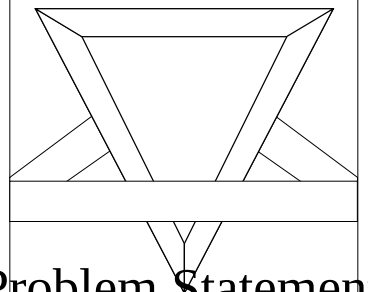
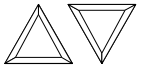
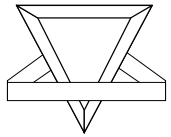


Problem Statement, Classical Approaches, and Adaptive Learning

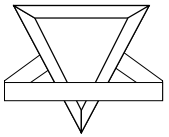
Learning From Data
Chapter 2





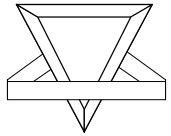
Outline (1/2)

- 2.1 Mathematical formulation of predictive learning problem
  - Classification, regression, density estimation, vector quantization
- 2.2 Classical statistical approaches to estimation from data
  - Parametric modeling (ML, ERM): Very rigid assumption about unknown dependency (discrepancy)
  - Nonparametric modeling: Learning with large samples (asymptotic case)



Outline (2/2)

- 2.3 Adaptive Methods
  - A priori assumption
  - Choosing a model of optimal complexity for finite data
    - How to incorporate a priori assumption into learning?
    - How to measure model complexity?
    - How to find an optimal balance between the data and a priori knowledge



2.1 Formulation of the Learning Problem

- Learning
  - Process of estimating an unknown dependency or structure of a system using a limited number of observations
- Three components
  - ① Generator: random input vector
  - ② System: returns an output for a given input vector
  - ③ Learning machine: estimates an unknown mapping of the system from the observed samples

2.1 Problem Formulation

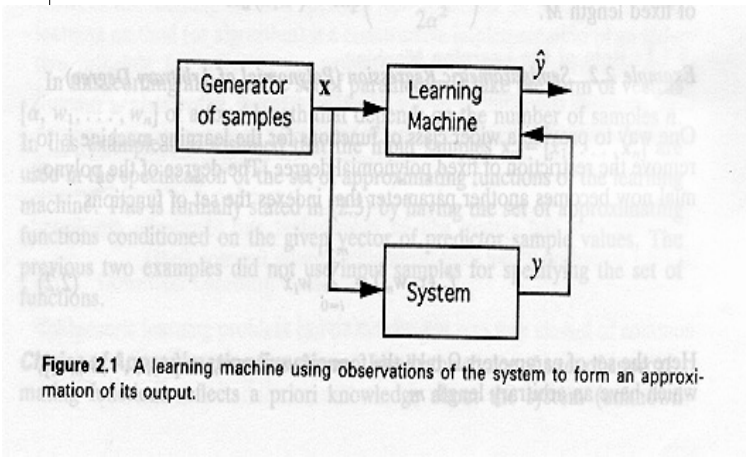


Figure 2.1 A learning machine using observations of the system to form an approximation of its output.

2.1 Problem Formulation

- Generator
  - Produces random vectors $\mathbf{x} \in R^d$ drawn independently from a fixed probability density $\boldsymbol{p}(\mathbf{x})$, which is unknown (*observational*)
  - A modeler has had no control over which input values were supplied to the system

2.1 Problem Formulation

- System
  - Produces an output value y for every input vector $\mathbf{x}$ according to the fixed conditional density $\boldsymbol{p}(y|\mathbf{x})$, which is also unknown.
  - Changing unobserved inputs on the output of system is characterized as random and represented as a prob. distribution

$y = f(\mathbf{x})$: deterministic

$y = f(\mathbf{x}) + \varepsilon$: regression formulation

- Learning Machine
  - Implement a set of functions $f(\mathbf{x}, \omega), \omega \in \Omega$

    where Ω is a set of abstract parameters used only to index the set of functions

Example 2.1

- Parametric Regression (Fixed Degree Polynomial)
  - The set of functions is specified as a polynomial of fixed degree
  - The training data have a single predictor variable

$$f(x, \mathbf{w}) = \sum_{i=0}^{M-1} w_i x^i$$

Ω = { $\mathbf{w} = [w_0, \ldots, w_{M-1}]$ }

M : fixed

Example 2.2

- Semiparametric Regression (Polynomial of Arbitrary Degree)
  - Remove the restriction of fixed polynomial degree
  - Degree becomes another parameter

$$f_m(x, \mathbf{w}_m) = \sum_{i=0}^{m-1} w_i x^i$$

Ω = { $\mathbf{w}_m = [w_0,...,w_{m-1}]$ }
m : arbitrary

Example 2.3

- Nonparametric Regression (Kernel Smoothing)
  - Kernel averaging ($\mathbf{x}_n = [x_1,...,x_n]$)

$$f_\alpha(x, \mathbf{w}_n \mid \mathbf{x}_n) = \frac{\sum_{i=1}^{n} w_i K_\alpha(x, x_i)}{\sum_{i=1}^{n} K_\alpha(x, x_i)}$$

where n is number of samples, $K_\alpha(x, x_i)$ is a kernel function with bandwidth α

Example 2.3:
Nonparametric Regression

- Properties ($\mathbf{x} \in R^p$)
  - ① $K(\mathbf{x}, \mathbf{x'})$ takes on its maximum value when $\mathbf{x'} = \mathbf{x}$
  - ② $|K(\mathbf{x}, \mathbf{x'})|$ decreases with $|\mathbf{x}-\mathbf{x'}|$
  - ③ $K(\mathbf{x}, \mathbf{x'})$ is a function of $2p$ variables
- Radially symmetric: $|\mathbf{x} - \mathbf{x'}|$ to one variable
- Gaussian

$$K_\alpha(x, x') = \exp\left(-\frac{(x-x')^2}{2\alpha^2}\right)$$

Ω takes vector form [α ,w_1,...,w_n])
n : the number of samples

2.1 Problem Formulation

- Choice of Approximating Functions
  - Reflects a priori knowledge about the system
  - Need to incorporate a priori knowledge into the learning method with an already given set of approximating functions
  - Linear in parameters vs. nonlinear in parameters
    - Trigonometric expansion (linear parameterization)

$$f_m(x, \mathbf{v}_m, \mathbf{w}_m) = \sum_{j=1}^{m-1} (v_j \sin(jx) + w_j \cos(jx)) + w_0$$

