MACHINE LEARNING

Jen-Tzung Chien

National Chiao Tung University

CONTENTS

Introduction

Book web site http://research.microsoft.com/~cmbishop/PRML

- Pattern Recognition is concerned with automatic discovery of regularities in data through the use of computer algorithms and with the use of regularities to take actions such as classifying the data into different categories, e.g. hand-written recognition
- Training set $\mathrm{X} = \{x_1, \ldots, x_N\}$, target vector *t*, supervised learning using $\{x, t\}$
- In machine learning, we are finding $y(x)$ and decoding x as

  target vectors ⇒ Training phase/Learning phase

  Testing phase → generalization (insufficient training data)

  Preprocessing → feature extraction, e.g. face detection in video stream, fast to compute, preserve discriminatory information

  → dimension reduction is important

Introduction

- Classification: Input vector → one of discrete categories

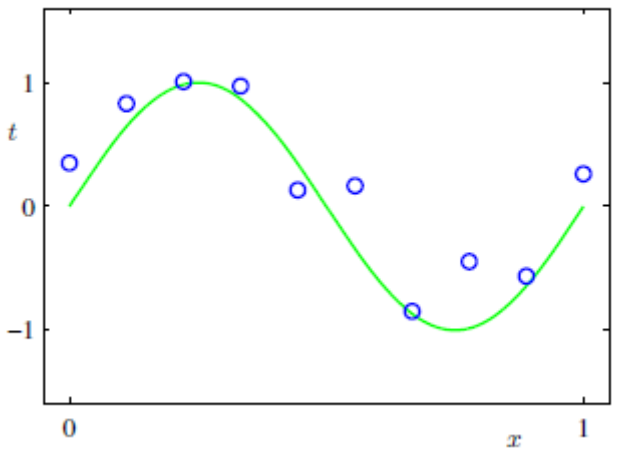
  Regression: Input vector → one or more continuous variables
- Unsupervised learning (without target values)

  clustering: discovering groups of similar samples

  density estimation: determine distribution of data within input space.

An Example of Regression Problem

Our goal is to predict the value of *t* for some new value of *x*, without knowledge of the curve.



- $x = (x_1, \ldots, x_N)^T$
- $t = (t_1, \ldots, t_N)^T$
- $\sin(2\pi x) +$ Gaussian random noise

Polynomial Curve Fitting

$\hat{x}(new\ value) \rightarrow \hat{t}$

make predictions that are optimal according to appropriate criteria

polynomial function of the form

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

nonlinear function of x , $\boldsymbol{w} = (\omega_0, \omega_1 \ldots, \omega_M)^T$

linear function of **w** (parameters) → Linear Model

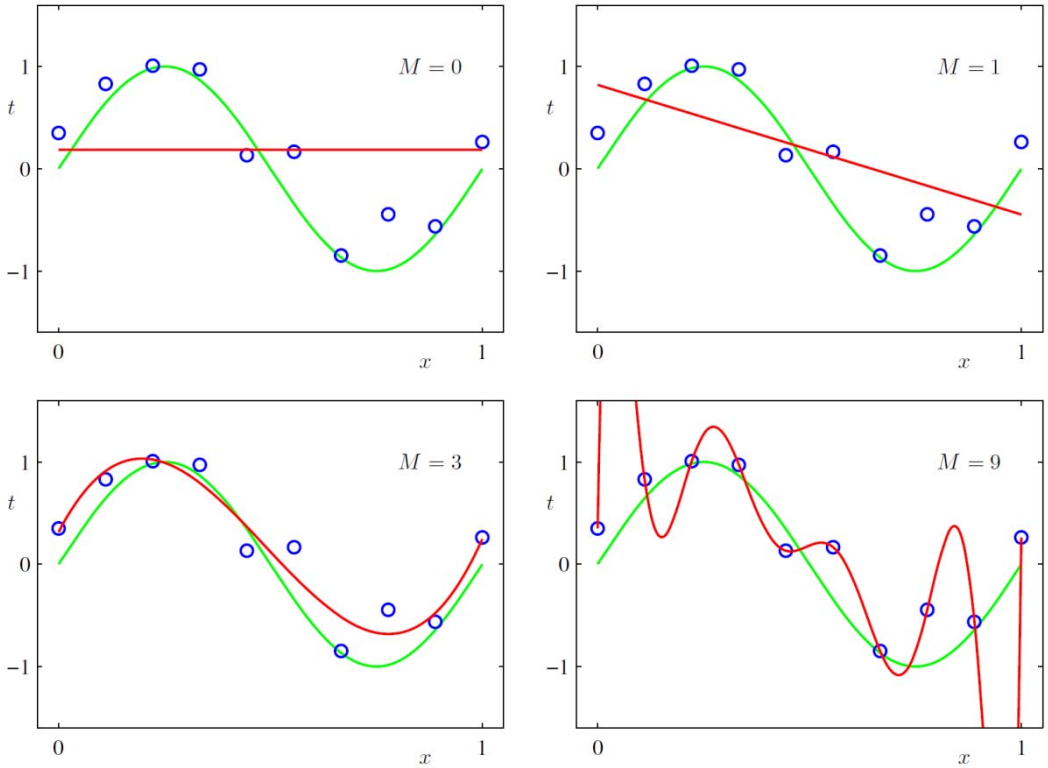
Least squares solution → sum-of-squares error function

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, E(\boldsymbol{w})$$

Selection of Order *M* Is Important

- Issue: Model comparison or Model selection



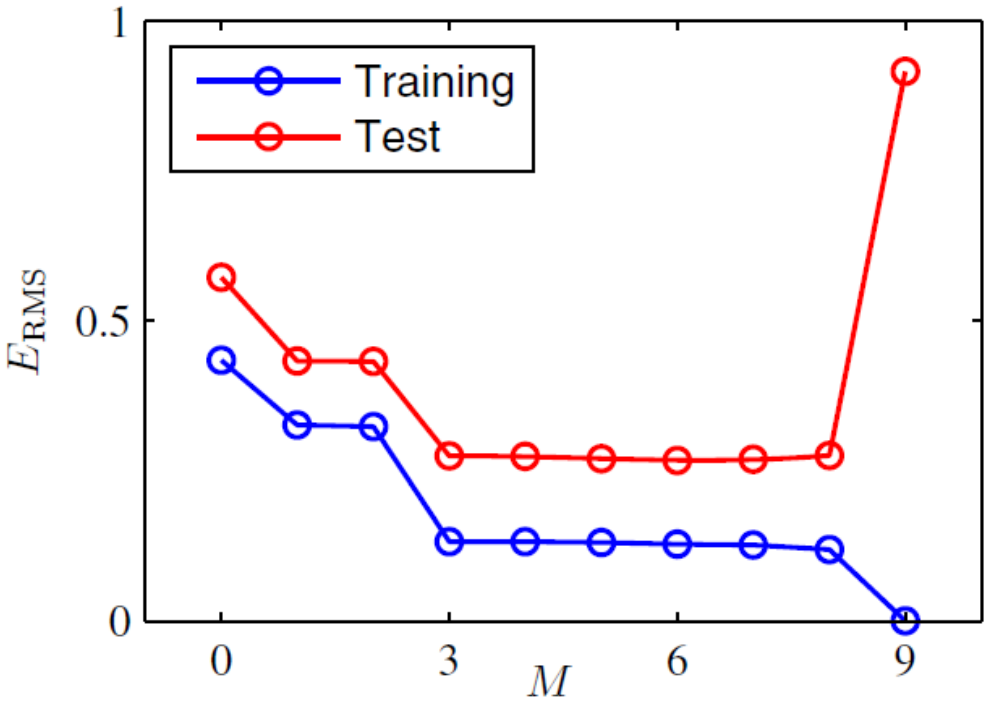
under estimation → over-estimation

over-fitting

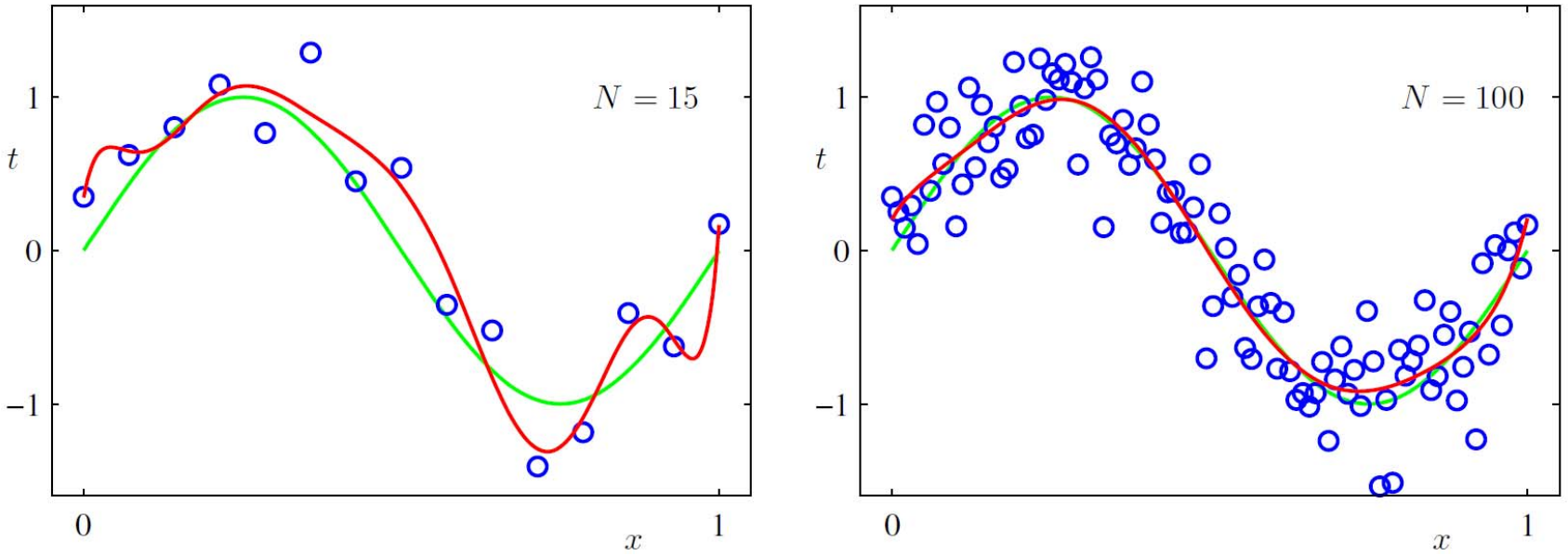
Root-Mean-Square Error vs. Model Order M

- Root-mean-square(RMS) error

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$$



Polynomial Curve Fitting under M=9



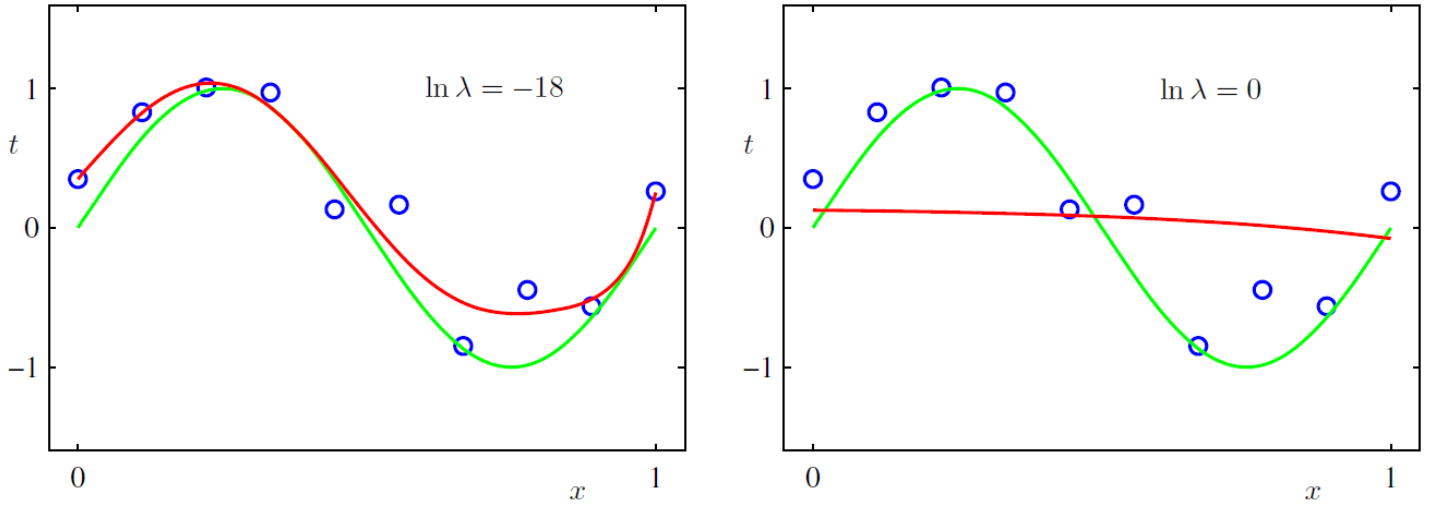
- Maximum likelihood is easy to suffer over-fitting problem. Bayesian approach is helpful for this issue.
- Using Bayesian model, the effective number of parameters adapts automatically to the size of data set.

