary

2.0 Objectives

- To quantify the notions of explanation, prediction and model
- Introduce terminology
- Describe common learning problems

- Past observations ~ data points
- Explanation (model) ~ function

Learning ~ function estimation (from data)

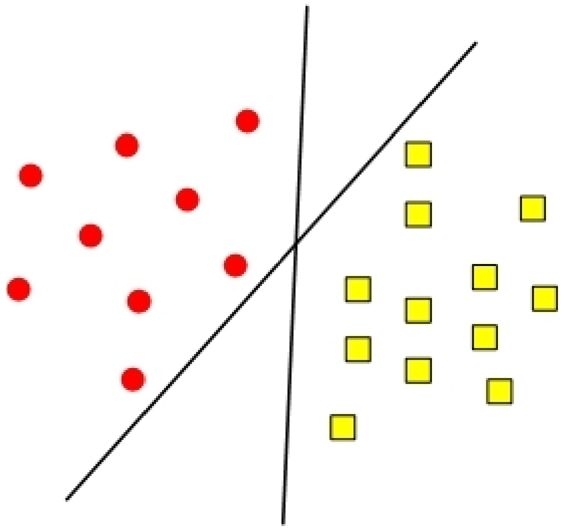
Prediction ~using the model to predict new inputs

- *Example:* classification problem

  training samples, model

  Goal 1: explain training data ~ min training error

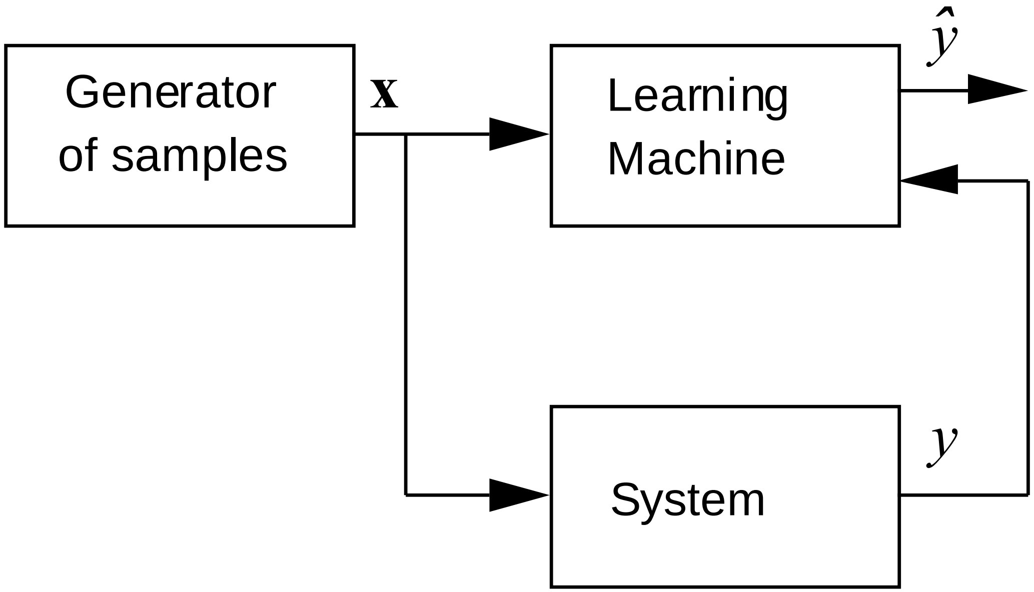
  Goal 2: generalization (for future data)



- Learning (model estimation) is ill-posed

Mathematical formalization

- Learning machine ~ predictive system



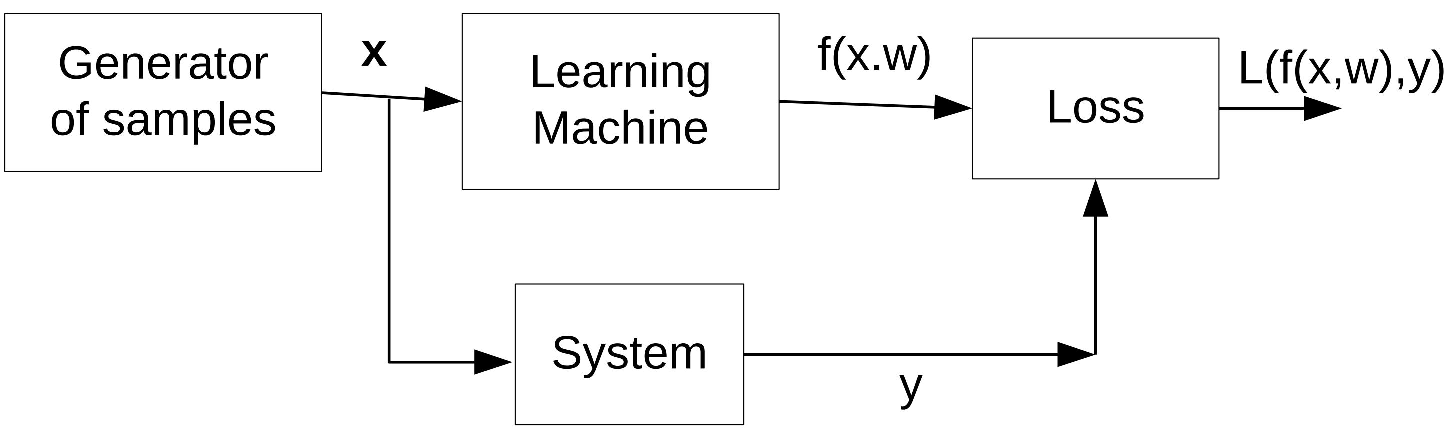
- Unknown joint distribution *P(**x**,y)*
- *Set of functions* (possible models) $\boldsymbol{f}(\mathbf{x}, \omega)$
- **Pre-specified** *Loss function* $L(y, \boldsymbol{f}(\mathbf{x}, \omega))$

(by convention, non-negative *Loss*)

Inductive Learning Setting

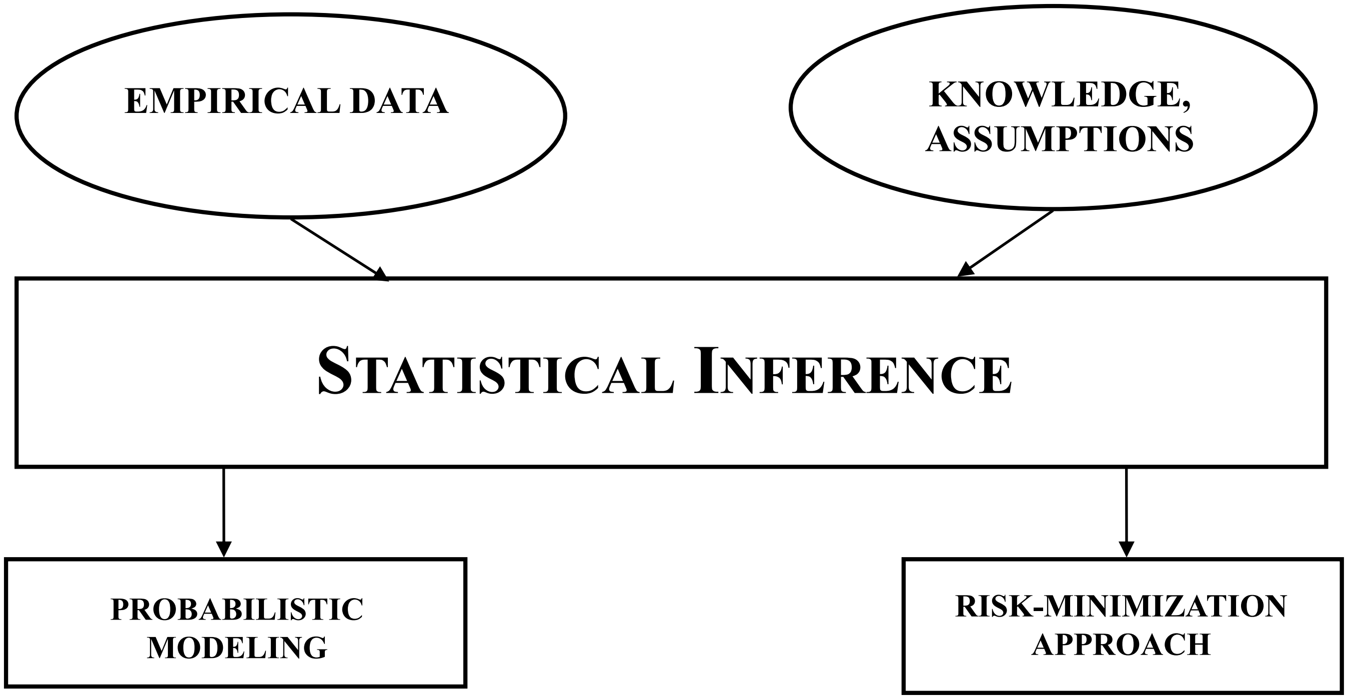
- The *learning machine* observes samples (**x** ,*y*), and returns an estimated response $\hat{y} = f(\mathbf{x}, w)$
- Two types of inference: identification vs imitation