    - Multilayer network (nonlinear parameterization)

$$f_m(\mathbf{x}, \mathbf{w}, V) = w_0 + \sum_{j=1}^{m} w_j g(v_{0j} + \sum_{i=1}^{d} x_i v_{ij})$$

2.1.1 Role of the Learning Machine

- Selects a function that best approximates the system's response
- Joint probability density function (pdf)

  $\boldsymbol{p}(\mathbf{x}, y) = \boldsymbol{p}(\mathbf{x})\boldsymbol{p}(y|\mathbf{x})$
- Training Data

  $(\mathbf{x}_i, y_i), \quad (i = 1,...,n)$
- Loss function: measures the quality of approximation

- Loss: discrepancy between the output produced by the system and the learning machine for a given point **x**

$$L(y, f(\mathbf{x}, \omega))$$

- Risk functional
  - The expected value of the loss

$$R(\omega) = \int L(y, f(\mathbf{x}, \omega))\, p(\mathbf{x}, y)\, d\mathbf{x}dy$$

- Learning
  - Is the process of estimating the function $f(\mathbf{x}, w_0)$ which minimizes the risk functional over the set of functions supported by the learning machine using training data ($\boldsymbol{p}(\mathbf{x}, y)$ is unknown).
  - Estimates $f(\mathbf{x}, w^*)$, the optimal solution obtained with finite data set using some learning procedure.

- Inductive principle
  - How should a leaning machine use training data?
  - A general prescription for obtaining an estimate $\boldsymbol{f}(\mathbf{x}, w^*)$ of the "true dependency" in the class of approximating functions, from the available (finite) training data.
  - Tell us *what* to do with the data.
- Leaning method
  - Specifies *how* to obtain an estimate.
  - A constructive implementation of an inductive principle for selecting an estimate $\boldsymbol{f}(\mathbf{x}, w^*)$ from a particular set of functions $\boldsymbol{f}(\mathbf{x}, w)$.
  - For a given inductive principle, there are many learning methods corresponding to a different set of functions of a learning machine.

2.1.2 Common Learning Tasks

- Classification
  - Output of learning machine need only take on two values.
  - Loss function: classification error

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

  - Risk functional: Quantifies the probability of misclassification

$$R(\omega) = \int L(y, f(\mathbf{x}, w))\, p(\mathbf{x}, y)\, d\mathbf{x}dy$$

  - Learning: To find the indicator function $\boldsymbol{f}(\mathbf{x}, w_0)$ minimizing the prob. of misclassification.

Regression

- Process of estimating a real-valued function based on a finite set of noisy samples.

$$y = g(\mathbf{x}) + \varepsilon$$

$$g(\mathbf{x}) = \int y p(y \mid \mathbf{x}) dy$$

$$L(y, f(\mathbf{x}, \omega)) = (y - f(\mathbf{x}, \omega))^2$$

$$R(\omega) = \int (y - f(\mathbf{x}, \omega))^2 p(\mathbf{x}, y) d\mathbf{x} dy$$

$$R(\omega) = \int (y - g(\mathbf{x}) + g(\mathbf{x}) - f(\mathbf{x}, \omega))^2 p(\mathbf{x}, y) d\mathbf{x} dy$$

$$= \int (y - g(\mathbf{x}))^2 p(\mathbf{x}, y) d\mathbf{x} dy + \int (f(\mathbf{x}, \omega) - g(\mathbf{x}))^2 p(\mathbf{x}) d\mathbf{x}$$

$$+ 2\int (y - g(\mathbf{x}))(g(\mathbf{x}) - f(\mathbf{x}, \omega)) p(\mathbf{x}, y) d\mathbf{x} dy$$

$$\int (y - g(\mathbf{x}))(g(\mathbf{x}) - f(\mathbf{x}, \omega)) p(\mathbf{x}, y) d\mathbf{x} dy$$

$$= \int \varepsilon (g(\mathbf{x}) - f(\mathbf{x}, \omega)) p(y \mid \mathbf{x}) p(\mathbf{x}) d\mathbf{x} dy$$

$$= \int (g(\mathbf{x}) - f(\mathbf{x}, \omega)) \left[\int \varepsilon p(y \mid \mathbf{x}) dy\right] p(\mathbf{x}) d\mathbf{x}$$

$$= \int (g(\mathbf{x}) - f(\mathbf{x}, \omega)) E_\varepsilon(\varepsilon \mid \mathbf{x}) p(\mathbf{x}) d\mathbf{x} = 0 \quad \because \text{Noise has zero mean}$$

$$R(\omega) = \int (y - g(\mathbf{x}))^2 p(\mathbf{x}, y) d\mathbf{x} dy + \int (f(\mathbf{x}, \omega) - g(\mathbf{x}))^2 p(\mathbf{x}) d\mathbf{x}$$

Since

$$\int (y - g(\mathbf{x}))^2 p(\mathbf{x}, y) d\mathbf{x} dy = \int \varepsilon^2 p(y \mid \mathbf{x}) p(\mathbf{x}) d\mathbf{x} dy$$

$$= \int \left[\int \varepsilon^2 p(y \mid \mathbf{x}) dy\right] p(\mathbf{x}) d\mathbf{x}$$

$$= \int E_\varepsilon(\varepsilon^2 \mid \mathbf{x}) p(\mathbf{x}) d\mathbf{x}$$

: noise variance

We have

$$R(\omega) = \int E_\varepsilon(\varepsilon^2 \mid \mathbf{x}) p(\mathbf{x}) d\mathbf{x} + \int (f(\mathbf{x}, \omega) - g(\mathbf{x}))^2 p(\mathbf{x}) d\mathbf{x}$$

$$R(\omega) = \int E_\varepsilon(\varepsilon^2 \mid \mathbf{x}) p(\mathbf{x}) d\mathbf{x} + \int (f(\mathbf{x}, \omega) - g(\mathbf{x}))^2 p(\mathbf{x}) d\mathbf{x}$$

- Since the first terms(noise variance) does not depend on w, minimizing the second term(function approximation error) is equivalent to minimizing $R(\omega)$
- Thus, obtaining smallest prediction risk is equivalent to the most accurate estimation of the unknown function $g(\mathbf{x})$ by a learning machine

Density Estimation

- Output of the "system" is not used.
- The output of the "learning machine" represents density, i.e. $f(\mathbf{x}, \omega), \omega \in \Omega$ becomes a set of densities
- Loss function $L(f(\mathbf{x}, \omega)) = -\ln f(\mathbf{x}, \omega)$
- Risk functional $R(\omega) = \int -\ln f(\mathbf{x}, \omega) p(\mathbf{x}) d\mathbf{x}$
- Minimizing $R(\omega)$ using the training data $\mathbf{x}_1, \ldots, \mathbf{x}_n$ leads to the density estimation.