Regularization

- One technique can be used to control the over-fitting problem that of '*regularization*'.
- *A* penalty term is involved in objective function → modified error parameter

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

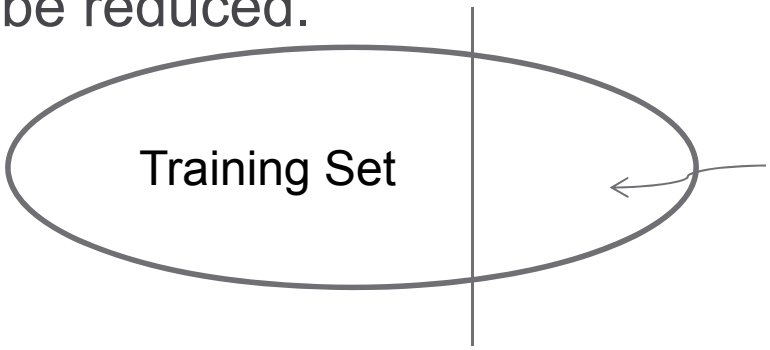
- Weight decay approach (similar to the one used in neural network)



(M = 9)

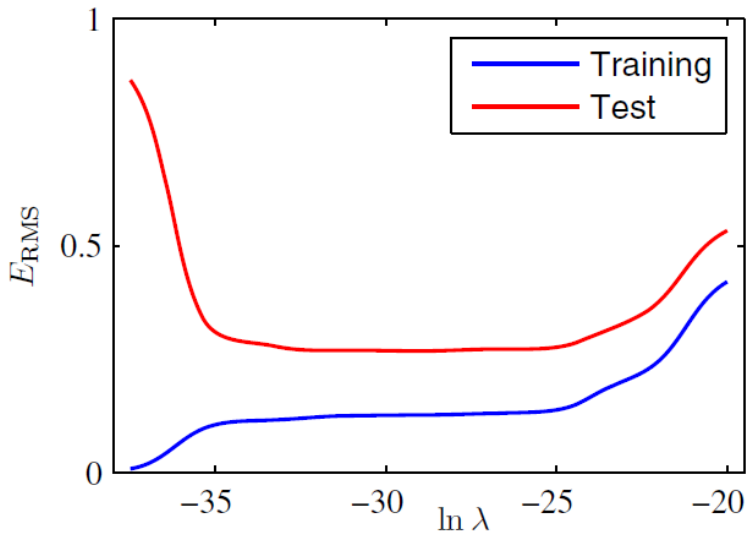
Cross Validation

- *M* can be reduced.



Validation set
held-out (correct) set
is used to find either *M* or λ.

partition



$M = 9$

Bayesian Probabilities

- See biography of Thomas Bayes

  classical interpretation of probability
- Bayesian view → quantification of uncertainty
- Probability theory is regarded as an extension of Boolean logic for situations involving uncertainty
- We are addressing the uncertainty of parameter W

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})} \qquad \mathcal{D} = \{t_1, \ldots, t_N\}$$

$$\text{posterior} \propto \text{likelihood} \times \text{prior}$$

$$p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{w})p(\mathbf{w})\,\mathrm{d}\mathbf{w}$$

- Maximum likelihood $w_{\mathrm{ML}} = \underset{w}{\operatorname{argmax}} \log P(D|w)$

$-\log p(D|w) \rightarrow$ error function

Gaussian Distribution

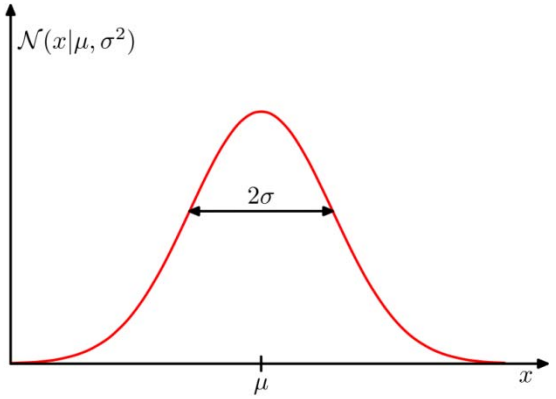
$$\mathcal{N}\left(x|\mu,\sigma^2\right) = \frac{1}{(2\pi\sigma^2)^{1/2}}\exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\} \quad > 0$$

$$\int_{-\infty}^{\infty}\mathcal{N}\left(x|\mu,\sigma^2\right)\,\mathrm{d}x = 1 \qquad \mathbb{E}[x] = \mu \quad \mathbb{E}[x^2] = \mu^2 + \sigma^2$$

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

$x \in R^D$

$= (x_1,\ldots,x_D)^{\mathrm{T}}$



i.i.d. independent and identically distributed

Let $\mathbf{x} = (x_1,\ldots,x_N)^{\mathrm{T}}$ is i.i.d.

$$p(\mathbf{x}|\mu,\sigma^2) = \prod_{n=1}^{N}\mathcal{N}\left(x_n|\mu,\sigma^2\right)$$

$$\ln p\left(\mathbf{x}|\mu,\sigma^2\right) = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2 - \frac{N}{2}\ln\sigma^2 - \frac{N}{2}\ln(2\pi)$$

Maximum Likelihood Estimation

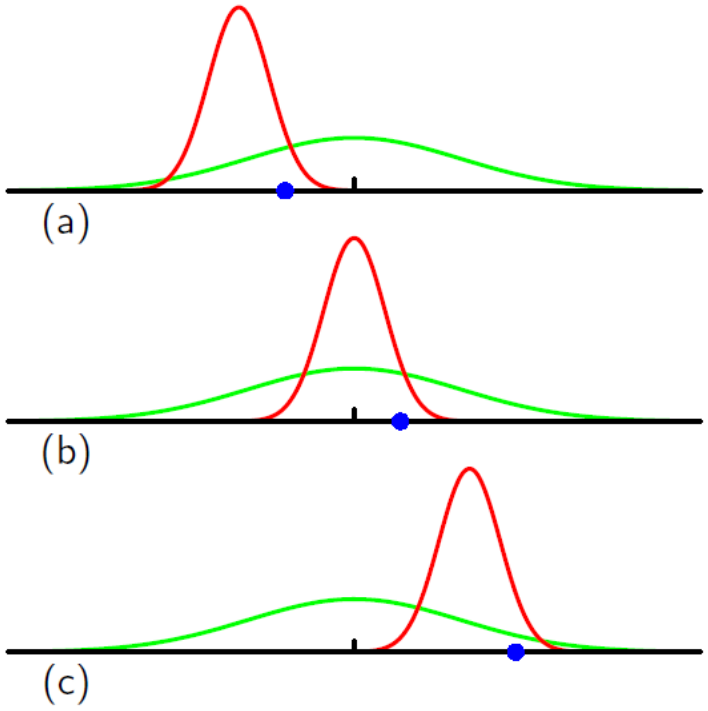
$$\frac{\partial \ln p(\mathbf{X}|\mu,\sigma^2)}{\partial \mu} = 0 \quad \Longrightarrow \quad \mu_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N} x_n$$

$$\frac{\partial \ln p(\mathbf{X}|\mu,\sigma^2)}{\partial \sigma^2} = 0 \quad \Longrightarrow \quad \sigma^2_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N} (x_n - \mu_{\mathrm{ML}})^2$$

- ML underestimates the variance σ^2 → bias

$$\mathbb{E}[\mu_{\mathrm{ML}}] = \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}\{x_n\} = \mu$$

$$\mathbb{E}[\sigma^2_{\mathrm{ML}}] = \left(\frac{N-1}{N}\right)\sigma^2$$



- Averaged across the three data sets, the mean is correct .

  Variance is under-estimated

$$\lim_{N\to\infty} \mathbb{E}[\sigma^2_{\mathrm{ML}}] = \sigma^2$$

- For unbiased variance parameter,

$$\widetilde{\sigma}^2 = \frac{N}{N-1}\sigma^2_{\mathrm{ML}} = \frac{1}{N-1}\sum_{n=1}^{N} (x_n - \mu_{\mathrm{ML}})^2$$

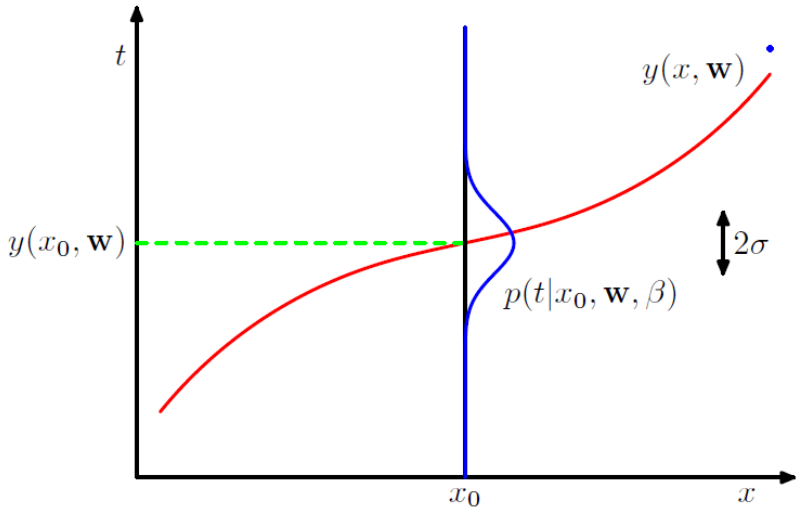
Curve Fitting Re-visited

Given $\mathbf{x} = (x_1, \ldots, x_N)^{\mathrm{T}}$ $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$

$$p(t|x, \mathbf{w}, \beta) = \mathcal{N}\left(t|y(x, \mathbf{w}), \beta^{-1}\right)$$

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}\left(t_n|y(x_n, \mathbf{w}), \beta^{-1}\right)$$

$$\ln p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = -\frac{\beta}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi)$$



- Maximum likelihood ↔ *sum-of-squares error function*

$$\frac{\partial \ln p(t|\mathbf{X}, w, \beta)}{\partial \beta} = 0 \quad \Longrightarrow \quad \frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}_{\mathrm{ML}}) - t_n\}^2$$

→ $\mathbf{w}_{\mathrm{ML}}$ → $p(t|x, \mathbf{w}_{\mathrm{ML}}, \beta_{\mathrm{ML}})$ point estimate

Maximum *a Posteriori* Estimation

- Bayesian approach considering a prior distribution over *W*

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}) = \left(\frac{\alpha}{2\pi}\right)^{(M+1)/2} \exp\left\{-\frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}\right\}$$

$$p(\mathbf{w}|\mathbf{x}, \mathbf{t}, \alpha, \beta) \propto p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta)p(\mathbf{w}|\alpha)$$

- Maximum a *posteriori* (MAP)

$$\mathbf{w}_{\mathrm{MAP}} = \underset{\mathbf{w}}{\operatorname{argmax}}\, p(\mathbf{w}|\mathbf{x}, \mathbf{t}, \alpha, \beta)$$

$$= \underset{\mathbf{w}}{\operatorname{argmin}} \left\{\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}\right\}$$

regularized sum-of-squares error function if regularization parameter $\lambda = \alpha/\beta$

Predictive Distribution

Full Bayesian approach treatment → Integrate over all values of **w** (marginalization)

- Predictive distribution

$$p(t|x,\mathbf{x},\mathbf{t}) = \int p(t|x,\mathbf{w})p(\mathbf{w}|\mathbf{x},\mathbf{t})\,\mathrm{d}\mathbf{w}$$

$$p(t|x,\mathbf{x},\mathbf{t}) = \mathcal{N}\left(t|m(x),s^2(x)\right)$$

$$\begin{aligned} m(x) &= \beta\boldsymbol{\phi}(x)^{\mathrm{T}}\mathbf{S}\sum_{n=1}^{N}\boldsymbol{\phi}(x_n)t_n \\ s^2(x) &= \beta^{-1} + \boldsymbol{\phi}(x)^{\mathrm{T}}\mathbf{S}\boldsymbol{\phi}(x). \end{aligned}$$

$$\phi(x) = \{\phi_i(x) = x^i\}_{i=0}^{M}$$

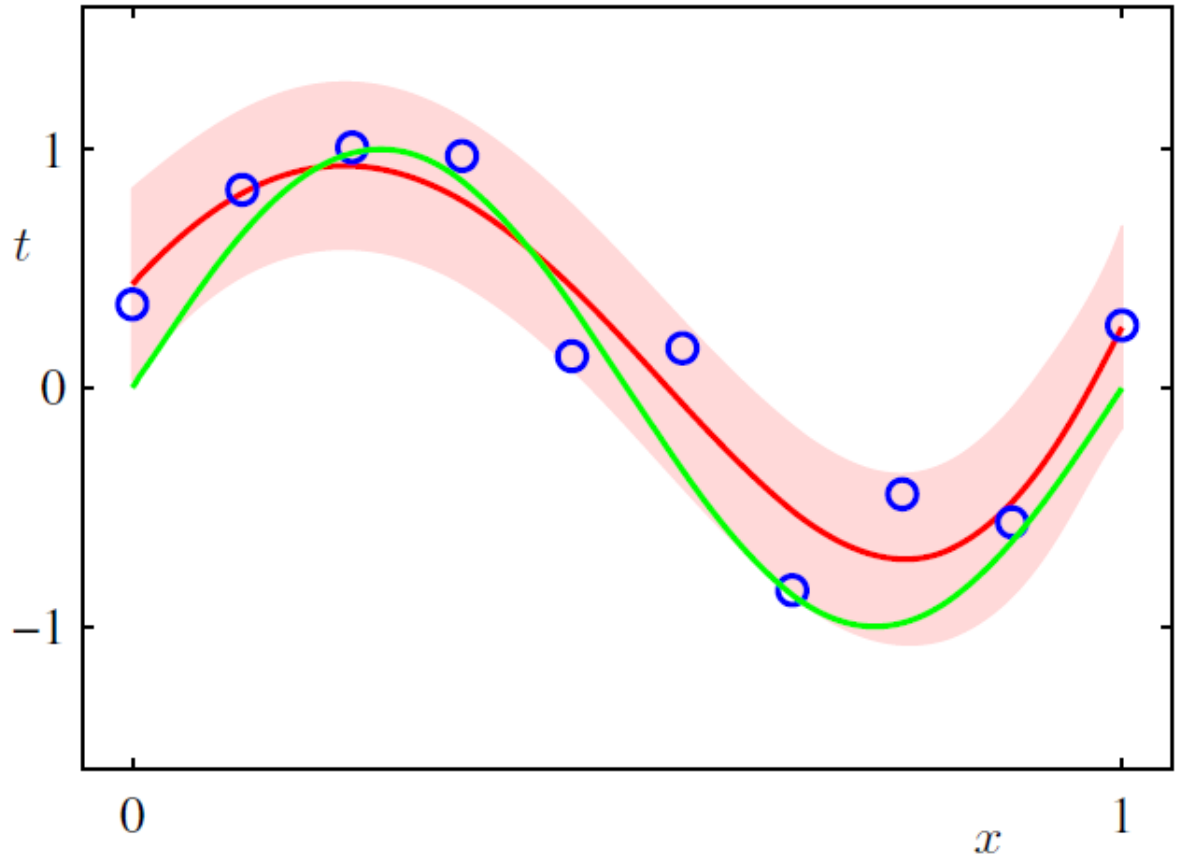
$$\mathbf{S}^{-1} = \alpha\mathbf{I} + \beta\sum_{n=1}^{N}\boldsymbol{\phi}(x_n)\boldsymbol{\phi}(x)^{\mathrm{T}}$$