- Risk $\int Loss(y, f(\mathbf{x},w))\, dP(\mathbf{x},y) \to min$



Two Views of Empirical Inference

- Two approaches to empirical or statistical inference



- These two approaches are different both technically and conceptually

Classical Approaches to Inductive Inference

Generic problem: *finite data → Model*

(1) Classical Science ~ hypothesis testing

experimental data is generated by a given model (*single function* ~ scientific theory)

(2) Classical statistics ~ *max likelihood*

~ data generated by a parametric model for density.

Note: loss fct ~ likelihood *(not problem-specific)*

~The same solution approach for all types of problems

R. Fisher: *"uncertain inferences" from finite data*

see: R. Fisher (1935), The Logic of Inductive Inference, *J. Royal Statistical Society*, available at http://www.dcscience.net/fisher-1935.pdf

Discussion

- Math formulation useful for quantifying
  - explanation ~ fitting error (training data)
  - generalization ~ prediction error
- Natural assumptions
  - future similar to past: *stationary P(**x**,y),* i.i.d.data
  - discrepancy measure or loss function, i.e. mean squared error (MSE)
- What if these assumptions do not hold?

OUTLINE

2.0 Objectives

2.1 Terminology and Learning Problems

- supervised/ unsupervised
- classification
- regression etc.

2.2 Basic Learning Methods and Complexity Control

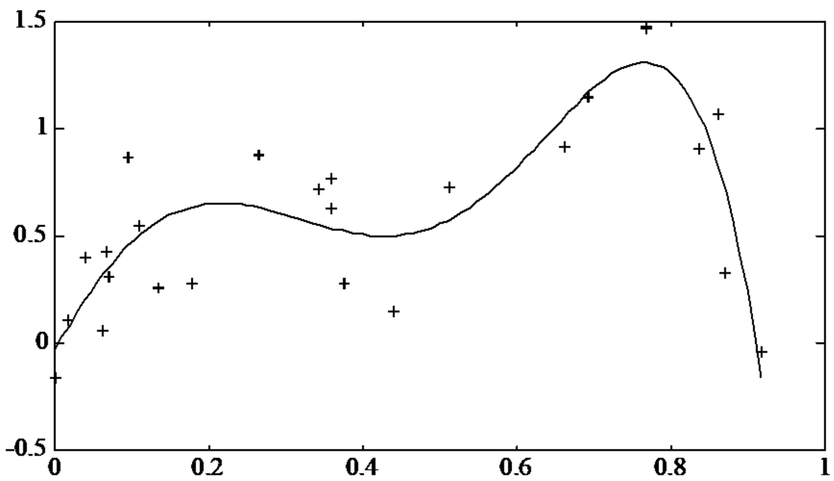
2.3 Inductive Principles

2.4 Alternative Learning Formulations

2.5 Summary

Supervised Learning: Regression

- Data in the form ($\mathbf{x}$,y), where
  - $\mathbf{x}$ is multivariate input (i.e. vector)
  - y is univariate output ('response')
- Regression: y is real-valued $L(y, f(\mathbf{x})) = \left(y - f(\mathbf{x})\right)^2$



→ Estimation of real-valued function

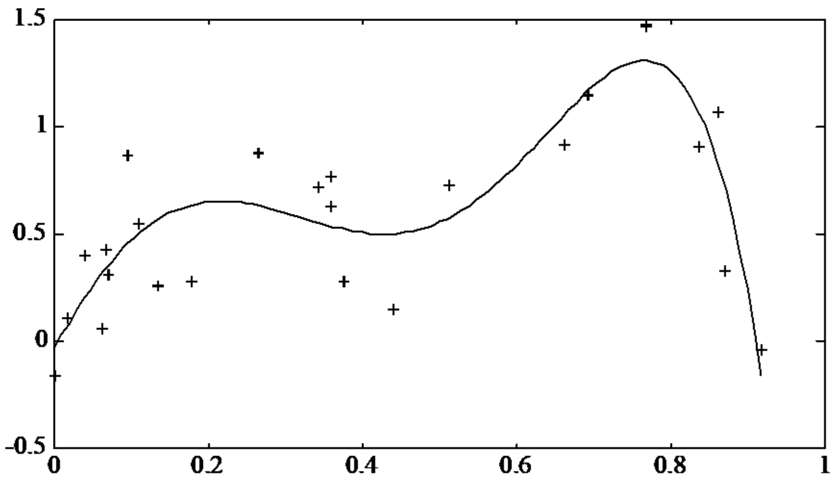
Regression Estimation Problem

Given: training data $(\mathbf{x}_i, y_i), i = 1,2,...n$

Find a function $f(\mathbf{x}, w^*)$ that minimizes squared error for a large number (*N*) of future samples:

$$\sum_{k=1}^{N} [(y_k - f(\mathbf{x}_k, w)]^2 \to min$$

$$\int (y - f(\mathbf{x},w))^2 \, dP(\mathbf{x},y) \to min$$



BUT future data is unknown ~ *P(**x**,y)* unknown

→ All estimation problems are ill-posed

Supervised Learning: Classification

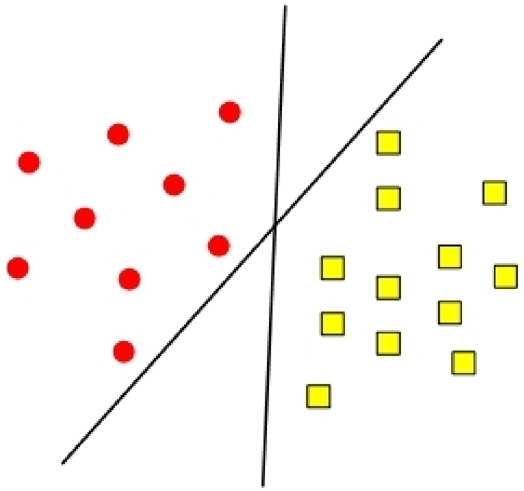
- Data in the form (**x**,y), where

  - **x** is multivariate input (i.e. vector)

  - y is univariate output ('response')

- Classification: y is categorical (class label)

$$L(y, f(\mathbf{x})) = \begin{cases} 0 & \text{if } y = f(\mathbf{x}) \\ 1 & \text{if } y \neq f(\mathbf{x}) \end{cases}$$



→ Estimation of indicator function

Density Estimation

- Data in the form (**x**), where
  - **x** is multivariate input (feature vector)
- Parametric form of density is given: $\boldsymbol{f}(\mathbf{x},\omega)$
- The loss function is likelihood or, more common, the negative log-likelihood

$$L(\boldsymbol{f}(\mathbf{x},\omega)) = -\boldsymbol{\ln} \boldsymbol{f}(\mathbf{x},\omega)$$