Clustering and Vector Quantization

- Optimal partitioning of the unknown distribution in **x**-space into a prespecified number of regions.
- Future samples can be approximated by a single point(cluster center or local prototype).

- The set of vector-valued functions $f(\mathbf{x},\omega),\omega\in\Omega$ are vector quantizers

$$\mathbf{x} \xrightarrow{f(\mathbf{x},\omega)} c(\mathbf{x})$$

Loss function: $L(f(\mathbf{x},w))=(\mathbf{x}-f(\mathbf{x},w))\cdot(\mathbf{x}-f(\mathbf{x},w))$ (squared error distortion)

Risk functional: $R(\omega)=\int(\mathbf{x}-f(\mathbf{x},\omega))\cdot(\mathbf{x}-f(\mathbf{x},w))p(\mathbf{x})d\mathbf{x}$

- Vector quantizer minimizing $R(\omega)$ optimally quantizes future data generated from generated from p(**x**)
- Dimensionality reduction find low-dimensional mappings of a high-dimensional distribution.

Summary

- Loss function
  - Classification

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

  - Regression

$$L(y,f(\mathbf{x},\omega))=(y-f(\mathbf{x},\omega))^2$$

  - Density Estimation

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

  - Clustering and Vector Quantization

$$L(\mathrm{f}(\mathbf{x},\omega))=(\mathbf{x}-\mathrm{f}(\mathbf{x},\omega))\cdot(\mathbf{x}-\mathrm{f}(\mathbf{x},\omega))$$

- Risk Functional
  - Classification

$$R(\omega)=\int L(y,f(\mathbf{x},w))p(\mathbf{x},y)d\mathbf{x}dy$$

  - Regression

$$R(\omega)=\int(y-f(\mathbf{x},\omega))^2 p(\mathbf{x},y)d\mathbf{x}dy$$

  - Density Estimation

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

  - Clustering and Vector Quantization

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

2.1.3 Scope of the Learning Problem Formulation

- Informed part of learning systems
  - Selection of input/output variables
  - Data encoding/representation
  - Incorporating a priori knowledge into the system
- The conceptual range of the formal learning model and the role of the human participant during an informal stage
  - Do preliminary work
  - Define the system, delineate its scope by selecting input variable
  - Choose output based on usefulness and feasibility

2.1.3 Scope of the Learning Problem Formulation

- Have control over the generator for sampling rate or distribution
- Select the most suitable set of functions

- Typical assumptions imposed on the generator distribution
  - Produce independently drawn samples from a fixed probability distribution

    Exception: time series prediction problem(samples generated by a dynamical system)
  - Unchanging generator distribution

    Exception: active learning

2.2 Classical Approaches

- Two parts of the learning problem(classical)
  - ① Specification: determine the parametric form of the unknown underlying distribution
  - ② Estimation: determine parameters that characterize the specified distributions
- Two inductive principles common in classical learning
  - Empirical risk minimization (ERM)
  - Maximum likelihood (ML): a specific form of ERM

2.2.1 Density Estimation

- The classical approach restricts the class of density functions supported by the learning machine to a parametric set.
  - $f(\mathbf{x}, \mathbf{w}), \mathbf{w} \in \Omega$ is a set of densities where $\mathbf{w}$ is an M-dimensional vector (Ω is in R^M, M is fixed)
- Likelihood function
  - Given a set of i.i.d. training data $\mathbf{X} = [\mathbf{x}_1,...,\mathbf{x}_n]$, the probability of $\mathbf{X}$ as a function of $\mathbf{w}$ is

$$P(\mathbf{X} \mid \mathbf{w}) = \prod_{i=1}^{n} f(\mathbf{x}_i, \mathbf{w})$$

- Maximum likelihood inductive principle
  - We should choose the parameters **w** which maximize the likelihood function
  - correspond to choosing a **w***, i.e., model ***f***(**x**, **w***), which is most likely to generate **X**
  - to make problem more tractable *log likelihood function is used*
  - ML risk functional

$$R_{\mathrm{ML}}(\mathbf{w}) = -\sum_{i=1}^{n} \ln f(\mathbf{x}_i, \mathbf{w}) \quad \text{(to be minimized)}$$

- Empirical risk minimization inductive principle
  - empirically estimates the risk functional using training data
  - Empirical risk: average risk for the training data minimized by choosing the appropriate parameters
  - Expected risk(density estimation)

$$R(\mathbf{w}) = \int L(f(\mathbf{x}, \mathbf{w}))\, p(\mathbf{x}) d\mathbf{x}$$

  - This expectation is estimated by taking an average of the risk over the training data

$$R_{emp}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} L(f(\mathbf{x}_i, \mathbf{w}))$$

  - The optimum parameter values **w*** are found by minimizing the empirical risk with respect to **w**
- ERM is more general than ML since it does not specify the particular form of the loss function.
- ERM is equivalent to ML for density estimation, if the loss function is $L(\boldsymbol{f}(\mathbf{x},\mathbf{w})) = -\ln f(\mathbf{x}, \mathbf{w})$

Example 2.4

- ***Estimating the Parameters of the Normal Distribution Using Finite Data***
  - n samples of **x**, denoted by $x_1,\ldots, x_n$, were generated according to the normal distribution (mean μ and variance σ^2 are the two unknown parameters)

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}$$

  - Log likelihood function for this problem is

$$P(\mathbf{X} \mid \mu, \sigma^2) = -\frac{1}{2} n \ln(2\pi) - n \ln(\sigma) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (x_i - \mu)^2$$

- This can be maximized by taking partial derivatives, leading to the estimates

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})^2$$

Example 2.5

▼ ***Mixture of Normals (Vapnik, 1995)***

- The estimation for a more complicated density
- Let n samples of $\boldsymbol{x}$, denoted by $x_1,\ldots,x_n$, be generated according to the distribution

$$p(x) = \frac{1}{2\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\} + \frac{1}{2\sqrt{2\pi}}\exp\left\{-\frac{x^2}{2}\right\}$$