Bayesian Curve Fitting

$M = 9$

$\alpha = 5 \times 10^{-3}$

$\beta = 11.1$



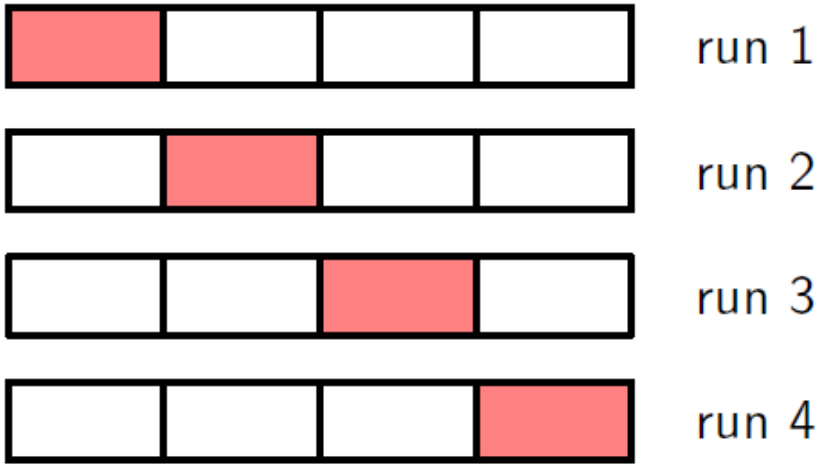
Cross Validation

Optimal order → the best generalization

→ Model complexity → predictive performance on new data

validation set → test set

- Illustration of *S*-fold cross-validation



Decision Theory

- Make optimal decisions in situation involving uncertainty in pattern recognition

  Input **x** → target t X-ray image

  C_1 (presence of cancer) C_2 (absence of cancer)

  $t = 0$ $t = 1$

→ give the treatment to the patient or not

$$\hat{C}_{MAP} = \text{argmax}_{C_k} P(C_k|\mathbf{x}) = \text{argmax}_{C_k} P(\mathbf{x}, C_k)$$

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}$$

Minimizing the Expected Loss

- Loss matrix

  $\{L_{kj}\}$

$$\begin{array}{c} \\ \text{cancer} \\ \text{normal} \end{array} \begin{array}{c} \begin{array}{cc} \text{cancer} & \text{normal} \end{array} \\ \begin{pmatrix} 0 & 1000 \\ 1 & 0 \end{pmatrix} \end{array}$$

- Minimize total loss incurred

  L_{kj} : we assign x to class C_j if the true class is C_k

- We are minimizing the Bayes risk or the expected loss

$$\mathbb{E}[L] = \sum_k \sum_j \int_{\mathcal{R}_j} L_{kj} p(\mathbf{x}, \mathcal{C}_k) \, \mathrm{d}\mathbf{x}$$

- Equivalently, we assign new x to class C_j if

  $\hat{C}_j = \text{argmin}_{C_j} \sum_k L_{kj} P(C_k|\mathbf{x})$ is minimum

CONTENTS

Parametric distribution → Nonparametric distribution

↳depends on size of data set

Conjugate prior

D= $\mathbf{x}_1, \ldots, \mathbf{x}_N \Longrightarrow$ p(x)

Binary Variables

Bernoulli (Swiss) ,1654-1705 $\quad x \in \{0, 1\} \quad \mathrm{Bern}(x|\mu) = \mu^x (1-\mu)^{1-x}$

$$p(\mathcal{D}|\mu) = \prod_{n=1}^{N} p(x_n|\mu) = \prod_{n=1}^{N} \mu^{x_n}(1-\mu)^{1-x_n}$$

$$\mu_{\mathrm{ML}} = \mathrm{argmax}_{\mu} \ln p(\mathcal{D}|\mu) = \frac{1}{N}\sum_{n=1}^{N} x_n$$

$$\mathrm{Bin}(m|N,\mu) = \binom{N}{m}\mu^m(1-\mu)^{N-m}$$

Number *m* of observations of x=1

$$\mathbb{E}[m] \equiv \sum_{m=0}^{N} m\mathrm{Bin}(m|N,\mu) = N\mu$$

$$\mathrm{var}[m] \equiv \sum_{m=0}^{N} (m-\mathbb{E}[m])^2 \mathrm{Bin}(m|N,\mu) = N\mu(1-\mu)$$

Beta Distribution

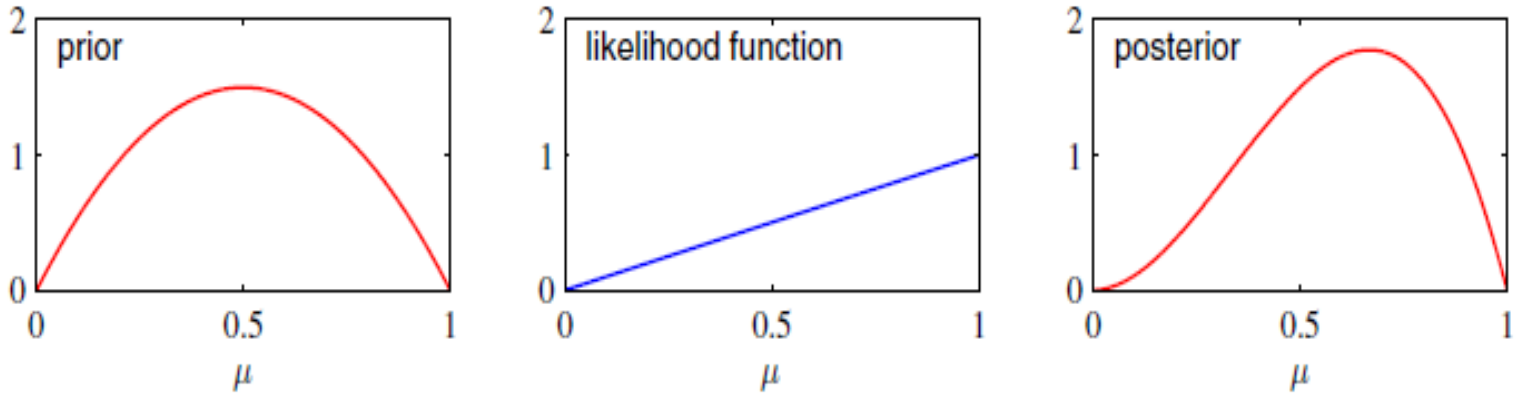
$$\mathrm{Beta}(\mu|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\mu^{a-1}(1-\mu)^{b-1}$$

$$\mathbb{E}[\mu] = \frac{a}{a+b}$$

$$\mathrm{var}[\mu] = \frac{ab}{(a+b)^2(a+b+1)}$$

$$p(\mu|m,l,a,b) \propto \mu^{m+a-1}(1-\mu)^{l+b-1}$$

$$= \frac{\Gamma(m+a+l+b)}{\Gamma(m+a)\Gamma(l+b)}\mu^{m+a-1}(1-\mu)^{l+b-1}$$



Multinomial Variables

$$\mathbf{x} = (0, 0, 1, 0, 0, 0)^{\mathrm{T}}$$

Constraint $\sum_{k=1}^{K} x_k = 1 \quad \sum_k \mu_k = 1$

$$p(\mathbf{x}|\boldsymbol{\mu}) = \prod_{k=1}^{K} \mu_k^{x_k} \qquad \text{generalization of the Bernoulli distribution.}$$

$$= p(\mathbf{x}_1, ..., \mathbf{x}_k | \mu_1, ..., \mu_k)^{k=1}$$

$$\sum_{\mathbf{x}} p(\mathbf{x}|\boldsymbol{\mu}) = \sum_{k=1}^{K} \mu_k = 1 \qquad \mathbb{E}[\mathbf{x}|\boldsymbol{\mu}] = \sum_{\mathbf{x}} p(\mathbf{x}|\boldsymbol{\mu})\mathbf{x} = (\mu_1, \ldots, \mu_M)^{\mathrm{T}} = \boldsymbol{\mu}$$

$$p(\mathcal{D}|\boldsymbol{\mu}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \mu_k^{x_{nk}} = \prod_{k=1}^{K} \mu_k^{\left(\sum_n x_{nk}\right)} = \prod_{k=1}^{K} \mu_k^{m_k}$$

$\mu_{ML} = \operatorname{argmax}_{\mu} \ln p(D|\mu)$ Lagrange optimization is applied

$$\Longrightarrow \sum_{k=1}^{K} m_k \ln \mu_k + \lambda \left(\sum_{k=1}^{K} \mu_k - 1 \right)$$

$$\mu_k^{\mathrm{ML}} = \frac{m_k}{N}$$

$$\mathrm{Mult}(m_1, m_2, \ldots, m_K | \boldsymbol{\mu}, N) = \binom{N}{m_1 m_2 \ldots m_K} \prod_{k=1}^{K} \mu_k^{m_k}$$

$$\sum_{k=1}^{K} m_k = N$$

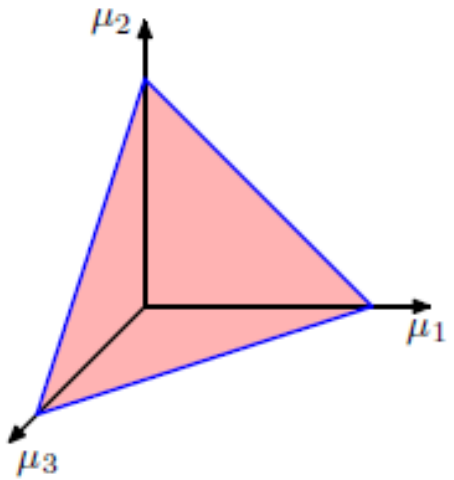
Dirichlet Distribution

Prior distribution for the parameters $\{\mu_k\}$ of a multinomial distribution is a *Dirichlet* distribution.

$$p(\boldsymbol{\mu}|\boldsymbol{\alpha}) \propto \prod_{k=1}^{K} \mu_k^{\alpha_k - 1} \qquad 0 \leqslant \mu_k \leqslant 1 \qquad \sum_k \mu_k = 1$$

Dirichlet distribution is confined to a simplex (a bounded linear manifold)

$$\mathrm{Dir}(\boldsymbol{\mu}|\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_K)} \prod_{k=1}^{K} \mu_k^{\alpha_k - 1}$$



Gaussian Distribution

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

Central limit theorem

Eigenvector equation

$$\boldsymbol{\Sigma}\mathbf{u}_i = \lambda_i\mathbf{u}_i$$

$$\boldsymbol{\Sigma} = \sum_{i=1}^{D}\lambda_i\mathbf{u}_i\mathbf{u}_i^{\mathrm{T}}$$

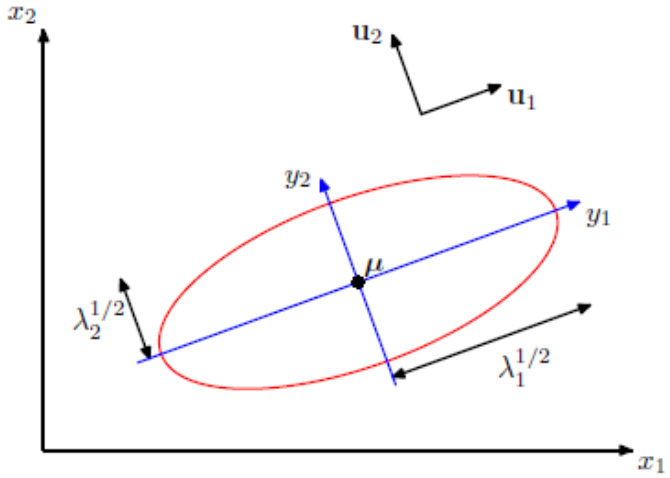
$$\boldsymbol{\Sigma}^{-1} = \sum_{i=1}^{D}\frac{1}{\lambda_i}\mathbf{u}_i\mathbf{u}_i^{\mathrm{T}}$$

$$\mathbf{u}_i^{\mathrm{T}}\mathbf{u}_j = I_{ij}$$

$$\Delta^2 = \sum_{i=1}^{D}\frac{y_i^2}{\lambda_i}$$

$$y_i = \mathbf{u}_i^{\mathrm{T}}(\mathbf{x}-\boldsymbol{\mu})$$

$$\mathbf{y} = \mathbf{U}(\mathbf{x}-\boldsymbol{\mu})$$



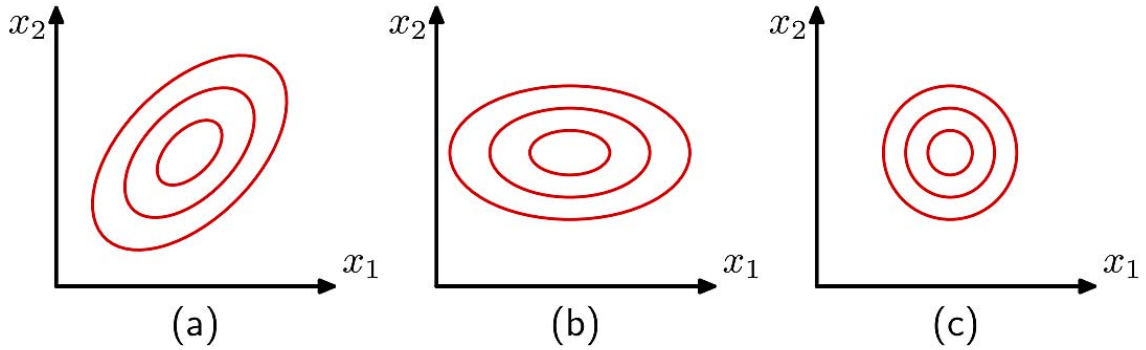
positive definite → positive semidefinite

$$\lambda_i > 0 \qquad \lambda_i \geq 0$$

D grows quadratically → High computational load

Two simplifications $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_i^2)$ diagonal no. of par =2*D*