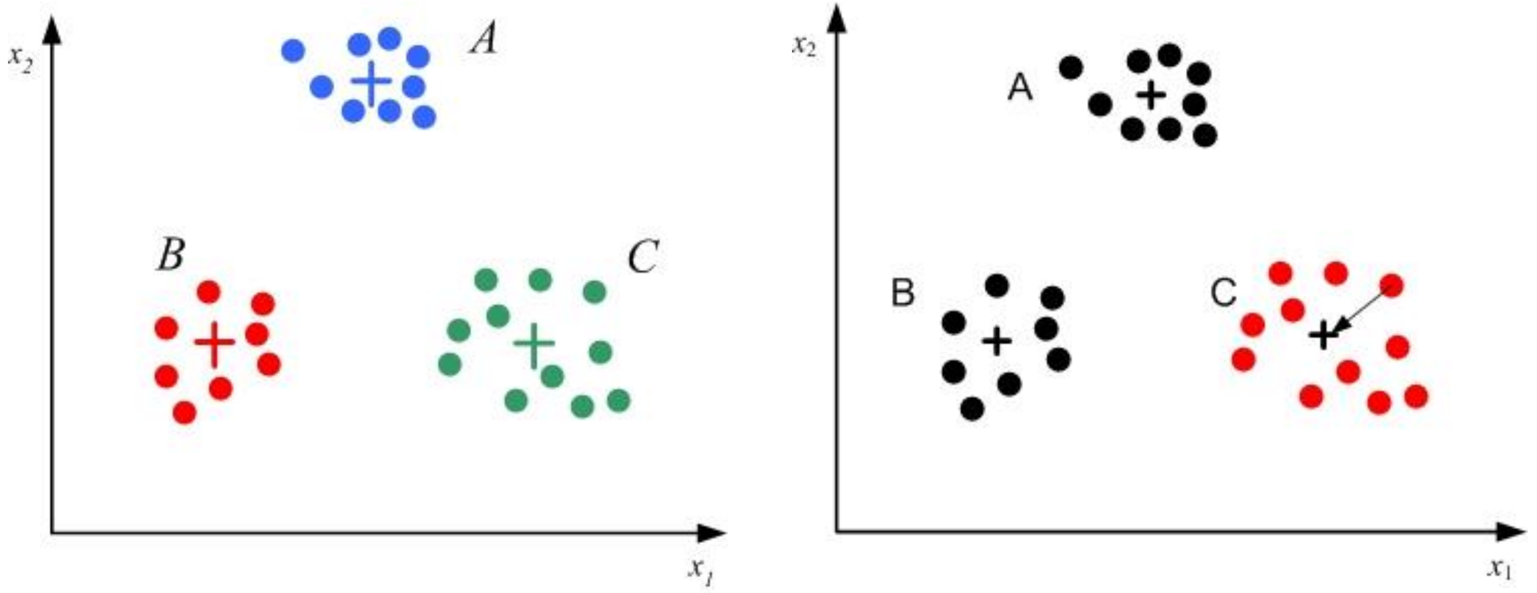
- The goal of learning is minimization of

$$R(\omega) = \int -\boldsymbol{\ln} \boldsymbol{f}(\mathbf{x},\omega)\boldsymbol{p}(\mathbf{x})d\mathbf{x}$$

from finite training data, yielding $\boldsymbol{f}(\mathbf{x},\omega_0)$

Unsupervised Learning 1

- Data in the form (**x**), where
  - **x** is multivariate input (i.e. feature vector)
- Goal: data reduction or clustering



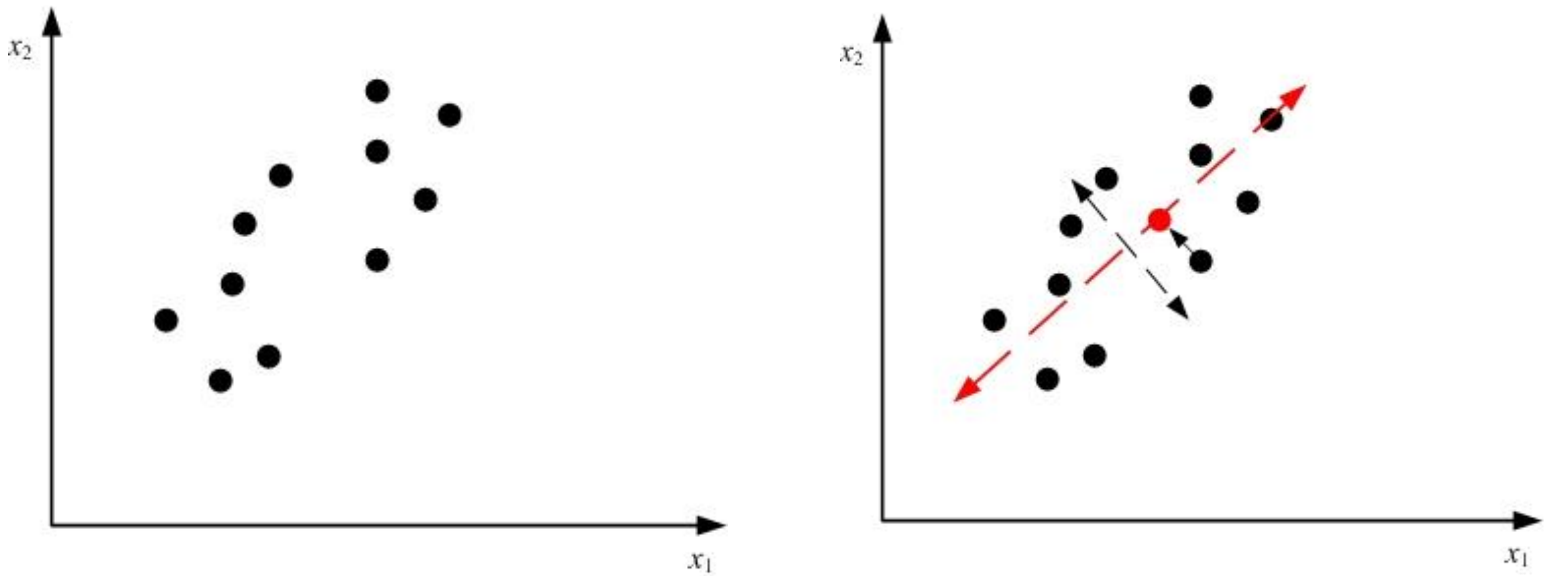
→ Clustering = estimation of mapping **X**→ **C**, where $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m\}$ and $L(\mathbf{x}, f(\mathbf{x})) = \left\| \mathbf{x} - f(\mathbf{x}) \right\|^2$

Unsupervised Learning 2

- Data in the form (**x**), where

  \- **x** is multivariate input (i.e. vector)

- Goal: dimensionality reduction



→ Mapping $f(\mathbf{x})$ is projection of the data onto low-dimensional subspace, minimizing loss

$$L(\mathbf{x}, f(\mathbf{x})) = \left\| \mathbf{x} - f(\mathbf{x}) \right\|^2$$

OUTLINE

2.0 Objectives

2.1 Terminology and Learning Problems

2.2 Basic Learning Methods and Complexity Control

- Parametric modeling
- Non-parametric modeling
- Data reduction
- Complexity control

2.3 Inductive Principles

2.4 Alternative Learning Formulations

2.5 Summary

Basic learning methods

General idea

- Specify a **wide set** of possible models $f(\mathbf{x},\omega)$ where ω is an abstract set of 'parameters'
- Estimate model parameters ω^* by minimizing *given loss function* for training data (~ ERM)

Learning methods differ in

- Chosen parameterization
- Loss function used
- Optimization method used for parameter estimation