- Log likelihood function

$$P(\mathbf{X}\,|\,\mu,\sigma^2) = \sum_{i=1}^{n}\ln\left(\frac{1}{2\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{(x_i-\mu)^2}{2\sigma^2}\right\} + \frac{1}{2\sqrt{2\pi}}\exp\left\{-\frac{x_i^2}{2}\right\}\right)$$

- We can show that for certain values of μ and σ^2 there does not exist a global maximum,indicating that ML fails to provide a definite solution
  - Specifically, if μ is set to the value of any training data point, then there is no value of σ^2 that gives a global maximum.
  - Evaluate the likelihood for the choice $\mu = x_1$

$$P(\mathbf{X}\,|\,\mu = x_1,\sigma^2) = \ln\left(\frac{1}{2\sqrt{2\pi\sigma^2}} + \frac{1}{2\sqrt{2\pi}}\exp\left\{-\frac{x_i^2}{2}\right\}\right)$$

$$+\sum_{i=2}^{n}\ln\left(\frac{1}{2\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{(x_i-\mu)^2}{2\sigma^2}\right\} + \frac{1}{2\sqrt{2\pi}}\exp\left\{-\frac{x_i^2}{2}\right\}\right)$$

  - Consider a lower bound by assuming that some of the terms take on their minimum values

$$P(\mathbf{X}\,|\,\mu = x_1,\sigma^2) > \ln\left(\frac{1}{2\sqrt{2\pi\sigma^2}} + 0\right) + \sum_{i=2}^{n}\ln\left(0 + \frac{1}{2\sqrt{2\pi}}\exp\left\{-\frac{x_i^2}{2}\right\}\right)$$

$$P(\mathbf{X}\,|\,\mu = x_1,\sigma^2) > -\ln\sigma - \sum_{i=2}^{n}\ln\frac{x_i^2}{2} - n\ln\left(2\sqrt{2\pi}\right)$$

- The lower bound of the likelihood continues to increase for decreasing σ, which means that a global maximum does not exist.

2.2.2 Classification (Discriminant Analysis)

- Classical Classification
  - Conditional densities for each class $p(\mathbf{x}|y=0)$ and $p(\mathbf{x}|y=1)$ are estimated by classical (parametric) density estimation and the ML inductive principle
  - ML estimates (parametric) $p_0(\mathbf{x},\alpha^*)$, $p_1(\mathbf{x},\beta^*)$
  - Prior Probabilities
    - The probability of occurrence of each class
    - $P(y=0)$, $P(y=1)$
  - Posterior Probabilities
    - The probability of observation belonging to each class
    - $p(y=0|\mathbf{x})$,$p(y=1|\mathbf{x})$

- Construct a discriminant rule that describes how an observation x should be classified so as to minimize the prob. of error
- Choose output class having maximum posterior prob.
- Bayes rule is used to calculate the posterior prob. for each class

$$P(y=0\,|\,\mathbf{x}) = \frac{p_0(\mathbf{x},\alpha^*)P(y=0)}{p(\mathbf{x})}$$

$$P(y=1\,|\,\mathbf{x}) = \frac{p_1(\mathbf{x},\beta^*)P(y=1)}{p(\mathbf{x})}$$

- Denominator : normalizing constant, no need to compute since the decision rule is a comparison of relative magnitudes of posterior probabilities

$$p(\mathbf{x}) = p_0(\mathbf{x},\alpha^*)P(y=0) + p_1(\mathbf{x},\beta^*)P(y=1)$$

- Once posterior probabilities are determined, the discriminant function is used to classify **x**:

$$f(\mathbf{x}) = \begin{cases} 0 \text{ if } p_0(\mathbf{x},\alpha^*)P(y=0) > p_1(\mathbf{x},\beta^*)P(y=1) \\ 1 \text{ otherwise} \end{cases}$$

$$f(\mathbf{x}) = I\left\{\ln p_1(\mathbf{x},\beta^*) - \ln p_0(\mathbf{x},\alpha^*) + \ln\frac{P(y=1)}{P(y=0)} > 0\right\}$$

where I( ) is the indicator function that takes 1 if its argument is true and 0 otherwise

- Conditional class densities
  - Determining the parameters α^* and β^* using ML or ERM
  - Apply the ERM indirectly first to estimate the density
  - Use them to formulate the decision rule
- This differs from applying the ERM principle "directly" to minimize the empirical risk

$$R_{emp}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} I(y_i = f(\mathbf{x}_i,\mathbf{w}))$$

- By estimating $R(\omega)$ for classification using average of the risk over the data.

2.2.3 Regression

- Classical formulation of the regression problem
  - Seek to estimate a vector of parameters of an unknown function $\boldsymbol{f}(\mathbf{x}, \mathbf{w}_0)$ by making measurements of the function with error at any point $\mathbf{x}_i$

$$y_i = f(\mathbf{x}_i, \mathbf{w}_0) + \varepsilon_i$$

    where error is independent of $\mathbf{x}$ and is distributed according to a known density $p_\varepsilon(\varepsilon)$.
  - On data $\mathbf{Z} = \{(\mathbf{x}_i, y_i), i = 1,...,n\}$, likelihood is

$$P(\mathbf{Z} \mid \mathbf{w}) = \sum_{i=1}^{n} \ln p_\varepsilon(y_i - f(\mathbf{x}_i, \mathbf{w}))$$

- Likelihood for Gaussian error with zero mean, and fixed variance σ

$$P(\mathbf{Z} \mid \mathbf{w}) = -\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i, \mathbf{w}))^2 - n \ln(\sqrt{2\pi}\sigma)$$

- Maximizing the likelihood is equivalent to minimizing functional

$$R_{emp}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i, \mathbf{w}))^2$$

  - which is the risk functional obtained by ERM for the squared from function.
- Squared loss function
  - Appropriate only for gaussian noise.
  - Often used in practical application where the noise is not gaussian.