$\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$ isotropic no. of par =*D*+1



Gaussian → Unimodal

→ multimodal → introduce hidden variable mixture of Gaussians

Markov random field, linear dynamical system

↓ ↓

images tracking Chap 8,12

Maximum Likelihood for the Gaussian

$$\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^{\mathrm{T}}$$

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{ND}{2}\ln(2\pi) - \frac{N}{2}\ln|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \boldsymbol{\mu})$$

$$\frac{\partial}{\partial \boldsymbol{\mu}}\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \boldsymbol{\mu})$$

$$\boldsymbol{\mu}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

$$\boldsymbol{\Sigma}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu}_{\mathrm{ML}})(\mathbf{x}_n - \boldsymbol{\mu}_{\mathrm{ML}})^{\mathrm{T}}$$

joint maximization over $\mu \& \Sigma$

$$\mathbb{E}[\boldsymbol{\mu}_{\mathrm{ML}}] = \boldsymbol{\mu}$$

$$\mathbb{E}[\boldsymbol{\Sigma}_{\mathrm{ML}}] = \frac{N-1}{N}\boldsymbol{\Sigma}$$

$$\widetilde{\boldsymbol{\Sigma}} = \frac{1}{N-1}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu}_{\mathrm{ML}})(\mathbf{x}_n - \boldsymbol{\mu}_{\mathrm{ML}})^{\mathrm{T}}$$

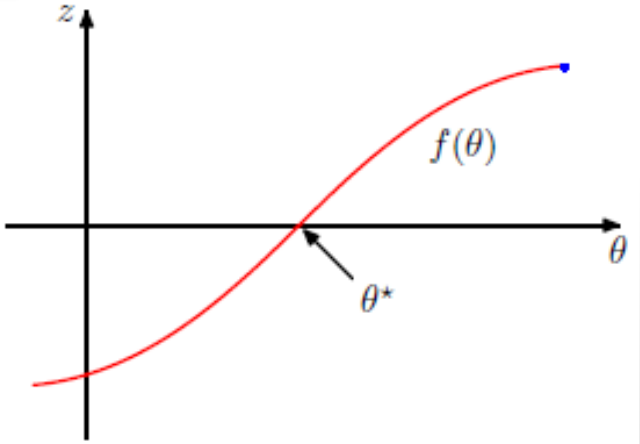
Sequential Estimation

Sequential methods allow data points to be processed one at a time and then discarded → on-line application

$$\begin{aligned}
\boldsymbol{\mu}_{\mathrm{ML}}^{(N)} &= \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \\
&= \frac{1}{N}\mathbf{x}_N + \frac{1}{N}\sum_{n=1}^{N-1}\mathbf{x}_n \\
&= \frac{1}{N}\mathbf{x}_N + \frac{N-1}{N}\boldsymbol{\mu}_{\mathrm{ML}}^{(N-1)} \\
&= \boldsymbol{\mu}_{\mathrm{ML}}^{(N-1)} + \frac{1}{N}(\mathbf{x}_N - \boldsymbol{\mu}_{\mathrm{ML}}^{(N-1)})
\end{aligned}$$

Robbins-Monro algorithm : consider a pair of variables θ and z governed by $p(z,\theta)$

$$f(\theta) \equiv \mathbb{E}[z|\theta] = \int zp(z|\theta)\,\mathrm{d}z$$



Bayesian Inference for Gaussian Mean

Inferring μ given $\mathbf{X} = \{x_1, \ldots, x_N\}$ & σ^2 is known

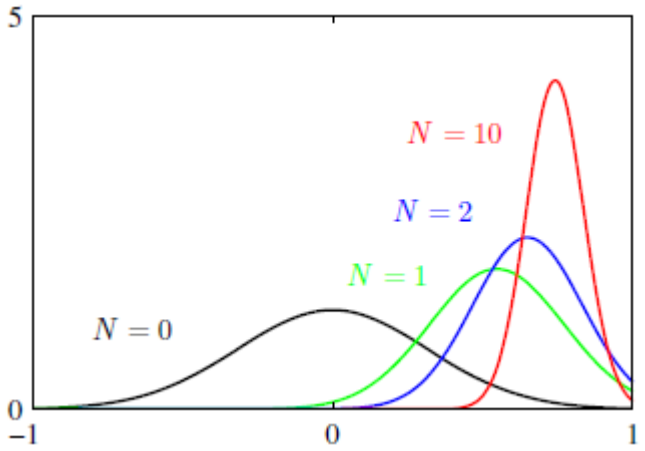
$$p(\mathbf{X}|\mu) = \prod_{n=1}^{N} p(x_n|\mu) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n - \mu)^2\right\}$$

Conjugate prior: $p(\mu) = \mathcal{N}\left(\mu|\mu_0, \sigma_0^2\right)$

Posterior distribution : $p(\mu|\mathbf{X}) \propto p(\mathbf{X}|\mu)p(\mu)$

$$p(\mu|\mathbf{X}) = \mathcal{N}\left(\mu|\mu_N, \sigma_N^2\right)$$

$$\begin{aligned} \mu_N &= \frac{\sigma^2}{N\sigma_0^2 + \sigma^2}\mu_0 + \frac{N\sigma_0^2}{N\sigma_0^2 + \sigma^2}\mu_{\mathrm{ML}} \\ \frac{1}{\sigma_N^2} &= \frac{1}{\sigma_0^2} + \frac{N}{\sigma^2} \end{aligned}$$



Bayesian paradigm leads naturally to

a sequential view of the inference problem

$$p(\boldsymbol{\mu}|D) \propto \left[p(\boldsymbol{\mu})\prod_{n=1}^{N-1} p(\mathbf{x}_n|\boldsymbol{\mu})\right] p(\mathbf{x}_N|\boldsymbol{\mu})$$

Bayesian Inference for Gaussian Precision

Suppose mean is known, we wish to infer the variance or precision $\lambda \equiv 1/\sigma^2$

Likelihood :

$$p(\mathbf{X}|\lambda) = \prod_{n=1}^{N} \mathcal{N}(x_n|\mu, \lambda^{-1}) \propto \lambda^{N/2} \exp\left\{-\frac{\lambda}{2}\sum_{n=1}^{N}(x_n-\mu)^2\right\}$$

Conjugate prior :

$$\mathrm{Gam}(\lambda|a,b) = \frac{1}{\Gamma(a)} b^a \lambda^{a-1} \exp(-b\lambda)$$

$$\mathbb{E}[\lambda] = \frac{a}{b} \qquad \mathrm{var}[\lambda] = \frac{a}{b^2}$$

In case of using $\mathrm{Gam}(\lambda|a_0, b_0)$, we have

$$p(\lambda|\mathbf{X}) \propto \lambda^{a_0-1}\lambda^{N/2} \exp\left\{-b_0\lambda - \frac{\lambda}{2}\sum_{n=1}^{N}(x_n-\mu)^2\right\}$$

$$= \mathrm{Gam}(\lambda|a_N, b_N)$$

$$a_N = a_0 + \frac{N}{2} \qquad b_N = b_0 + \frac{1}{2}\sum_{n=1}^{N}(x_n-\mu)^2 = b_0 + \frac{N}{2}\sigma^2_{\mathrm{ML}}$$

Inference for Gaussian Mean and Precision

Suppose both the mean and the precision are unknown.

Likelihood :
$$p(\mathbf{X}|\mu,\lambda)=\prod_{n=1}^{N}\left(\frac{\lambda}{2\pi}\right)^{1/2}\exp\left\{-\frac{\lambda}{2}(x_n-\mu)^2\right\}$$
$$\propto\left[\lambda^{1/2}\exp\left(-\frac{\lambda\mu^2}{2}\right)\right]^N\exp\left\{\lambda\mu\sum_{n=1}^{N}x_n-\frac{\lambda}{2}\sum_{n=1}^{N}x_n^2\right\}$$

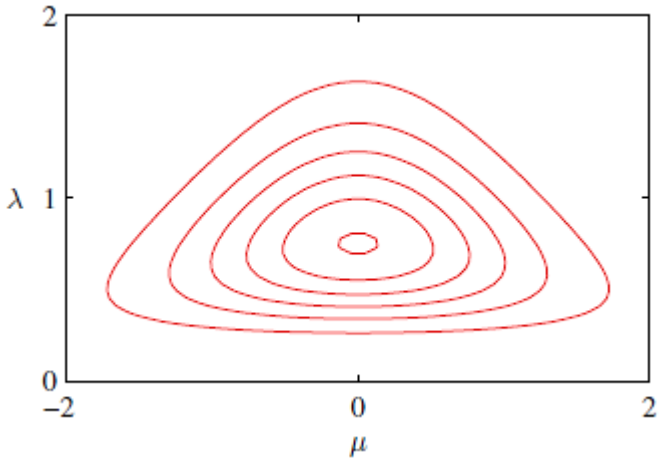
Conjugate prior :
$$p(\mu,\lambda)\propto\left[\lambda^{1/2}\exp\left(-\frac{\lambda\mu^2}{2}\right)\right]^{\beta}\exp\{c\lambda\mu-d\lambda\}$$
$$=\exp\left\{-\frac{\beta\lambda}{2}(\mu-c/\beta)^2\right\}\lambda^{\beta/2}\exp\left\{-\left(d-\frac{c^2}{2\beta}\right)\lambda\right\}$$

Normal-gamma distribution $\boxed{p(\mu,\lambda)=\mathcal{N}(\mu|\mu_0,(\beta\lambda)^{-1})\mathrm{Gam}(\lambda|a,b)}$

$$\mu_0=c/\beta,\ a=1+\beta/2,\ b=d-c^2/2\beta$$

In case of multivariate Gaussian distribution

$$\mathcal{N}\left(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Lambda}^{-1}\right)\quad\mathbf{x}\in R^D$$



For known mean & unknown precision matrix $\mathbf{\Lambda}$,

the conjugate prior is Wishart distribution

$$\mathcal{W}(\mathbf{\Lambda}|\mathbf{W},\nu) = B|\mathbf{\Lambda}|^{(\nu-D-1)/2}\exp\left(-\frac{1}{2}\mathrm{Tr}(\mathbf{W}^{-1}\mathbf{\Lambda})\right)$$

If both mean and precision are unknown, the conjugate prior is given by

$$p(\boldsymbol{\mu},\mathbf{\Lambda}|\boldsymbol{\mu}_0,\beta,\mathbf{W},\nu) = \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{\mu}_0,(\beta\mathbf{\Lambda})^{-1})\,\mathcal{W}(\mathbf{\Lambda}|\mathbf{W},\nu)$$

Student's *t*-Distribution

$$\begin{aligned}
p(x|\mu,a,b) &= \int_0^\infty \mathcal{N}(x|\mu,\tau^{-1})\mathrm{Gam}(\tau|a,b)\,\mathrm{d}\tau \\
&= \int_0^\infty \frac{b^a e^{(-b\tau)}\tau^{a-1}}{\Gamma(a)}\left(\frac{\tau}{2\pi}\right)^{1/2}\exp\left\{-\frac{\tau}{2}(x-\mu)^2\right\}\,\mathrm{d}\tau \\
&= \frac{b^a}{\Gamma(a)}\left(\frac{1}{2\pi}\right)^{1/2}\left[b+\frac{(x-\mu)^2}{2}\right]^{-a-1/2}\Gamma(a+1/2)
\end{aligned}$$

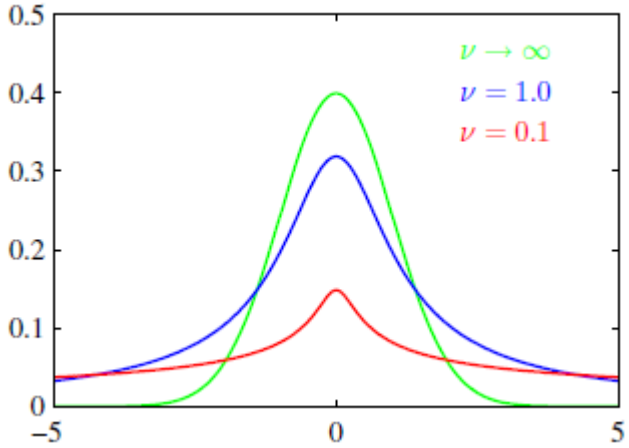
$$\mathrm{St}(x|\mu,\lambda,\nu) = \frac{\Gamma(\nu/2+1/2)}{\Gamma(\nu/2)}\left(\frac{\lambda}{\pi\nu}\right)^{1/2}\left[1+\frac{\lambda(x-\mu)^2}{\nu}\right]^{-\nu/2-1/2}$$

$$\nu = 2a \text{ and } \lambda = a/b$$

λ :precision ν :degree of freedom

Student's *t*-Distribution

$$\nu \rightarrow \infty \qquad \mathrm{St}(x|\mu,\lambda,\nu) \longrightarrow \mathcal{N}(x|\mu,\lambda^{-1})$$



t-distribution is much less sensitive than a Gaussian to the presence of outliers

→ robustness

Compared to a Gaussian,

t distribution contains the Gaussian as a special case.