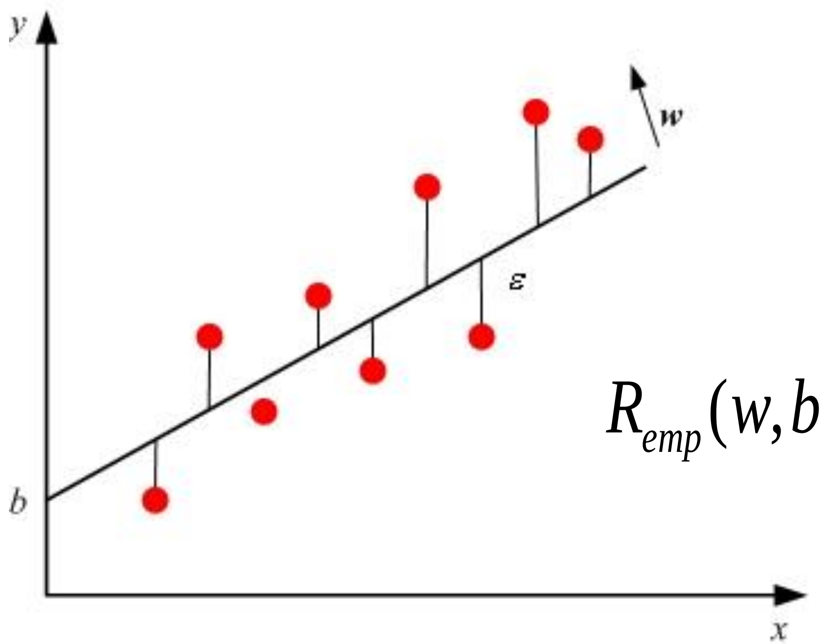
Parametric Modeling (~ERM)

Given training data $(\mathbf{x}_i, y_i), i = 1,2,\ldots n$

(1) Specify parametric model

(2) Estimate its parameters (via fitting to data)

- Example: Linear regression F(**x**)= (**w x**) + b



$$R_{emp}(w,b) = \frac{1}{n}\sum_{i=1}^{n}\left[y_i - (w \cdot x_i) - b\right]^2 \to min$$

Parametric Modeling: classification

Given training data $(\mathbf{x}_i, y_i), i = 1,2,...n$

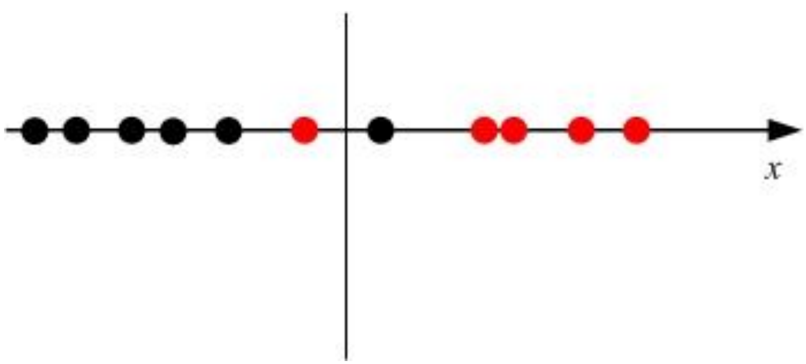
(1) Specify parametric model

(2) Estimate its parameters (via fitting to data)

Example: univariate classification data set

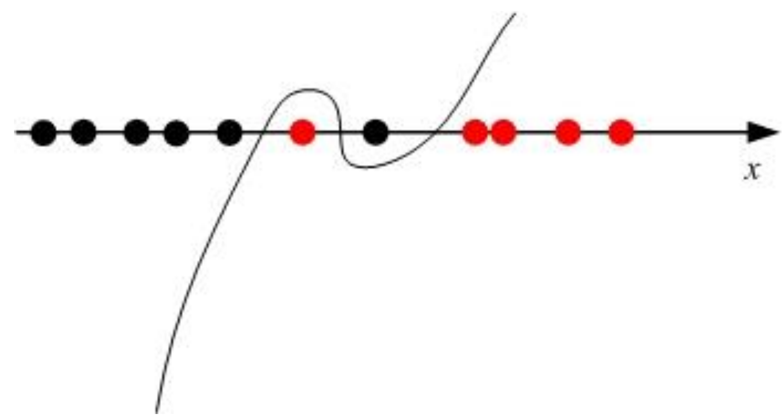
(a) Linear decision boundary

$$f(x) = sign(x - b)$$



(b) third-order polynomial

$$f(x) = sign(x^2 + wx + b)$$



Parametric Methods in Classical Statistics

- Learning ~ density estimation, i.i.d. data
- **Maximum Likelihood** inductive principle:

Given *n* training samples **X**, find **w*** maximizing

$$P[\text{data}|\text{model}] = P(\mathbf{X}|\mathbf{w}) = \prod_{i=1}^{n} p(\mathbf{x}_i\,;\mathbf{w})$$

equivalently, *minimize negative log-likelihood*

See textbook, Section 2.2, for example:

- Estimate two parameters of normal distribution from i.i.d. data samples via max likelihood
- → empirical mean and empirical variance)

Maximum Likelihood (cont'd)

- Similar approach for regression ~ for known parametric distribution (normal noise) → maximizing likelihood ~ min squared loss
- Similar approach for classification: for known class distributions (Gaussian) maximizing likelihood → second-order decision boundary

General approach: (statistical decision theory)

- Start with parametric form of a distribution
- Estimate its parameters via max likelihood
- Use estimated distributions for making decision (prediction)

Non-Parametric Modeling

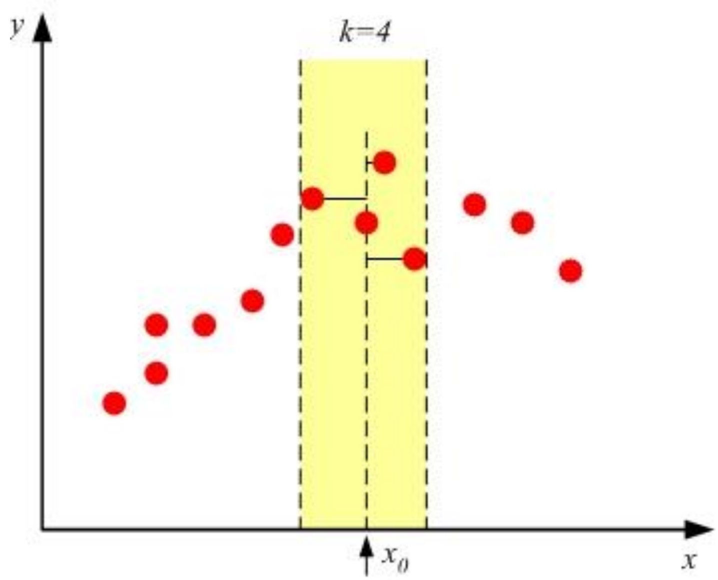
Given training data $(\mathbf{x}_i, y_i), i = 1,2,\ldots n$

Estimate the model (for given $\mathbf{x}_0$) as

'local average' of the data.

Note: need to define 'local', 'average'

- Example: k-nearest neighbors regression



$$f(\mathbf{x}_0) = \frac{\sum_{j=1}^{k} y_j}{k}$$

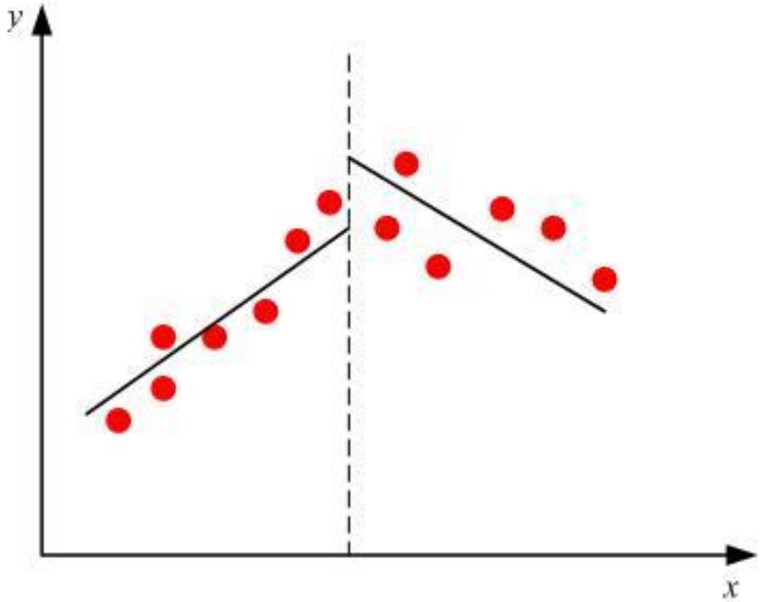
Data Reduction Approach

Given training data, estimate the model as 'compact encoding' of the data.