2.2.4 Stochastic Approximation

- Stochastic Approximation (Robbins and Monroe, 1951)
  - Parameters in an approximating function are estimated sequentially.
  - For each individual data sample presented, a new parameter estimate is produced.
  - As samples becomes large, risk converges to minimum.
  - General expected risk functional

$$R(\omega) = \int L(\mathbf{z}, \omega)\, p(\mathbf{z})\, d\mathbf{z}$$

- stochastic approximation procedure

$$\omega(k+1) = \omega(k) - \gamma_k \, grad_\omega L(\mathbf{z}_k, \omega(k)), \quad k = 1,...n,...$$

  (where $\mathbf{z}_1,...,\mathbf{z}_n$ is sequence of data samples)
  - General condition

$$\lim_{k \to \infty} \gamma_k = 0, \quad \sum_{k=1}^{\infty} \gamma_k = \infty, \quad \sum_{k=1}^{\infty} \gamma_k^2 < \infty$$

- Initial motivation
  - Generate parameter estimates in a “real-time” fashion as data are collected
- Benefits
  - Large amount data need not be stored at one time.
  - Adapt to slowly changing data-generating system.

- Recycling (each cycle = epoch)
  - Stored batch of data is presented sequentially.
  - Produces an asymptotically large training sequence.
  - Computationally less complicated than batch one.
- When to stop the updating process
  - Monitor the gradient for each presented samples
    - Check threshold
    - Obey ERM (not used if learning is halted early before small gradients are seen)

2.2.5 Solving Problems with Finite Data

- "Do not attempt to solve a specified problem by indirectly solving a harder general problem as an intermediate step"
  - More general learning problem, larger samples required
  - Specific task should be solved directly
- Estimate features of the density that are critical for solving our particular problem.

Example 2.6

- ***Discriminant Analysis***
  - Build a two-class classifier from data
    - data are generated according to the multivariate normal probability distribution $N(\mu_0, \Sigma_0)$ and $N(\mu_1, \Sigma_1)$
    - estimate $\mu_0, \mu_1, \Sigma_0, \Sigma_1$ using ML based on the training data
    - optimal decision rule

$$f(\mathbf{x}) = I\left\{\frac{1}{2}(\mathbf{x}-\mu_0)^T\sum_0^{-1}(\mathbf{x}-\mu_0) - \frac{1}{2}(\mathbf{x}-\mu_1)^T\sum_1^{-1}(\mathbf{x}-\mu_1) + c > 0\right\}$$

$$c = \ln\frac{\det(\Sigma_0)}{\det(\Sigma_1)} - \ln\frac{P(y=1)}{P(y=0)}$$

- Solution to not enough data
  - To impose artificial constraint $\Sigma_0 = \Sigma_1 = \Sigma$
  - Linear decision rule

$$f(\mathbf{x}) = I\left\{(\mu_0-\mu_1)^T\Sigma^{-1}\mathbf{x} + \frac{1}{2}(\mu_1^T\Sigma^{-1}\mu_1) - \frac{1}{2}(\mu_0^T\Sigma^{-1}\mu_0) - \ln\frac{P(y=0)}{P(y=1)} > 0\right\}$$

    - estimate two means and only one covariance matrix
  - The simpler linear decision rule often performs better than the quadratic decision rule even when $\Sigma_0 \neq \Sigma_1$
- Demonstration
  - 20 data samples (10 per class), according to

Class zero: $N\left([0,0], \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right)$ Class one: $N\left([2,0], \begin{bmatrix}1 & 0.5\\ 0.5 & 1\end{bmatrix}\right)$

  - Class density is gausian, means and covariance matrices are unknown
  - Quadratic decision rule vs. linear decision rule
  - Linear decision rule does not match underlying class distribution
  - First-order model provides the lowest classification error (Fig. 2.2)

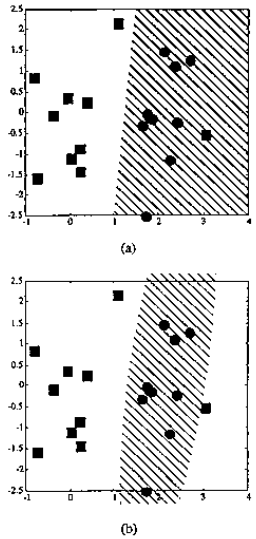


Figure 2.2 Discriminant analysis using finite data. (a) The linear decision rule has an accuracy rate of 83%. (b) The quadratic decision rule has an accuracy of 77% (note that the parabolic decision boundary has been truncated in the plot). Out of 100 repetitions of the experiment, the linear decision boundary is better than the quadratic 73% of the time.

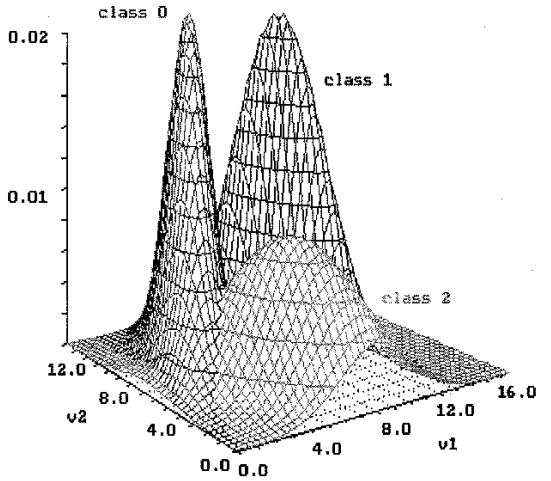


Figure 4.6. Three-dimensional plot of three class-specific normal distribution functions in form of $\max_k[P_k N(\mathbf{v}, \boldsymbol{\mu}_k, \mathbf{K}_k)]$ as defined by example of (4.35). Vertical axis is scaled to represent prob($\mathbf{v}$, k).

2.2.6 Nonparametric Methods

- ▼ Development of nonparametric method
  - Attempt to deal with the main shortcoming of classical techniques
    - Specify the parametric form of unknown distributions and dependencies
  - Require few assumptions but large number of samples
- ▼ Nonparametric Density Estimation
  - Histogram
    - Divide sample space into bins of constant width
    - Determine number of samples falling into each bin (Fig. 2.3)

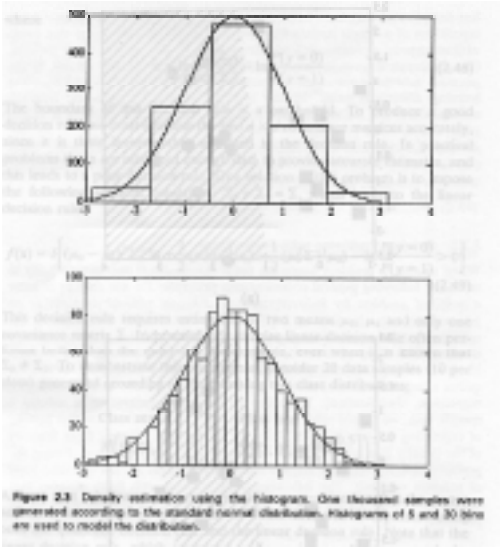


Figure 2.3 Density estimation using the histogram. One thousand samples were generated according to the standard normal distribution. Histograms of 5 and 30 bins are used to model the distribution.