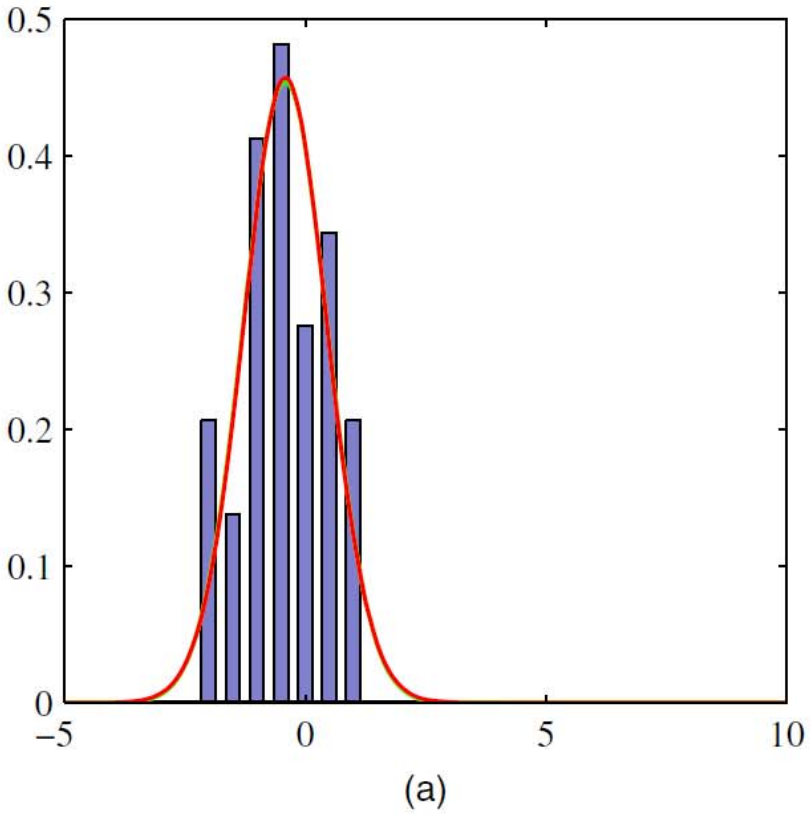
$$\mathrm{St}(x|\mu,\lambda,\nu) = \int_0^\infty \mathcal{N}\left(x|\mu,(\eta\lambda)^{-1}\right)\mathrm{Gam}(\eta|\nu/2,\nu/2)\,\mathrm{d}\eta$$

Multivariate case:

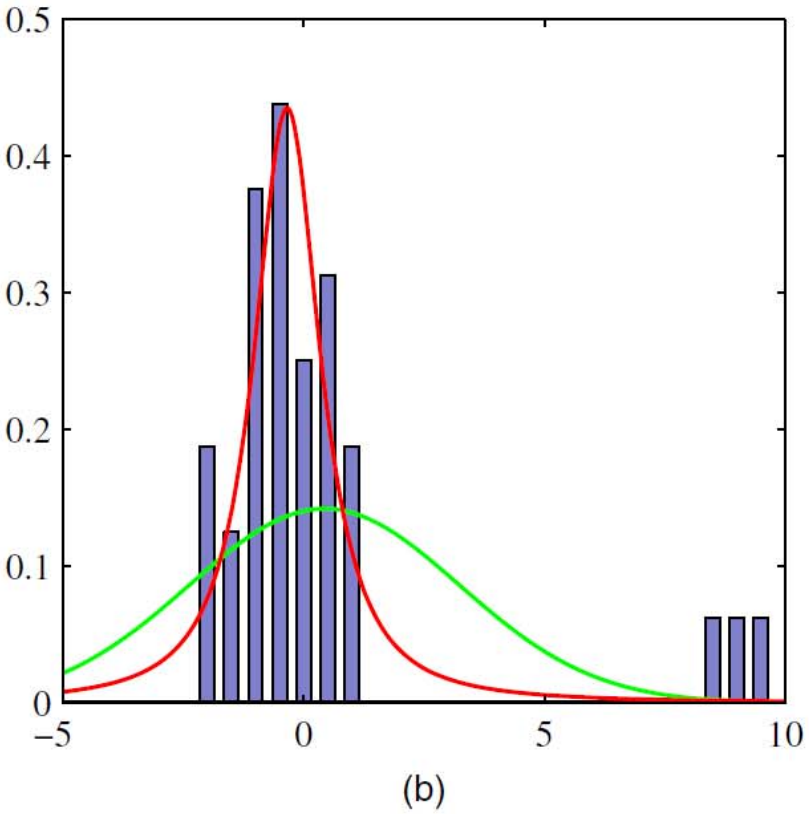
$$\mathrm{St}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Lambda},\nu) = \int_0^\infty \mathcal{N}(\mathbf{x}|\boldsymbol{\mu},(\eta\boldsymbol{\Lambda})^{-1})\mathrm{Gam}(\eta|\nu/2,\nu/2)\,\mathrm{d}\eta$$

$$\mathrm{St}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Lambda},\nu) = \frac{\Gamma(D/2+\nu/2)}{\Gamma(\nu/2)}\frac{|\boldsymbol{\Lambda}|^{1/2}}{(\pi\nu)^{D/2}}\left[1+\frac{\Delta^2}{\nu}\right]^{-D/2-\nu/2}$$

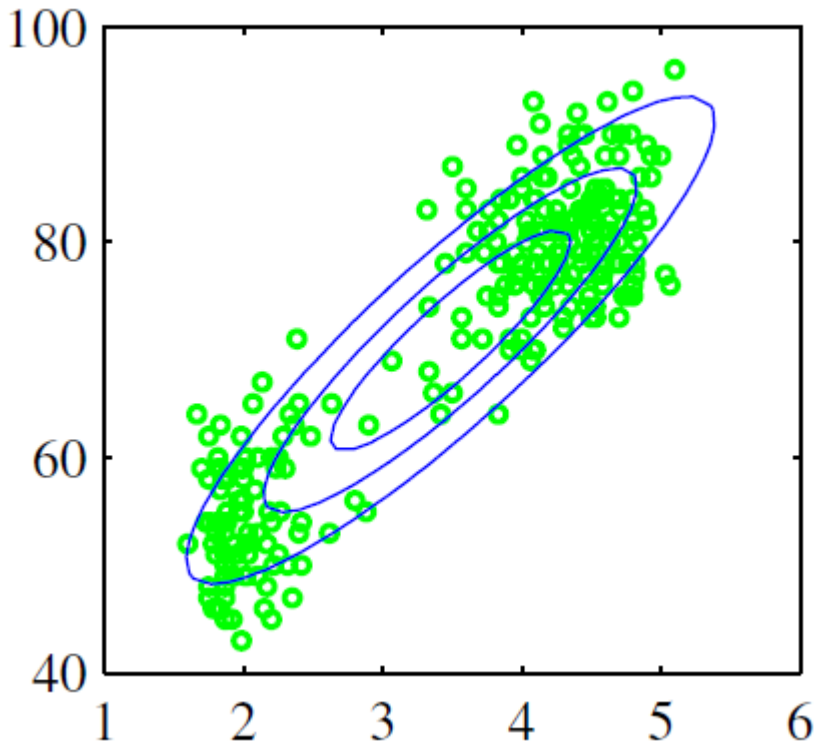
$$\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu} \qquad \mathrm{cov}[\mathbf{x}] = \frac{\nu}{(\nu-2)}\boldsymbol{\Lambda}^{-1} \qquad \mathrm{mode}[\mathbf{x}] = \boldsymbol{\mu}$$

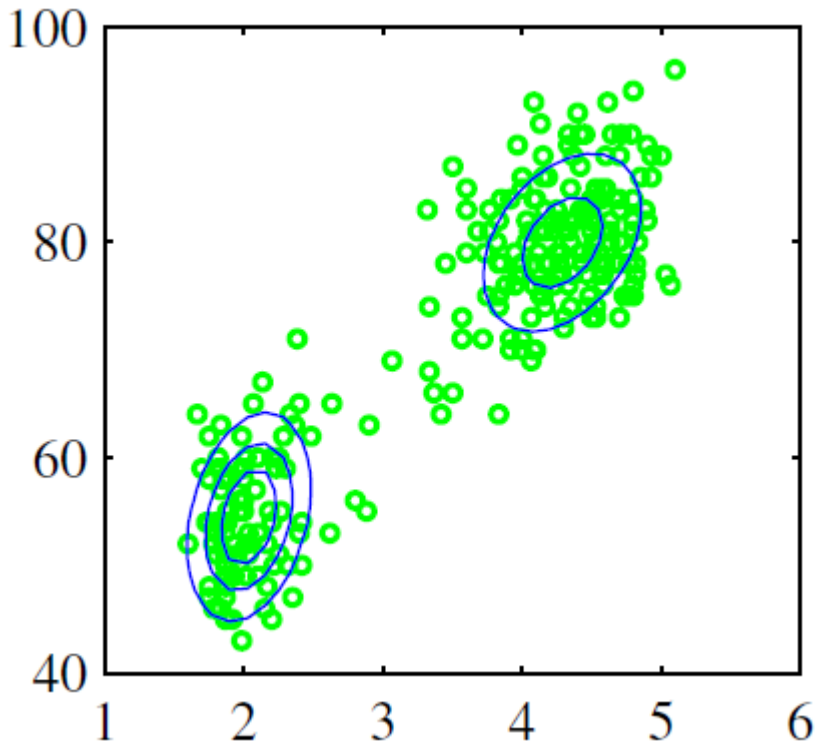


(a)

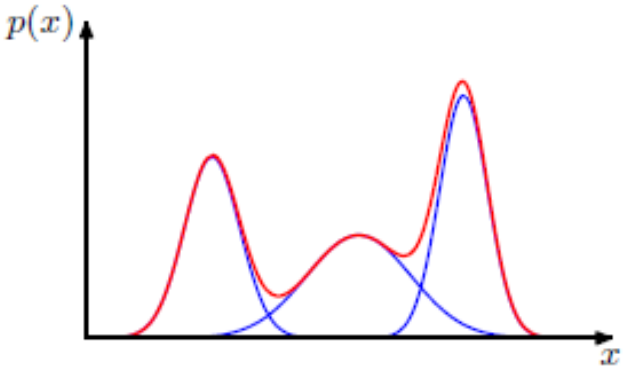


(b)


100
80
60
40
1
2
3
4
5
6


100
80
60
40
1
2
3
4
5
6

Mixtures of Gaussians



A linear combination of Gaussians can give rise very complex densities.

mixtures of Bernoulli distribution → Discrete vari

mixtures of Gaussians $p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

$$\sum_{k=1}^{K} \pi_k = 1 \qquad 0 \leqslant \pi_k \leqslant 1 \qquad p(\mathbf{x}) = \sum_{k=1}^{K} p(k)p(\mathbf{x}|k)$$

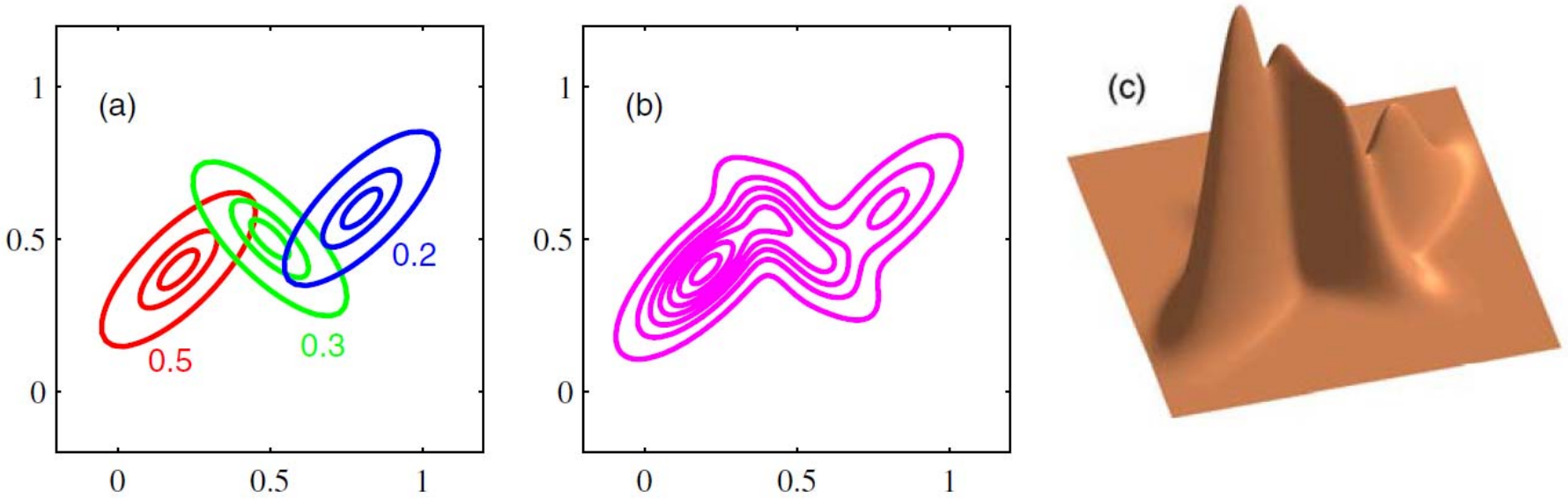
$$\Lambda = \{ \boldsymbol{\pi} \equiv \{\pi_1, \ldots, \pi_K\}, \boldsymbol{\mu} \equiv \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K\}, \boldsymbol{\Sigma} \equiv \{\boldsymbol{\Sigma}_1, \ldots \boldsymbol{\Sigma}_K\} \}$$

ML: $$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

$$\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$$

No closed-form solution 1. iterative numerical optimization

2. expectation-maximization algorithm


(a)
(b)
(c)
0.2
0.3
0.5
0
0.5
1

CONTENTS

Linear Models for Regression

- Unsupervised learning $\Longrightarrow$ Supervised learning
- Input variables $\longrightarrow$ Target variables

$$\{x_n\} \qquad \{t_n\} \qquad n = 1, \ldots, N$$

- We are finding $y(\mathbf{x}) \longrightarrow t$ & even the predictive distribution

Linear Basis Function Models

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D \qquad \mathbf{x} = (x_1, \ldots, x_D)^{\mathrm{T}}$$

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

- Linear combinations of fixed nonlinear functions where $\phi_0(\mathbf{x}) = 1$ $\{\phi_j(\mathbf{x})\}$ can be interpreted as the process of feature extraction.
- Linear models using nonlinear basis model.