Note: 'compact' ~ # of bits to encode the model

or # of bits to encode the data (MDL)

- Example: piece-wise linear regression



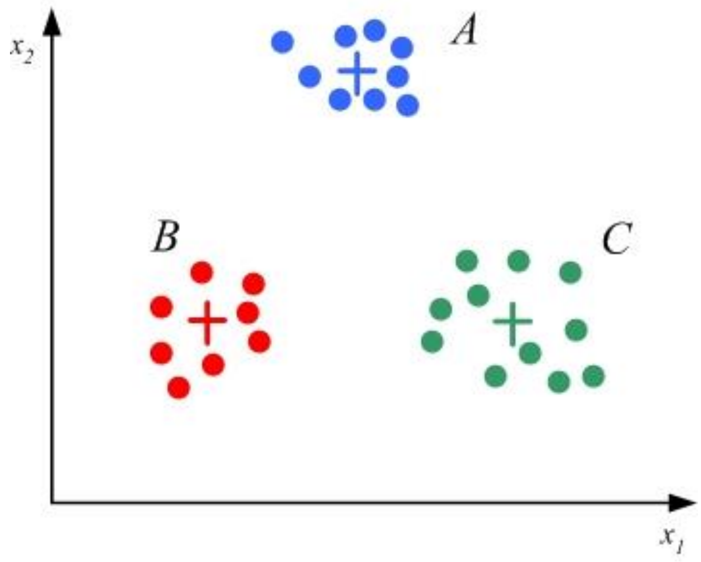
How many parameters needed for two-linear-component model?

Data Reduction Approach (cont'd)

Data Reduction approaches are commonly used for unsupervised learning tasks.

- Example: clustering.

Training data encoded by 3 points (cluster centers)



Issues:

- How to find centers?
- How to select the number of clusters?

Diverse terminology (for learning methods)

- Many methods differ in parameterization of admissible models or approximating functions $\hat{y} = f(\mathbf{x}, w)$

  - neural networks

  - decision trees

  - signal processing (~ wavelets)

- How training samples are used:

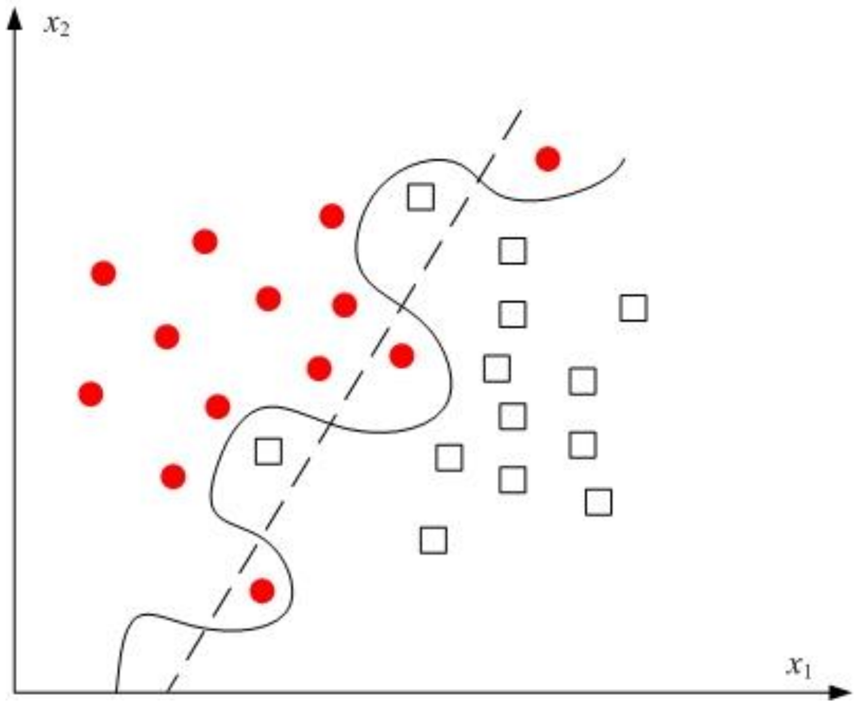
  Batch methods

  On-line or flow-through methods

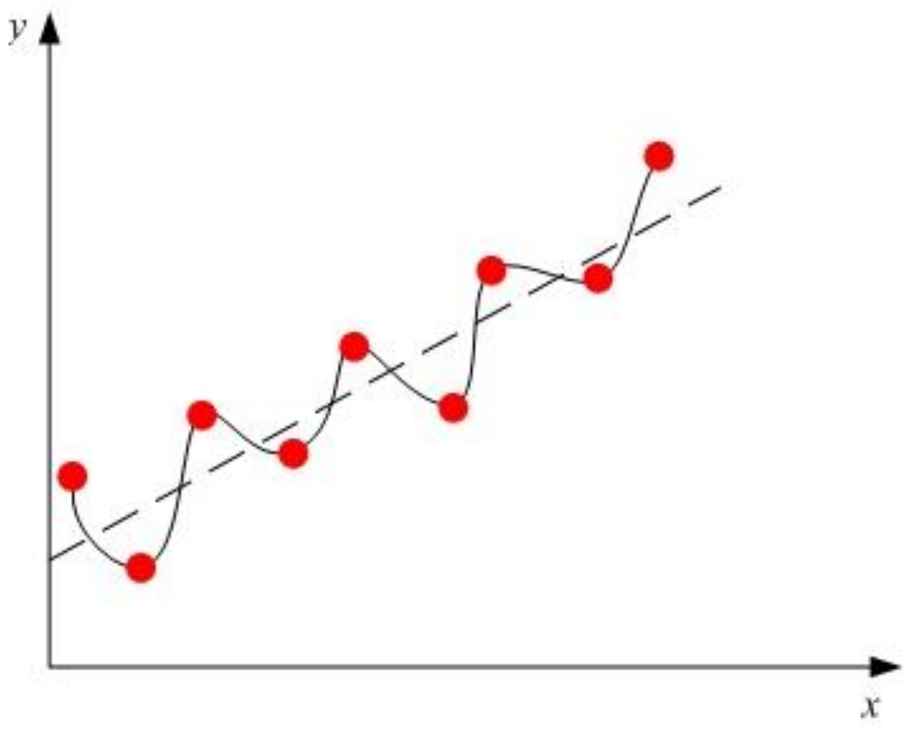
Motivation for Complexity Control

Effect of model control on generalization

(a) Classification



(b) Regression



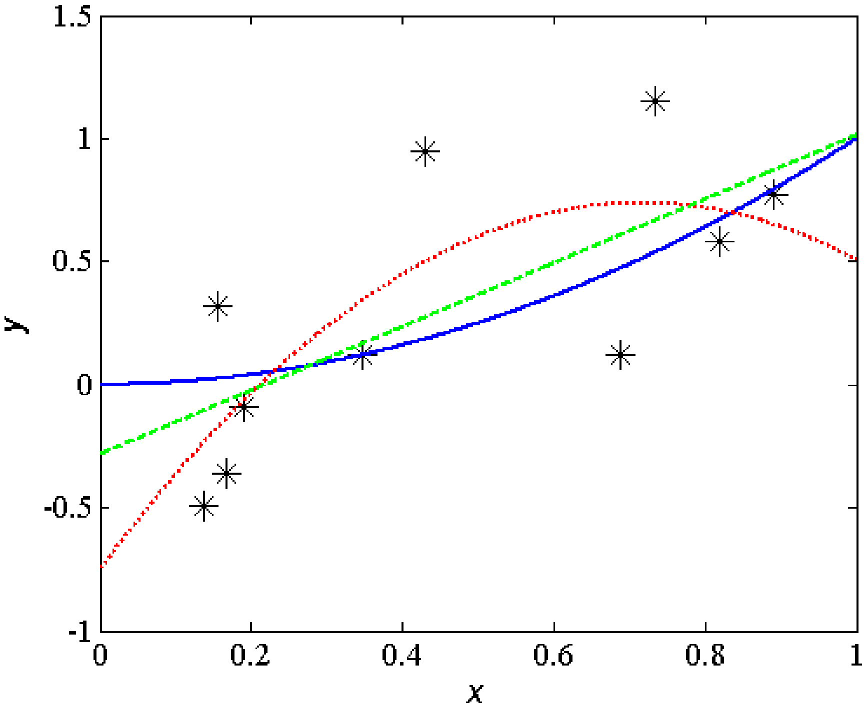
Complexity Control: parametric modeling

Consider regression estimation

- Ten training samples

$$y = x^2 + N(0, \sigma^2), where \sigma^2 = 0.25$$

- Fitting linear and 2-nd order polynomial:



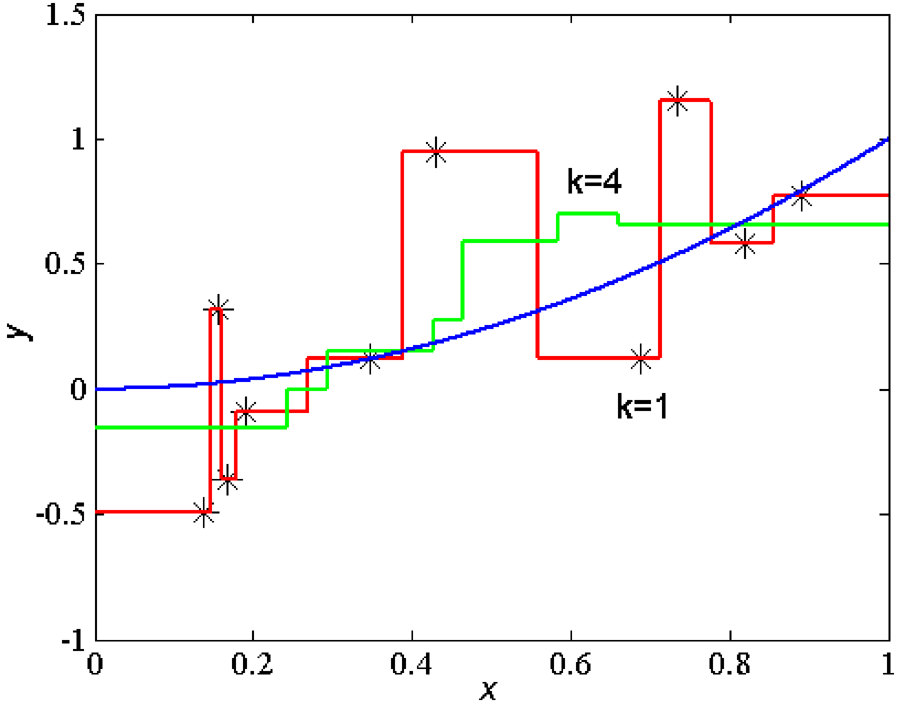
Complexity Control: local estimation

Consider regression estimation

- Ten training samples from

$$y = x^2 + N(0, \sigma^2), where \sigma^2 = 0.25$$

- Using k-nn regression with k=1 and k=4:



Complexity Control (summary)

- Complexity (of admissible models) affects generalization (for future data)
- Specific complexity indices for
  - Parametric models: ~ # of parameters
  - Local modeling: *size of local region*
  - Data reduction: # of clusters
- Complexity control = choosing optimal complexity (~ good generalization) for given (training) data set
- not well-understood in classical statistics

OUTLINE

2.0 Objectives

2.1 Terminology and Learning Problems

2.2 Basic Learning Methods and Complexity Control

2.3 Inductive Principles

- Motivation
- Inductive Principles: Penalization, SRM, Bayesian Inference, MDL

2.4 Alternative Learning Formulations

2.5 Summary

Conceptual Motivation

- Generalization from finite data requires:

  a priori knowledge = *any info* outside training data, e.g. ???

  inductive principle = general strategies for combining a priori knowledge and data

  learning method = constructive implementation of inductive principle

- **Example: Empirical Risk Minimization** ~ parametric modeling approach

Question: what are possible limitations of ERM?

Motivation (cont'd)

- Need for flexible (adaptive) methods

  - **wide (~ flexible) parameterization**

  → ill-posed estimation problems

  - need provisions for **complexity control**

- **Inductive Principles** originate from statistics, applied math, info theory, learning theory – and they adopt distinctly different terminology & concepts

Inductive Principles

- **Inductive Principles differ in terms of**

  - representation of a priori knowledge

  - **mechanism** for combining a priori knowledge with training data

  - **applicability** when the true model does not belong to admissible models

  - **availability** of constructive procedures (learning methods/ algorithms)

Note: usually prior knowledge about parameterization

PENALIZATION

- Overcomes the limitations of ERM
- Penalized empirical risk functional

$$R_{pen}(\omega) = R_{emp}(\omega) + \lambda \phi[\mathrm{f}(\mathbf{x}, \omega)]$$

$\phi[\mathrm{f}(\mathbf{x}, \omega)]$ is non-negative penalty functional specified *a priori* (independent of the data); its larger values penalize complex functions.

λ is regularization parameter (non-negative number) tuned to training data

Example: ridge regression

Structural Risk Minimization

- Overcomes the limitations of ERM
- Complexity ordering on a set of admissible models, as a nested structure

$$S_0 \subset S_1 \subset S_2 \subset \ldots$$

Examples: a set of polynomial models, Fourier expansion etc.

- Goal of learning ~ minimization of empirical risk for an optimally selected element S_k

Bayesian Inference

- Probabilistic approach to inference
- Explicitly defines a priori knowledge as **prior probability** (distribution) on a set of model parameters
- Bayes formula for updating prior probability using the evidence given by training data:

$$P[\text{model}|\text{data}] = \frac{P[\text{data}|\text{model}]P[\text{model}]}{P[\text{data}]}$$

$P[\text{model}|\text{data}]$ ~ posterior probability

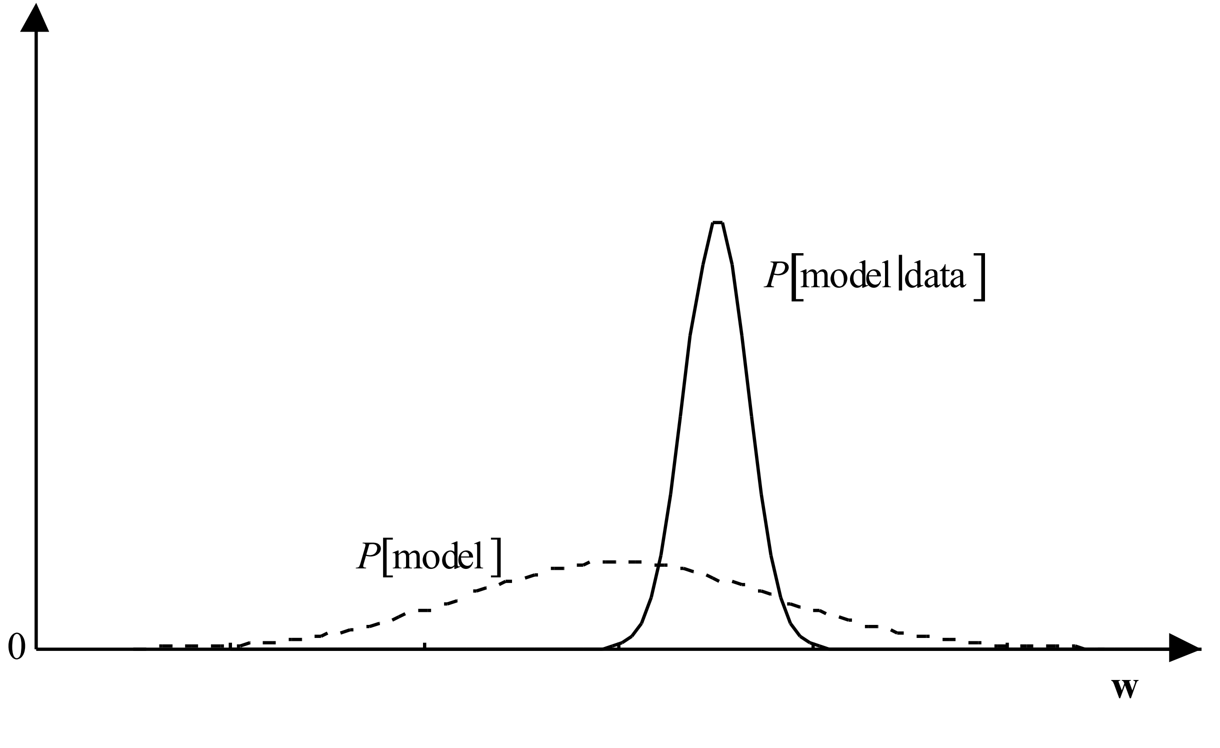
$P[\text{data}|\text{model}]$ ~ likelihood (probability that the data are generated by a model)