- Drawback : resulting density is discontinuous

- Sliding window kernel function
  - Results in a smooth estimate

- General principle for nonparametric density estimator.
  - Solving the integral equation defining the density

$$\int_{-\infty}^{x} p(u)\,du = F(x)$$

(where F(x) is cumulative distribution function: cdf)

  - Since the cdf is unknown, it is approximated by the empirical cdf (converge to true cdf as samples tends to infinity)

$$F_n(x) = \sum_{i=1}^{n} I(x \geq x_i)$$

  - Cannot be solved by straightforward since the empirical cdf has discontinuities.

  - An approach to finding the continuous solution to the density
    - Replace the Dirac function with a continuous function
  - approximate the density as a sum of kernel functions at each data point

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} K_\alpha(x, x_i)$$

(where $K_\alpha(x, x')$ is a kernel function defined pp19)

- Drawbacks
  - poor scaling properties for high-dimensional data
  - a volume enclosing enough data point is not local any more
  - radius of volume can be a significant fraction of total range of data
- Asymptotic assumptions (classical nonparametric methods)
  - not designed for small number of samples
  - poor results in practical situations with limited data

2.3 Adaptive Learning: Concepts and Inductive Principles

▼ Flexible (or adaptive) learning methods

- "Flexibility"
  - A method's capability to estimate arbitrary dependencies from finite data
- Previous problem
  - Parametric methods
    - Impose very stringent assumption
    - Are likely to fail if the true parametric form of a dependency is not known
  - Classical nonparametric methods
    - Fail for high-dimensional problems with finite samples

- Universal approximation property
  - Use flexible (very wide) class of approximating functions
  - Approximate any continuous function with a prespecified accuracy
- Finiteness of available data
  - Set of functions needs to be constrained to produce a unique solution
- Inductive principle
  - Provide a framework for selecting a unique solution from a wide class of functions using finite data

2.3.1 Philosophy, Major Concepts, and Issues

- Two steps in predictive learning (Fig. 2.4)
  - ① Leaning unknown dependency from samples (Induction)
  - ② Using dependency estimated in (1) to predict output for future input values (Deduction)
- *Transductive* approach
  - Estimate outputs of unknown function for several points of interest *directly* from the training data
- *Local Estimation*
  - A special case of transduction
  - The prediction is made at a *single* point.
  - Lead to local risk formulation

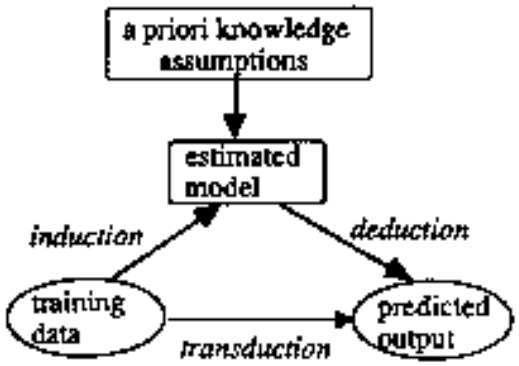


Figure 2.4 Two types of inference: induction-deduction and transduction.

- Difference between transduction and local estimation
  - Assume that the transduction refers to predictions at two or more input values simultaneously.
- Predictive learning
  - First step: challenging one
    - Its solution requires a priori knowledge in addition to data
  - Second step: simply calculating the value of function
- All learning methods use a priori knowledge in the form of the class of approximating functions of a learning machine
  - Parametric Method
    - Use a very restricted set of approximating functions of prespecified parametric form

- Adaptive Methods
  - Additional a priori knowledge is needed for imposing additional constraints (penalty) on a potential of a function to be a solution to the learning problem
  - Two types of a priori knowledge
    1. Choosing a (wide, flexible) set of approximating functions of a learning machine
    2. Imposing additional constraints on the functions within this set
  - Second type to “a priori knowledge”

- Require followings to form a unique generalization (model) from finite data
  1. A set of approximating functions.
  2. A priori knowledge for constraints
  3. An inductive principle for combining a priori knowledge with data
  4. A learning method of inductive principle for a given class of approximating functions
- Inductive Principle vs. Learning Methods
  - For a given inductive principle, there may be many learning methods, corresponding to different classes of approximating functions and/or different optimization techniques

2.3.2 A Priori Knowledge and Model Complexity

- No more things should be presumed to exist than are absolutely necessary
  - “Occam’s razor” principle attributed to W. Occam c. 1280-1349.
- General belief
  - For adaptive learning methods with finite samples, the best prediction performance is provided by a model of optimum complexity

- Model Selection
  - Seek simpler models over complex ones.
  - Optimize the trade-off between model complexity and the accuracy of model’s description (fit) to the training data.
- Model Complexity
  - Usually controlled by a priori knowledge
- A Priori Knowledge
  - Cannot assume a model of *fixed* complexity.
  - Should not be automatically used for predictive learning with finite samples even if the true parametric form of a model is known a priori.

Example 2.7

- ***Parametric Estimation for Finite Data***
- Parametric regression problem
  - 10 data points are generated according to

$$y = x^2 + \varepsilon$$

where the noise is gaussian with zero mean and variance $\sigma^2 = 0.25$, quantity x has uniform distribution [0, 1].

  - Assume
    - Known: a polynomial of second order has generated the data
    - Unknown: the coefficients of the polynomial

- First-order polynomial vs. second-order polynomial
  - First-order polynomial provides the lowest risk (Fig. 2.5)

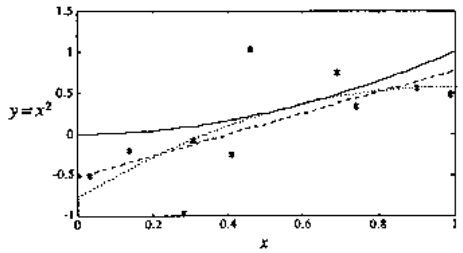


Figure 2.5 For finite data, limiting model complexity is more important than using true assumptions. The solid curve is the true function, the asterisks are data points with noise, the dashed line is a first-order model (mse = 0.0596), the dotted curve is a second-order model (mse = 0.0845).