- Polynomial regression is built using basis function Ⅰ. $\phi_j(x) = x^j$
- Also, we can use Ⅱ. $\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$ changes in one region will affect all other regions.

μ_j :location of basis function in input space

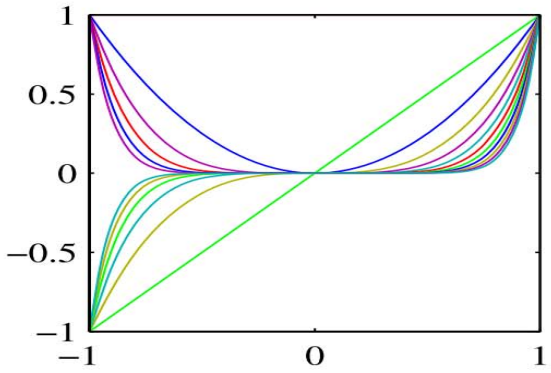
Ⅱ. Gaussian basis function with spatial scale S

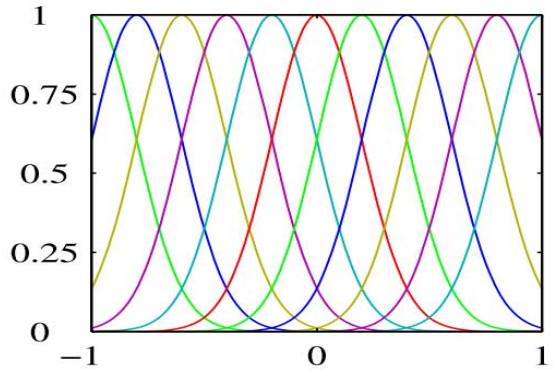
Ⅲ. Sigmoid basis function

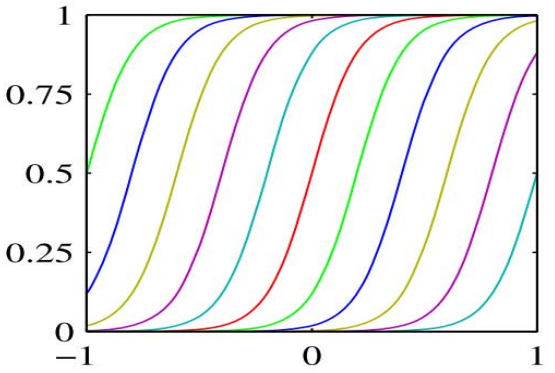
$$\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$$

$$\sigma(a) = \frac{1}{1+\exp(-a)}$$

$$\tanh(a) = 2\sigma(a) - 1 = \frac{1-e^a}{1+e^a}$$







Maximum Likelihood and Least Squares

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \qquad \epsilon \infty N(O, \beta^{-1})$$

$$\longrightarrow \quad p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

$$\longrightarrow \quad \mathbb{E}[t|\mathbf{x}] = \int t p(t|\mathbf{x})\, \mathrm{d}t = y(\mathbf{x}, \mathbf{w})$$

- Input data $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ Target variables $t = \{t_1, ..., t_N\}$

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1})$$

$$\begin{aligned} \ln p(\mathbf{t}|\mathbf{w}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \\ &= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathbf{w}) \end{aligned} \qquad E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

$$\nabla \ln p(\mathbf{t}|\mathbf{w}, \beta) = \sum_{n=1}^{N} \{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\} \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} \qquad 0 = \sum_{n=1}^{N} t_n \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} - \mathbf{w}^{\mathrm{T}} \left(\sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}_n) \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} \right)$$

$$\mathbf{w}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}$$ *normal equations* for the least squares problem

where $$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}$$

- If we make the bias parameter explicit, the error function becomes

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - w_0 - \sum_{j=1}^{M-1} w_j\phi_j(\mathbf{x}_n)\}^2$$

$$\frac{\partial E_D(w_0)}{\partial w_0} = 0 \quad \Longrightarrow \quad \boxed{w_0 = \bar{t} - \sum_{j=1}^{M-1} w_j\overline{\phi_j}}$$

$$\bar{t} = \frac{1}{N}\sum_{n=1}^{N} t_n, \qquad \overline{\phi_j} = \frac{1}{N}\sum_{n=1}^{N}\phi_j(\mathbf{x}_n)$$

w_0 compensates for the difference between $\bar{t}$ and the weighted sum of $\overline{\phi_j}$

$$\frac{\partial \ln P(t|\mathbf{x},\mathbf{w},\beta)}{\partial \beta} = 0 \qquad \boxed{\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{t_n - \mathbf{w}_{\mathrm{ML}}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2}$$

Geometry of Least Squares

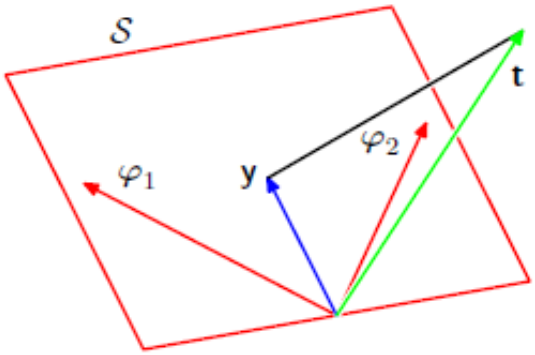
$t \longrightarrow y$

y is an "orthogonal projection" of data vector *t* onto the subspace spanned by $\phi_j(\mathbf{x})$

$\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}$ is close to singular

→ degeneracy →SVD

→ regularization term ensures nonsingular matrix.



Sequential Learning

- We can use "*stochastic gradient descent*" *or* "*sequential gradient descent*"
- Let error function $E = \sum_n E_n$
- After presentation of pattern n, $\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n$

$= \mathbf{w}^{(\tau)} + \eta(t_n - \mathbf{w}^{(\tau)\mathrm{T}}\boldsymbol{\phi}_n)\boldsymbol{\phi}_n$

$\boldsymbol{\phi}_n = \boldsymbol{\phi}(\mathbf{x}_n)$ *least-mean-squares* or the *LMS algorithm*

Regularized Least Squares

- To control over-fitting, we minimize

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$: weight decay

parameter shrinkage

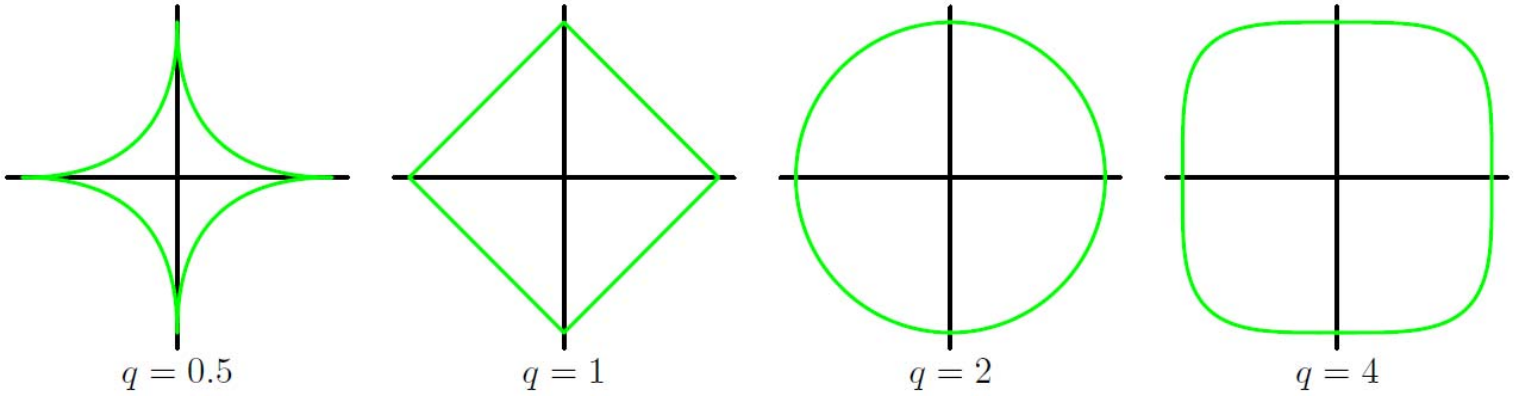
$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

$$\mathbf{w} = \left(\lambda\mathbf{I} + \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}$$

- Sometimes, we use $E_W(\mathbf{w}) = \frac{1}{2}\sum_{j=1}^{M}|w_j|^q$

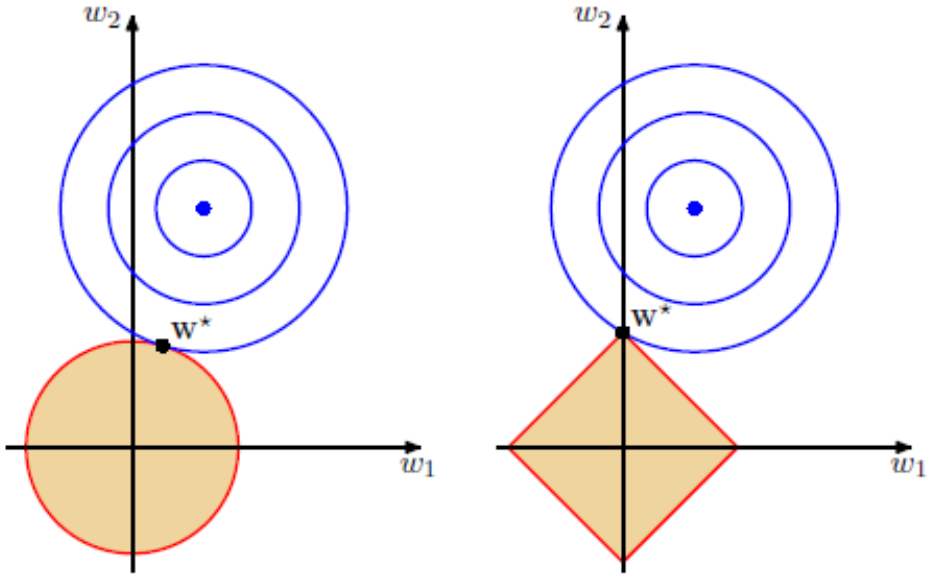
  q = 1 : *lasso* regularizer

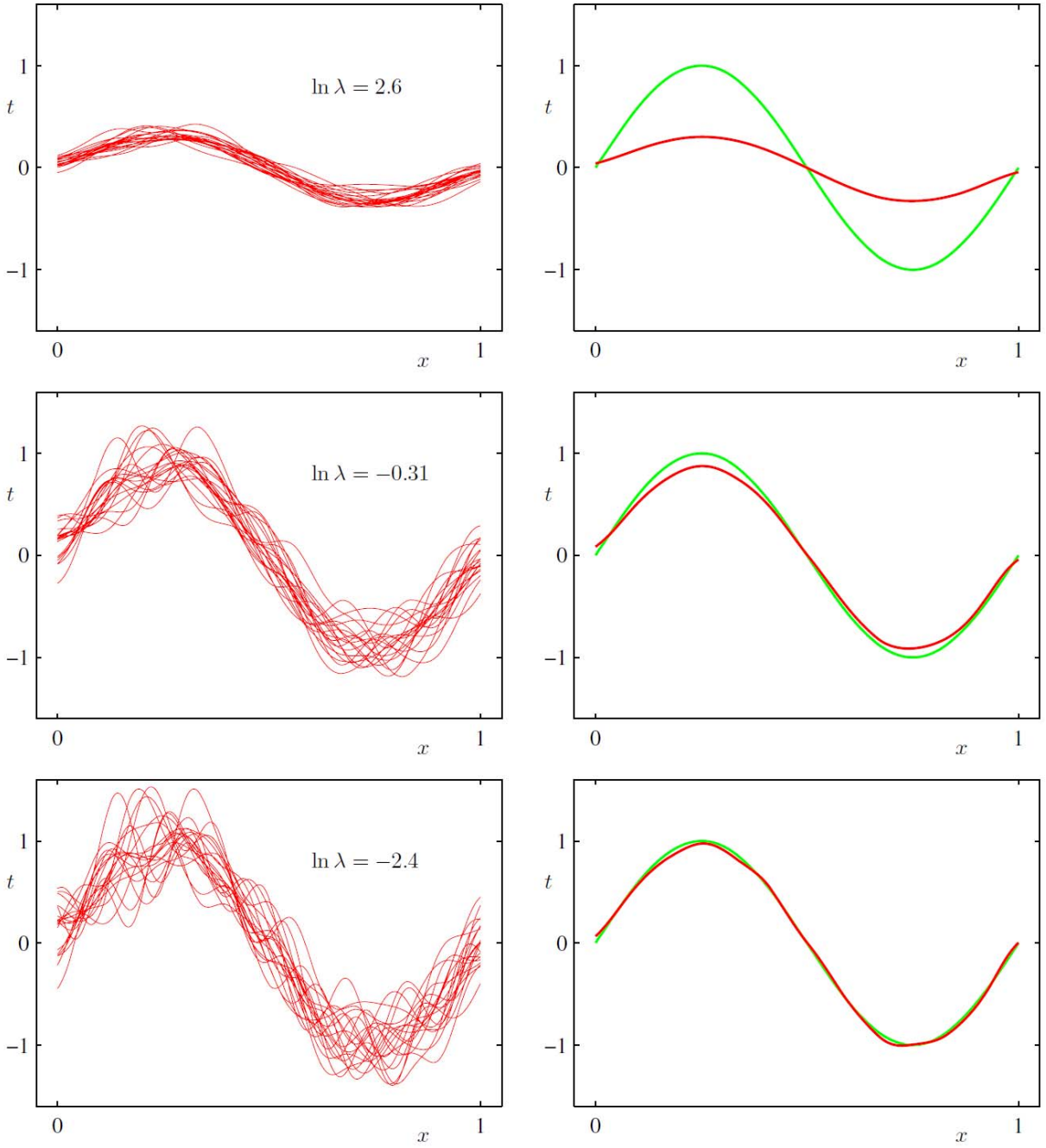
  q = 2 : quadratic regularizer



Lasso Regularizer

- In case $q = 1$ & λ is sufficiently large, some w_j vanishes & *sparse* model happens
- We should minimize the unregularized sum-of-squares error subject to $\sum_{j=1}^{M} |w_j|^q \leqslant \eta$




ln λ = 2.6
ln λ = −0.31
ln λ = −2.4
t
x
0
1
−1

Bayesian Linear Regression

- Number of basis functions should be dependent on number of data size.
- Using held-out data can be computationally expensive and wasteful of valuable data. Bayesian treatment using only training data.