Bayesian Density Estimation

- Consider parametric density estimation where prior probability distribution $P[\text{model}] = \boldsymbol{p}(\mathbf{w})$

  Given training data **X**, the posterior probability distribution is updated

$$\mathrm{p}(\mathbf{w}|\mathbf{X}) = \frac{P(\mathbf{X}|\mathbf{w})\mathrm{p}(\mathbf{w})}{P(\mathbf{X})}$$



Implementation of Bayesian Inference

- **Maximum Likelihood**,i.e. choose **w*** maximizing

$$P[\text{data}|\text{model}] = P(\mathbf{X}|\mathbf{w}) = \prod_{i=1}^{n} p(\mathbf{x}_i\,;\mathbf{w})$$

(equivalent to ERM)

- **True Bayesian inference** (averaging)

$$\Theta(\mathbf{x}|\mathbf{X}) = \int p(\mathbf{x};\mathbf{w})\,\boldsymbol{p}(\mathbf{w}|\mathbf{X})d\mathbf{w}$$

Where $p(\mathbf{x};\mathbf{w})$ is a set of admissible densities

and $\mathrm{p}(\mathbf{w}|\mathbf{X}) = \dfrac{P(\mathbf{X}|\mathbf{w})\mathrm{p}(\mathbf{w})}{P(\mathbf{X})}$

Minimum Description Length (MDL)

- Information-theoretic approach
  - any training data set can be optimally encoded
  - code length ~ generalization capability
- Related to the *Data Reduction* approach introduced (informally) earlier.
- Two possible implementations:
  - lossy encoding
  - lossless encoding of the data (as in MDL)

Binary Classification under MDL

- Consider training data set

  X={**x**k,yk} (k=1,2,...n) where y={0,1}
- Given data object **X**={**x**1**,..., x**n} is a binary string y1,...,yn random?

  if there is a dependency then the output string can be encoded by a **shorter code:**

  - the model having code length L (model)

  - the error term L(data | model)

  → the total length of such a code for string **y** is:

  b = L (model) + L(data | model)

  and the compression coefficient is K = **b** / n

Comparison of Inductive Principles

- Representation of a priori knowledge/ complexity:

  penalty term, structure, prior distribution, codebook
- Formal procedure for complexity control:

  penalized risk, optimal element of a structure, posterior distribution
- Constructive implementation of complexity control:

  resampling, analytic bounds, marginalization, minimum code length

  See Table 2.1 in [Cherkassky & Mulier, 2007]

OUTLINE

2.0 Objectives

2.1 Terminology and Learning Problems

2.2 Basic Learning Methods and Complexity Control

2.3 Inductive Principles

2.4 Alternative Learning Formulations

- Motivation
- Examples of non-standard formulations
- Formalization of application domain

2.5 Summary

Motivation

- **Estimation of predictive model**

  Step 1: Problem specification/ Formalization

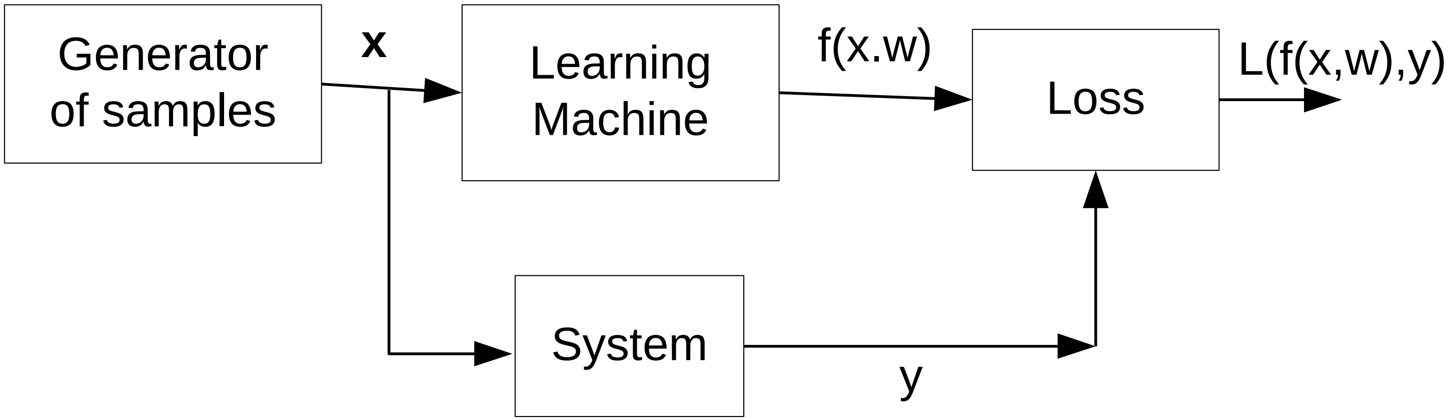
  Step 2: Model estimation, learning, inference

- **Standard Inductive Formulation**

  - usually assumed in all ML algorithms

  - certainly *may not be the best formalization* for given application problem

Standard Supervised Learning



- Available (training) data **format** (**x**,y)
- Test samples (**x**-values) are **unknown**
- **Stationary distribution, i.i.d samples**
- **Single model** needs to be estimated
- **Specific loss functions** adopted for common tasks (classification, regression etc.)

Non-standard Learning Settings

- **Available Data Format**

  - **x**-values of test samples are known during training

  → Transduction, semi-supervised learning

- **Different (non-standard) Loss Function**

  - see later example ‘learning the sign of a function’

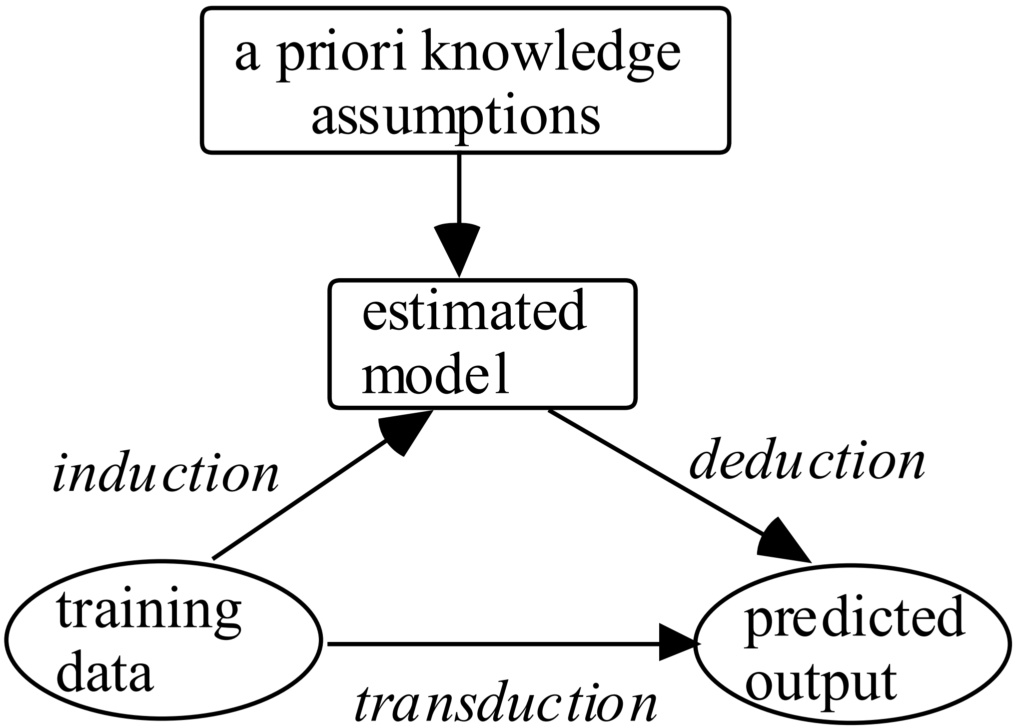
- **Univariate Output (~ a single model)**