  - Remark:
    - For finite data it is not the validity of the assumptions but the complexity of the model that determines prediction accuracy.

- Two Conclusions

① An optimal trade-off between the model complexity and available (*finite*) data are important even when the parametric form of the model is *known.*
  - 500 training data -> second-order polynomial
  - 5 samples -> a mean estimate (zero-order polynomial)

② A priori knowledge can be useful for learning predictive models only if it controls (explicitly or implicitly) the *model complexity.*
  - Two issues
    - How to define and measure the model complexity?
    - How to provide “good” parameterization for a family of approximating functions of a learning machine?

2.3.3 Inductive Principles

- Classical modeling
  - Model is given first and parameters are estimated from data using empirical risk minimization inductive principle.
- Adaptive modeling
  - Underlying model is not known, and it is estimated using a large number of candidate models.
  - Main issue:
    - Choosing the candidate model of the right complexity to describe the training data, as stated by Occam’s razor.

- Inductive Principles
  - Provide different quantitative formulation of Occam's principle
  - Differ in terms of
    - Representation (encoding) of a priori knowledge
    - Applicability when the true model does not belong to the set of approximating functions
    - Mechanism for combining a priori knowledge with training data
    - Availability of constructive procedures for a given principle

- Five inductive principles in review
  - Penalizatino(regularization) inductive principle
  - Early stopping rules
  - Structural risk minimization(SRM)
  - Bayesian inference
  - Minimum Description Length(MDL)

Penalization (Regularization) Inductive Principle

- Assumes a
  - Flexible class of approximating functions $f(\mathbf{x}, \mathrm{w}), \mathrm{w} \in \Omega$ where Ω is a set of abstract parameters.
- Penalization (regularization) term is used to restrict solutions

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

  - $R_{emp}(\omega)$: usual empirical risk
  - $\Phi[f(\mathbf{x}, \omega)]$: penality(nonnegative, independent of data)
  - $\lambda(> 0)$: regularization parameter
  - A priori knowledge is included in penalty form.

- Regularization parameter λ
  - Controls the strength of priori knowledge
  - Very large λ -> result of minimizing $R_{emp}(\omega)$ does not depend on data
  - Small λ -> final model does not depend on penalty functional
  - Optimal value of λ is chosen using resampling methods
  - Optimal model estimate is found as a result of a trade-off between fitting the data and a priori knowledge (i.e., a penalty term).

Early Stopping Rules

- Stopping Rules
  - A heuristic inductive principle often used in the applications of neural networks.
  - Avoid overfitting with overparameterized models.
  - Can be interpreted as an implicit form of penalization
    - A penalty is defined on a path corresponding to the successive model estimates obtained during gradient-descent training.
    - Solutions are penalized according to the number of gradient descent steps taken along the path, i.e., the distance from the starting point.
    - Difficult to control and interpret “penalization” via early stopping rule (Friedman, 1994)

Structural Risk Minimization (SRM)

- Risk Minimization
  - Approximating functions are ordered according to their complexity, forming a *nested structure*

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

  - Example: Class of polynomial approximating functions
    - The elements of a structure are polynomials of a given degree.
    - Polynomials of degree *m* are a subset of polynomials of degree (*m* + 1).
  - Goal of learning
    - Choose an optimal element of a structure (i.e., polynomial degree), and estimate its coefficients from a given training set.

- Complexity
  - Number of free parameters (linear functions)
  - VC-dimension (nonlinear functions)
- Optimal Choice of Model Complexity
  - Provides the minimum of the expected risk.
  - Statistical learning theory provides analytic upper-bound estimates for expected risk.
  - These estimates are used for model selection, choosing an optimal element of a structure under the SRM inductive principle.

Bayesian Inference

- Bayesian Inference
  - Uses additional a priori information about approximating functions.
    - Obtain a unique predictive model from finite data.
  - Prior Probability Distribution
    - Probability of any function being the true function
    - Reflects *subjective* degree of belief
  - Provides an effective way of encoding prior knowledge.
  - Based on the classical Bayes formula for updating prior probability using the evidence provided by the data:

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

Where

P[model]: *Prior* probability (before the data are observed)

P[data]: The probability of observing training data

P[model|data]: *A posterior* probability of a model given the data

P[data|model]: The probability that the data are generated by a model also known as the *likelihood*

- Density estimation
  - $\boldsymbol{f}(\mathbf{x}, \mathbf{w}), \quad \mathbf{w} \in \Omega$ is a set of densities
    - $\mathbf{w}$: m-dimensional vector of "free" parameters (m is fixed)
    - $\boldsymbol{f}(\mathbf{x}, \mathbf{w}_0)$ belongs to this class
    - $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]$
  - Probability of seeing $\mathbf{X}$ as a function of $\mathbf{w}$

$$P[\text{data} \mid \text{model}] = P(\mathbf{W} \mid \mathbf{w}) = \prod_{i=1}^{n} f(\mathbf{x}_i, \mathbf{w})$$

- A priori density function

$$P[\text{model}] = p(\mathbf{w})$$

- Bayes formula

$$P(\mathbf{w} \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid \mathbf{w})\, p(\mathbf{w})}{P(\mathbf{X})}$$

- Prior distribution
  - is taken rather broadly, reflecting general uncertainty about "correct" parameter values
- Posterior distribution
  - converted from prior probability after having observed the data
  - more narrow
  - be consistent with the observed data

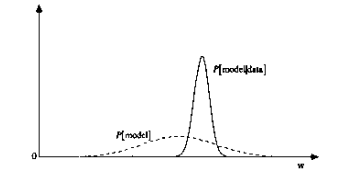


Figure 2.6 After observing the data, the wide prior distribution is converted into the more narrow posterior distribution using Bayes rule.