Parameter Distribution

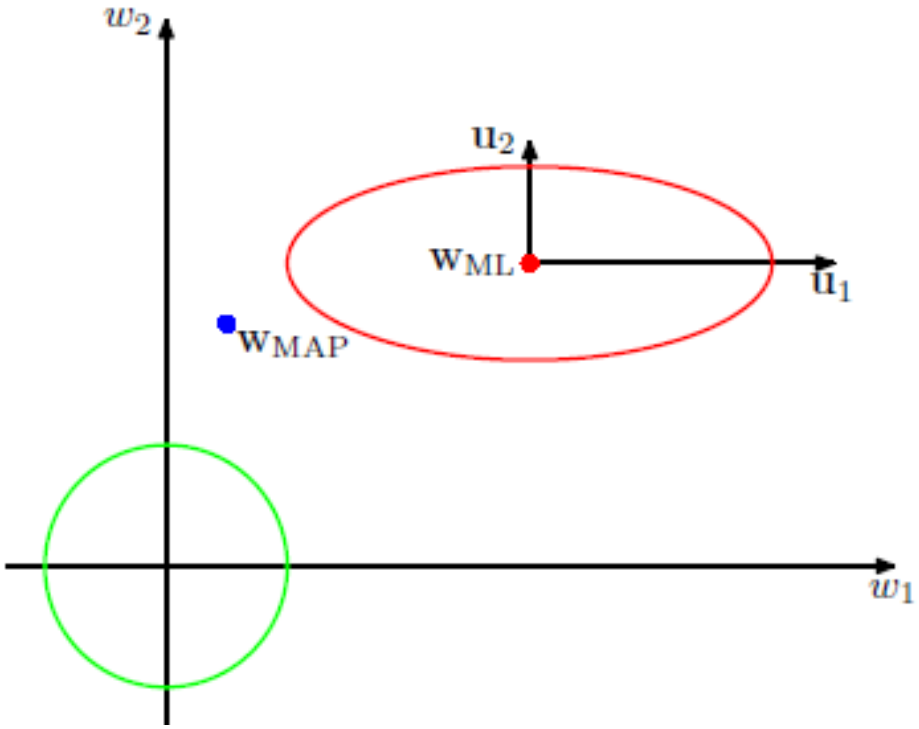
- $p(\mathbf{t}|\mathbf{w})$ has the exponential of a quadratic function of **w.**
- Conjugate prior for **w** is given by

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0)$$

$$p(\mathbf{w}|\mathbf{t}) \propto p(\mathbf{t}|\mathbf{w})p(\mathbf{w}) = N(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$$

$$\begin{aligned} \mathbf{m}_N &= \mathbf{S}_N\left(\mathbf{S}_0^{-1}\mathbf{m}_0 + \beta\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}\right) \\ \mathbf{S}_N^{-1} &= \mathbf{S}_0^{-1} + \beta\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}. \end{aligned}$$

To simplify the treatment, we consider a zero-mean isotropic Gaussian as a prior $p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$


w_2
$\mathbf{u}_2$
$\mathbf{w}_{\mathrm{ML}}$
$\mathbf{u}_1$
$\mathbf{w}_{\mathrm{MAP}}$
w_1

Predictive Distribution

- We are interested in predicting t for new x, predictive distribution is given by

$$p(t|\mathbf{t},\alpha,\beta) = \int p(t|\mathbf{w},\beta)p(\mathbf{w}|\mathbf{t},\alpha,\beta)\,\mathrm{d}\mathbf{w}$$

- We obtain

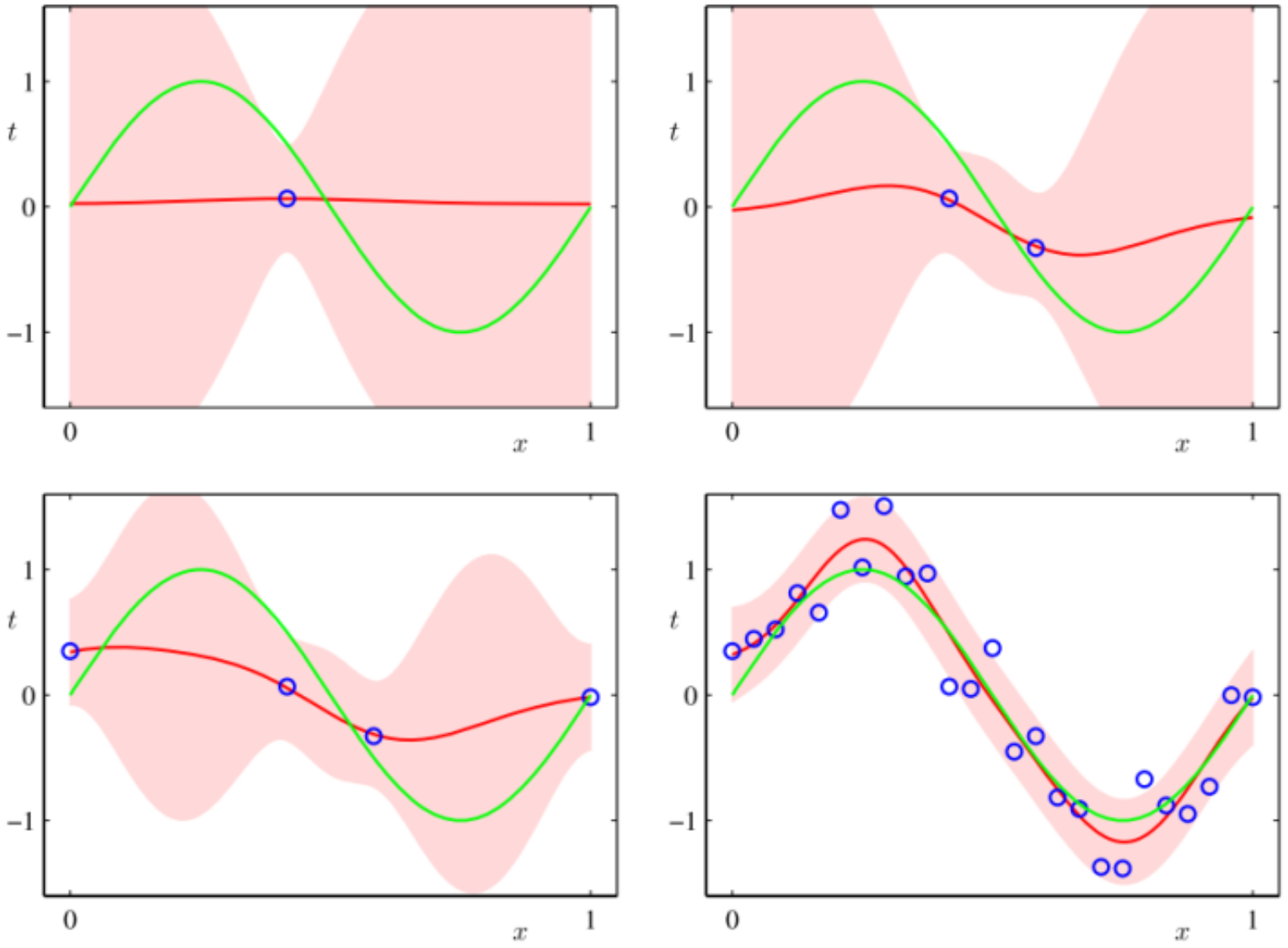
$$p(t|\mathbf{x},\mathbf{t},\alpha,\beta) = \mathcal{N}(t|\mathbf{m}_N^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}),\sigma_N^2(\mathbf{x}))$$

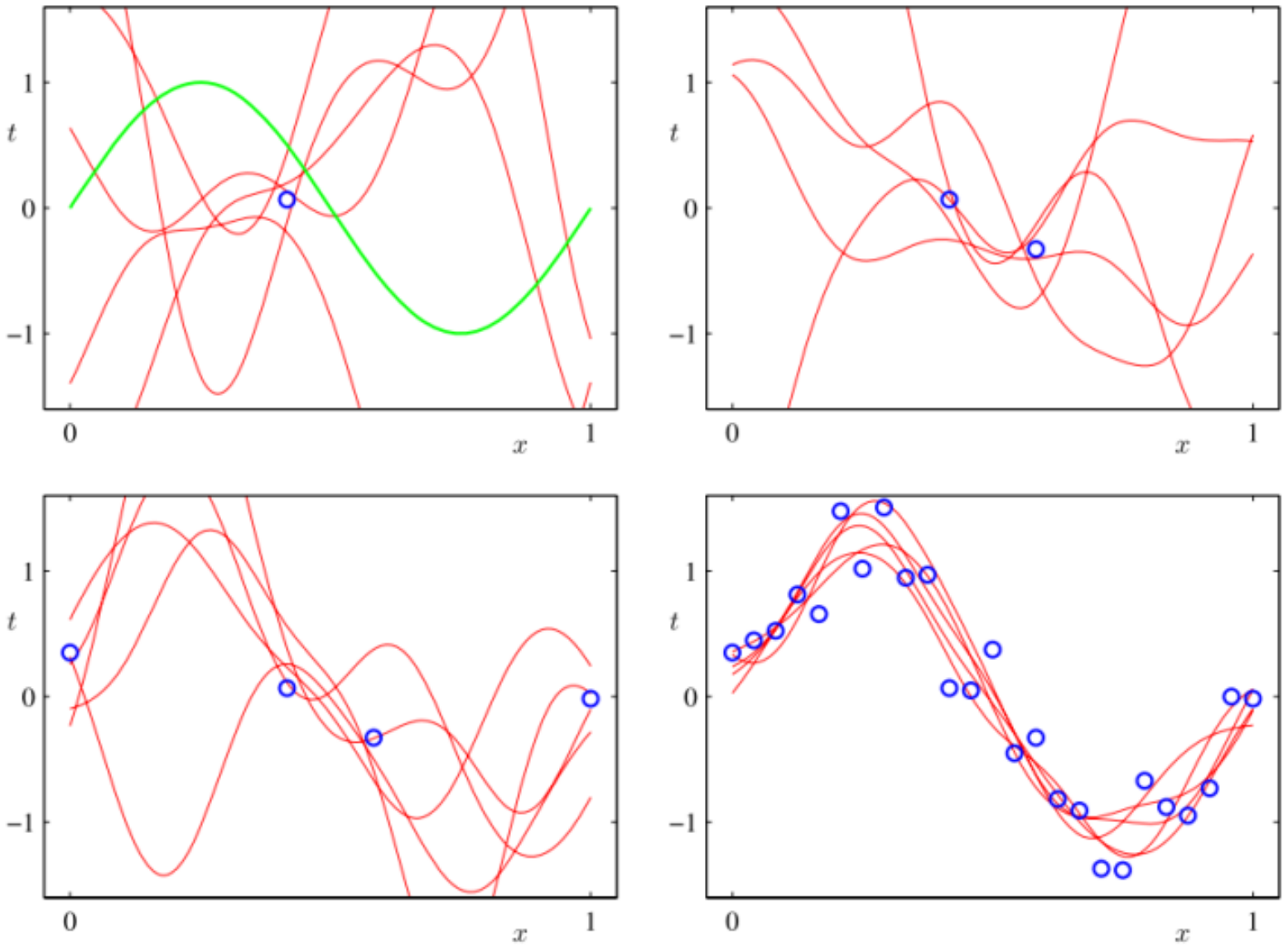
where

$$\sigma_N^2(\mathbf{x}) = \frac{1}{\beta} + \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x})$$

$$\sigma_{N+1}^2(\mathbf{x}) \leq \sigma_N^2(\mathbf{x}) \text{ when } N \to \infty \quad \phi^t(\mathbf{x})S_n\phi(\mathbf{x}) \longrightarrow 0$$

- See Figures red curve shows the mean of predictive distribution.
  red shaded region implies the variance or uncertainty.