  - multiple outputs may need to be estimated from available data

Transduction

~ predicting function values at given points:

- **Given** labeled training set + x-values of test data
- **Estimate (predict)** y-values for given test inputs



Learning sign of a function

- Given training data $(\mathbf{x}_i, y_i), i = 1,2,...n$

with y-values in a bounded range $y \in [-2,+2]$

Estimate function $f(\mathbf{x})$ predicting sign of y

Loss function $L(y, f(\mathbf{x})) = -yf(\mathbf{x})$

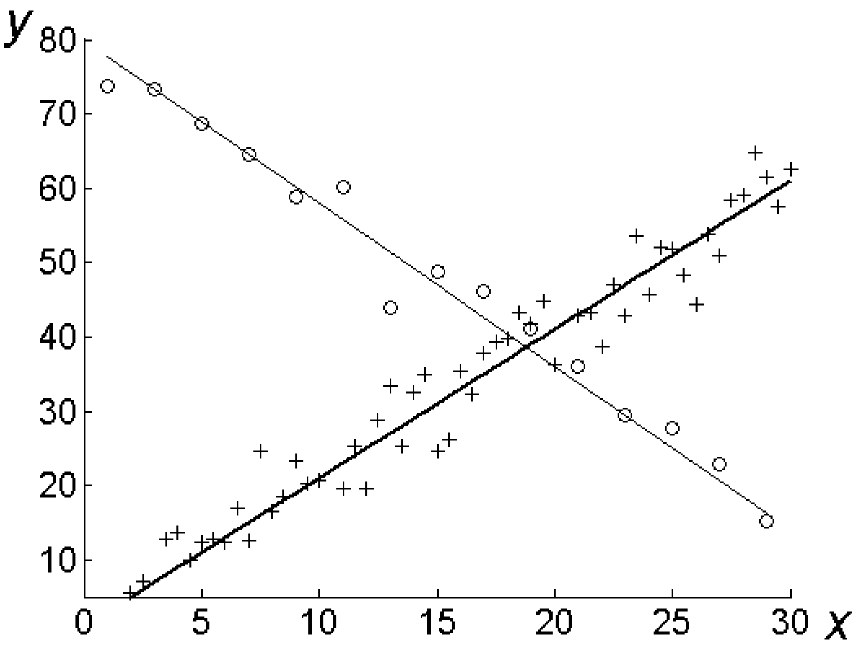
If prediction is wrong ~ real-valued loss $-|y|$

If prediction is correct ~ real-valued gain $+|y|$

- Neither standard regression, nor classification
- Practical application: frequent trading

Multiple Model Estimation

- Training data in the form (**x**,y), where
  - **x** is multivariate input
  - y is univariate real-valued output ('response')
- Similar to standard regression, but subsets of data may be described by different models



Formalization of Application Problems

- *Problem Specification Step* cannot be formalized

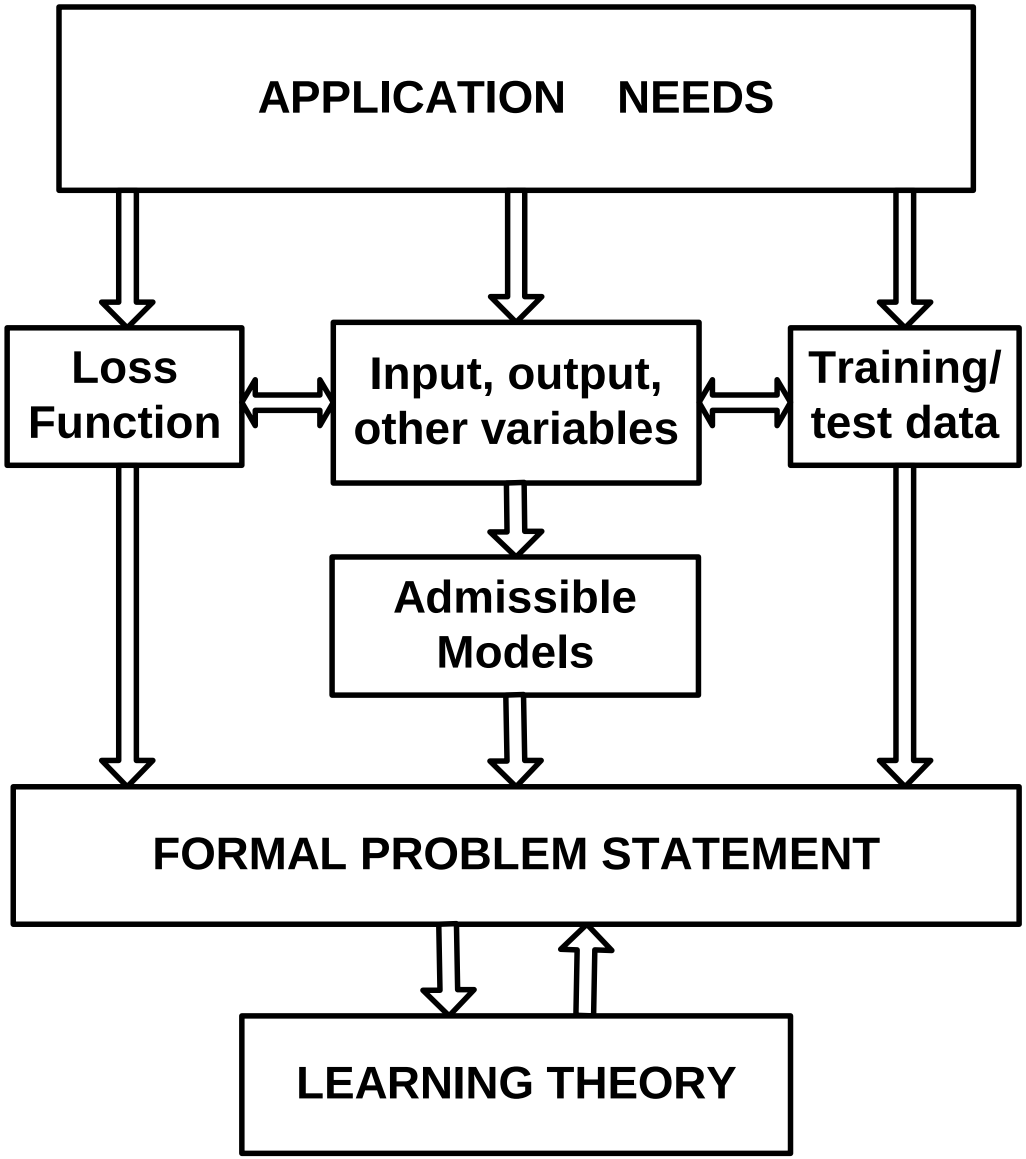
But

- **Several guidelines** can be helpful during formalization process

- Mapping process:

Application requirements → Learning formulation

- Specific components of this mapping process are shown next


APPLICATION NEEDS
Loss Function
Input, output, other variables
Training/ test data
Admissible Models
FORMAL PROBLEM STATEMENT
LEARNING THEORY

Summary

- Standard Inductive Learning ~ function estimation
- Goal of learning (empirical inference):
  to act/perform well, not system identification
- Important concepts:
  - training data, test data
  - loss function, prediction error (~ prediction risk)
  - basic learning problems
- Complexity control
- Inductive principles – which one is the 'best' ?

Summary (cont'd)

- Assumptions for inductive learning
- Non-standard learning formulations

Aside: predictive modeling of
physical systems vs social systems

Note: main assumption (stationarity) does not hold in social systems (business data, financial data etc.)

- For discussion think of example application that requires *non-standard* learning formulation

Note: (a) *do not* use examples similar to ones presented in my lectures and/or text book
(b) you can email your example to instructor (maximum half-a-page)