- Two Bayesian approaches for obtaining an estimate of unknown p.d.f.
  - *True Bayesian approach*
    - Average over all possible models

$$\Theta(\mathrm{x} \mid \mathbf{X}) = \int f(\mathbf{x}, \mathbf{w})\, p(\mathbf{w} \mid \mathbf{X})\, \mathrm{d}\,\mathbf{w}$$

where $\boldsymbol{p}(\mathbf{w}|\mathbf{X})$ is given by the Bayes formula

- *Marginalization* : involves integrating out redundant variables
- Final model is a weighted sum of all possible predictive models
  - with weights given by the evidence that each model is correct
- Multidimensional integration presents a challenging problem
- Standard numerical integration is impossible
- When gaussian assumptions do not hold, various forms of random sampling, such as Monte Carlo methods, have been proposed

- MAP approach
  - Choose an estimate $\boldsymbol{f}(\mathbf{x}, \mathbf{w}^*)$ maximizing posterior probability $\boldsymbol{p}(\mathbf{w}|\mathbf{X})$
    - Maximum a posterior probability (MAP) estimate
  - Equivalent to the penalization formulation as shown below
  - Regression formulation
    - Training data $(\mathbf{x}_i, y_i)$ generated according to

$$y = f(\mathbf{x}, \mathrm{w}_0) + \varepsilon$$

- $\mathbf{Z} = [\mathbf{X}, \mathbf{y}]$, where $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]$ and $\mathbf{y} = [y_1, ..., y_n]$
- $p(\mathbf{w})$ is a priori density

$$p(\mathbf{w} \mid Z) = \frac{P(\mathrm{Z} \mid \mathbf{w})\, p(\mathbf{w})}{P(\mathrm{Z})}$$

$$p(Z \mid \mathbf{w}) = \prod_{i=1}^{n} p(\mathbf{x}_i, y_i) = P(\mathbf{X}) \prod_{i=1}^{n} p(y_i - f(\mathbf{x}_i, \mathbf{w}))$$

$$R_{\mathrm{map}}(\mathbf{w}) = \sum \ln p(y_i - f(\mathbf{x}_i, \mathbf{w})) + \ln p(\mathbf{w})$$

$$\varepsilon_i = y_i - f(\mathbf{x}_i, \mathrm{w}_0) \sim N(0, \sigma^2)$$

$$\ln p(y_i - f(\mathbf{x}_i, \mathbf{w})) = (y_i - f(\mathbf{x}_i, \mathbf{w}))^2$$

$$R_{\mathrm{map}}(\mathbf{w}) = \frac{1}{n} \sum \ln (y_i - f(\mathbf{x}_i, \mathbf{w}))^2 + \frac{2\sigma^2}{n} \ln p(\mathbf{w})$$

- Equivalent to penalization formulation with regularization parameter (reflecting the knowledge of noise variance)
  - Estimating noisy variance == Estimating regularization parameter
  - Choice of a penalty term <-> priori information (distribution)
  - Choice of the regularization parameter <-> knowledge (estimate) of the amount of noise
- Choosing the value of regularization parameter(Finding a “good” prior)
  - Tailor priors to the data
    - Type II maximum likelihood technique
    - Contradicts the original notion of data-independent prior knowledge

Comparison of Bayesian and Penalization Methods

- Bayes: To encode a priori knowledge about multiple, general, user-defined characteristics of the target function.
- Penalization: To perform complexity control by encoding a priori knowledge about function smoothness in terms of a penalty functional.
- Bayesian model selection:
  - Penalize more complex models in choosing the model
  - Not guarantee the best generation performance
- Penalization and SRM: explicit minimization: explicit minimization of the prediction risk

Bayesian Model Comparison

- Two models: MDLs with a different no. of hidden units.

  $M_1 = \boldsymbol{f}_1(\mathbf{x}, \mathbf{w}_1)$ and $M_2 = \boldsymbol{f}_2(\mathbf{x}, \mathbf{w}_2)$
- Problem: choose the best model to describe a given data set $\mathbf{Z}$
- Estimate relative plausibilities of two models using Bayes factor

$$\frac{P(M_1 \mid \mathrm{Z})}{P(M_2 \mid \mathrm{Z})} = \frac{P(\mathrm{Z} \mid M_1) P(M_1)}{P(\mathrm{Z} \mid M_2) P(M_2)}$$

- Evidence of model M:

$$P(\mathrm{Z} \mid M_i) = \int P(\mathrm{Z}, \mathrm{w}_i \mid M_i)\, d\,\mathrm{w}_i = \int P(\mathrm{Z} \mid \mathrm{w}_i, M_i)\, p(\mathrm{w}_i \mid M_i)\, d\,\mathrm{w}_i$$

Minimum Description Length (MDL)

- MDL principle
  - Based on the information-theoretic analysis of the randomness concept
  - Regards *models as codes*, as encodings of training data
  - Main idea
    - Any data set can be appropriately encoded, and its *code length* represents an inherent property of the data which is indirectly related to the generalization capability of the model (i.e., code)
  - Characteristics of randomness of a data set
    - Be the shortest binary code describing the data
    - Related to the length of the binary code

- MDL inductive principle
  - Training data set: $(\mathbf{x}_i, y_i)$, $i = 1,..., n$
  - $y = \{0, 1\}$, $\mathbf{x}$ is d-dimensional feature vector
  - Problem
    - Given a data object $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_n)$, is a binary string $y_1, ..., y_n$ random?
  - Encode the output string y by a possibly shorter code
    - Model: code length L(model)
    - Error term: code length L(data | model)
    - Total length: l

      $l = L(\text{model}) + L(\text{data} \mid \text{model})$

      Coefficient of compression:

      $$K(\text{model}) = \frac{1}{n}$$

    - Small coefficient -> string is not random

$$\mathbf{y} = T(\mathbf{X})$$

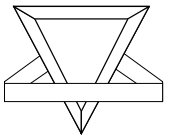
$$L(\text{model}) = \lceil \log_2 m \rceil$$

$$L(\text{data} \mid \text{model}) = \lfloor \log_2 C_n^e \rfloor + \lceil \log_2 e \rceil + \lceil 2 \log_2 \log_2 e \rceil$$

$$R(T) < 2\left(K(T) \ln 2 - \frac{\ln \eta}{n} \right)$$

- Summarizing Properties of Inductive Principle

	Penalization	SRM	Bayes	MDL
Representation of a priori knowledge or complexity	Penalty term	Structure	Prior distribution	Codebook
Constructive procedure for complexity control	Minimum of penalized risk	Optimum element of a structure	A posteriori distribution	Not defined
Methods for model selection	Resampling	Analytic bound on prediction risk	Marginalization	Minimum code length
Applicability when the true model does not belong to the set of approximating functions	Yes	Yes	No	Yes



	Large Sample Size	Small (Finite) Sample Size
Restricted class of functions	Parametric	Parametric (may not work well)
Wide class of functions (universal approximators)	Classical nonparametric	Adaptive (flexible)