1
t
0
−1
0
x
1


1
t
0
−1
0
x
1

Bayesian Model Comparison

- We consider the problem of model selection from a Bayesian perspective.
- We can determine regularization parameter. Models can be compared without validation data. Cross-validation can be avoided. In comparison of L models $\{\mathcal{M}_i\}$, we select optimal model according to the posterior distribution. $p(\mathcal{M}_i|\mathcal{D}) \propto p(\mathcal{M}_i)p(\mathcal{D}|\mathcal{M}_i)$
- Predictive distribution is given by

$$p(t|\mathbf{x}, \mathcal{D}) = \sum_{i=1}^{L} p(t|\mathbf{x}, \mathcal{M}_i, \mathcal{D})p(\mathcal{M}_i|\mathcal{D})$$

- Use the single most probable model alone to model prediction

  $\longrightarrow$ model selection

- For a model governed by w, the model evidence is given by

$$p(\mathcal{D}|\mathcal{M}_i) = \int p(\mathcal{D}|\mathbf{w}, \mathcal{M}_i)p(\mathbf{w}|\mathcal{M}_i)\,\mathrm{d}\mathbf{w}$$

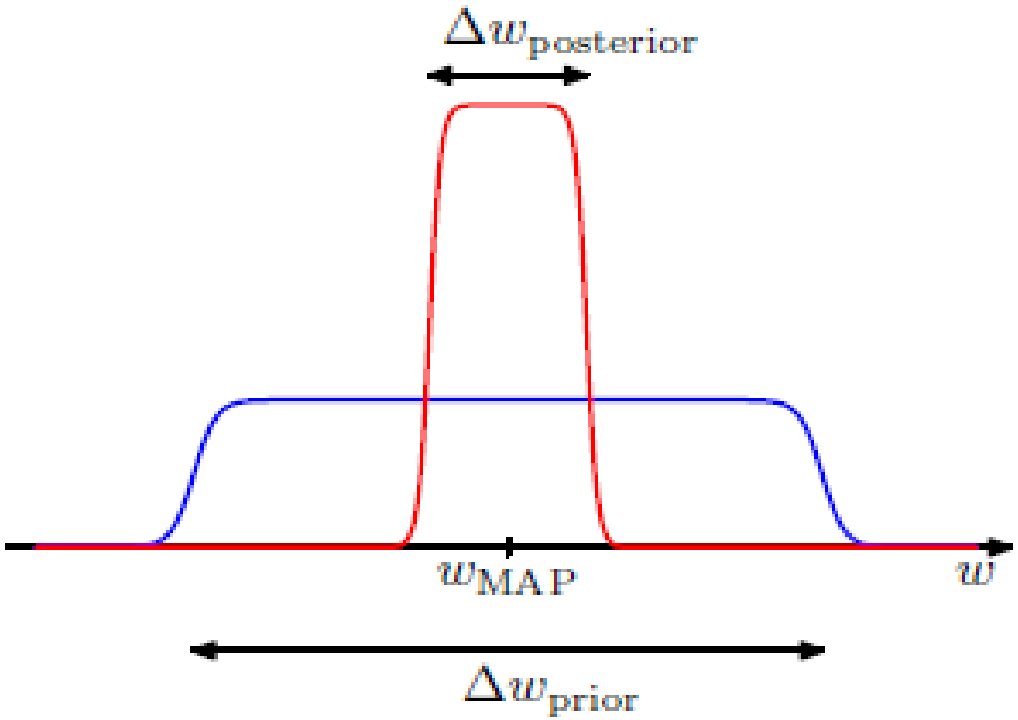
- Probability of generating *D* from a model whose parameters are sampled at random from a prior $p(\mathbf{w}|\mathcal{D},\mathcal{M}_i) = \dfrac{p(\mathcal{D}|\mathbf{w},\mathcal{M}_i)p(\mathbf{w}|\mathcal{M}_i)}{p(\mathcal{D}|\mathcal{M}_i)}$

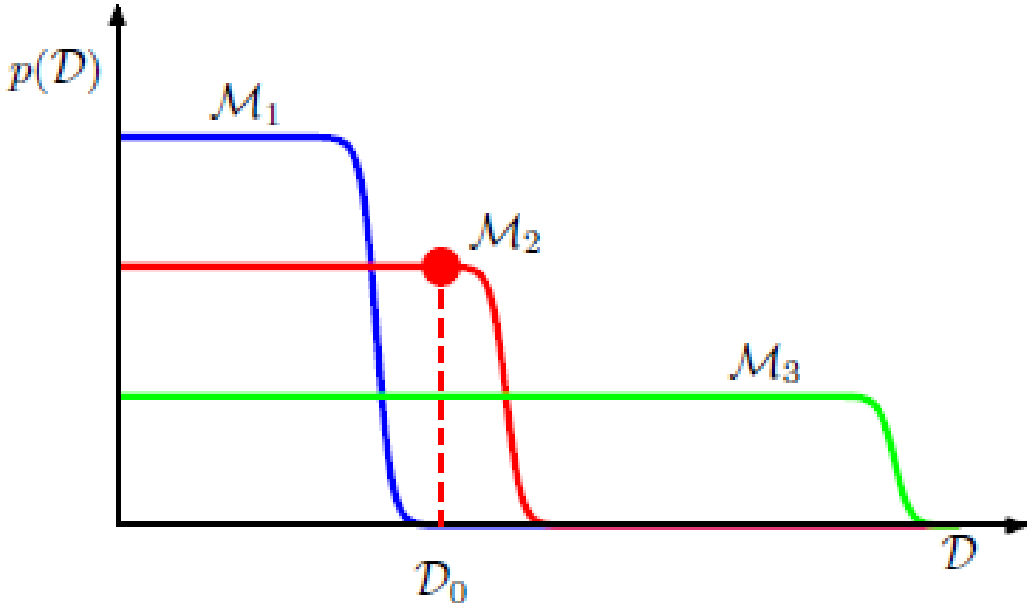
$$p(\mathcal{D}) = \int p(\mathcal{D}|w)p(w)\,\mathrm{d}w \simeq p(\mathcal{D}|w_{\mathrm{MAP}})\frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}}$$

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|w_{\mathrm{MAP}}) + \ln\left(\frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}}\right)$$

Here, $\Delta w_{\mathrm{posterior}} < \Delta w_{\mathrm{prior}}$,model penalty is negative

- Optimal model complexity is given by a trade-off between two competing terms.
  $p(w)\rightarrow p(D|w)\rightarrow p(D)$
- Simple model has little variability $p(D)$ is confined to a small region.
- Complex model can generate a variety of different datasets. $p(D)$ is spread over a large region. Its predictive probability spreads too broad.


$\Delta w_{\text{posterior}}$
w_{MAP}
w
Δw_{prior}


$p(\mathcal{D})$
$\mathcal{M}_1$
$\mathcal{M}_2$
$\mathcal{M}_3$
$\mathcal{D}_0$
$\mathcal{D}$

- Bayesian model comparison on average favor the correct model.

"Expected" Bayes factor: $\int p(\mathcal{D}|\mathcal{M}_1) \ln \frac{p(\mathcal{D}|\mathcal{M}_1)}{p(\mathcal{D}|\mathcal{M}_2)} \mathrm{d}\mathcal{D}$

$\mathcal{M}_1$ correct

$\mathcal{M}_2$ incorrect

Bayes factor

KL Divergence KL $>$ $\int p(\mathcal{D}|\mathcal{M}_1) \ln \frac{p(\mathcal{D}|\mathcal{M}_1)}{p(\mathcal{D}|\mathcal{M}_2)} \mathrm{d}\mathcal{D}$

These approaches need to make assumptions about the form of model.

The Evidence Approximation

- Usually, the complete marginalization over α, β & $\mathbf{w}$ is analytically intractable. we can approximate this by setting the optimal hyperparameters via maximization of marginal likelihood integrating over $\mathbf{w}$.

  $\Longrightarrow$ empirical Bayes or type 2 maximum likelihood or generalized maximum likelihood or "evidence approximation"

- Predictive distribution:

$$p(t|\mathbf{t}) = \iiint p(t|\mathbf{w}, \beta) p(\mathbf{w}|\mathbf{t}, \alpha, \beta) p(\alpha, \beta|\mathbf{t}) \, \mathrm{d}\mathbf{w} \, \mathrm{d}\alpha \, \mathrm{d}\beta$$

Laplace Approximation

- If $p(\alpha, \beta|t) = \delta(\alpha - \hat{\alpha})\delta(\beta - \hat{\beta})$

$$p(t|\mathbf{t}) \simeq p(t|\mathbf{t}, \widehat{\alpha}, \widehat{\beta}) = \int p(t|\mathbf{w}, \widehat{\beta}) p(\mathbf{w}|\mathbf{t}, \widehat{\alpha}, \widehat{\beta})\, \mathrm{d}\mathbf{w}$$

$$(\hat{\alpha}, \hat{\beta}) = \mathrm{argmax}_{(\alpha,\beta)} p(\alpha, \beta|t) = \mathrm{argmax}_{(\alpha,\beta)} p(\alpha, \beta)$$

- Training data only. No cross-validation

Marginal over **w** is intractable using *t*-distribution.

$\Longrightarrow$ Laplace Approximation (local Gaussian approximation)

- This approximation is centered on the mode of posterior distribution.

Poor approximation $\Longleftarrow$ " skewed " mode

CONTENTS

LINEAR MODELS FOR CLASSIFICATION

- Goal : take an input vector x and to assign it to one of k discrete classes $\mathcal{C}_k$ where $k = 1, \ldots, k$

  decision boundaries or decision surfaces, linearly separable

  Assignment of t : $k = 5$ $\quad x \epsilon c_2 \quad$ $\mathbf{t} = (0, 1, 0, 0, 0)^{\mathrm{T}}$

  Three approaches : 1. construct *discriminant function* $y(x)$ c_k

  2. find $p(c_k|x)$ & make decision

  3. use generative model $p(x|c_k)$ &

  prior probabilities $p(\mathcal{C}_k)$ & $p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}$

- Linear regression model $y(\mathbf{x}) = (\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0)$

$\longrightarrow$ $y(\mathbf{x}) = f\left(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0\right)$ $\longleftrightarrow$ link function

for classification

$0 \leq y(\mathbf{x}) \leq 1$

activation function
nonlinear

Generalized Linear model

Discriminant Functions

- Two classes $y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0 \geq 0, y \in c_1$, Decision boundary $y(\mathbf{x}) = 0$ $\leq 0, y \in c_2$
- If $\boldsymbol{x}_A$ & $\boldsymbol{x}_B$ lie in hyperplane $y(\mathbf{x}_\mathrm{A}) = y(\mathbf{x}_\mathrm{B}) = 0$

$$\mathbf{w}^\mathrm{T}(\mathbf{x}_\mathrm{A} - \mathbf{x}_\mathrm{B}) = 0$$

- Normal distance of $\hat{\mathbf{x}}$ to $y(\mathbf{x}) = 0$

$$||\hat{\mathbf{x}}||\cos\theta = ||\hat{\mathbf{x}}||\frac{\hat{\mathbf{x}^T}W}{||\hat{\mathbf{x}}||||W||}$$

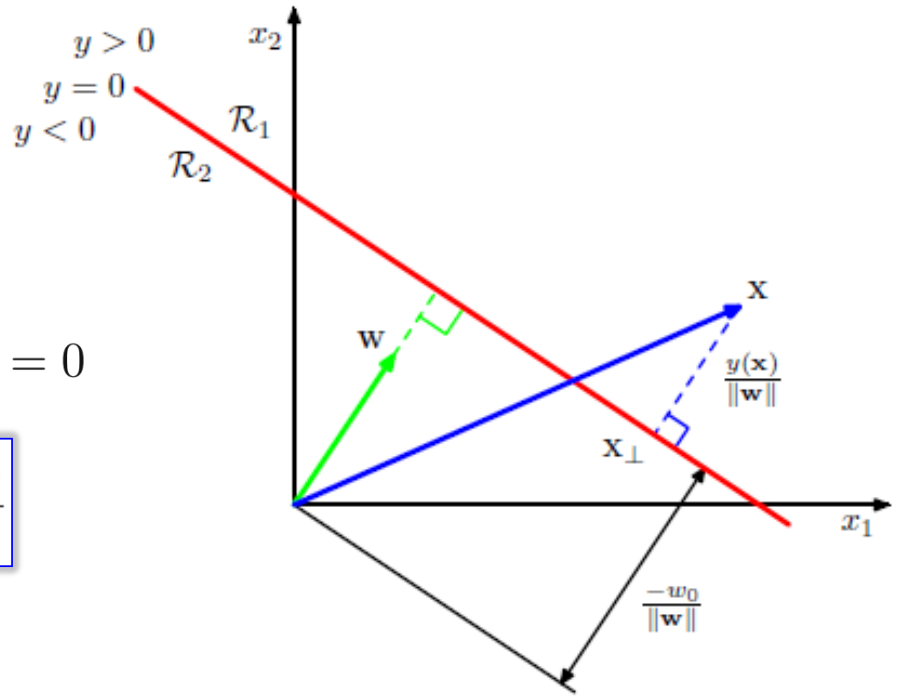
$$= \frac{\mathbf{w}^\mathrm{T}\mathbf{x}}{\|\mathbf{w}\|} = -\frac{w_0}{\|\mathbf{w}\|}$$

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|} \quad y(\mathbf{x}_\perp) = \mathbf{w}^\mathrm{T}\mathbf{x}_\perp + w_0 = 0$$

$$W^T\mathbf{x} = W^T\mathbf{x}_\perp + \gamma\frac{W^TW}{||W||} \implies r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}$$

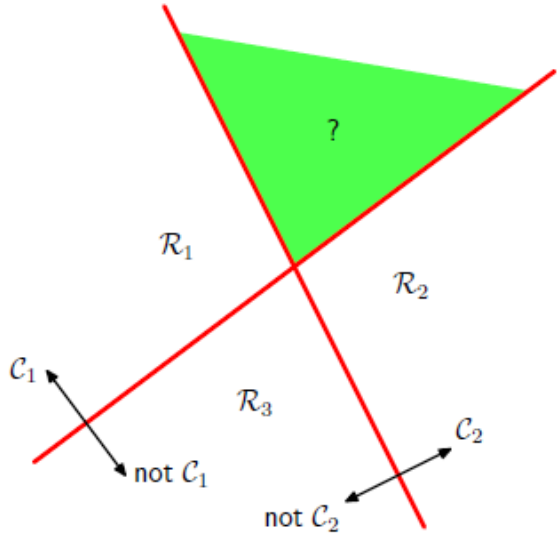
$$= -w_0 + \gamma\frac{W^TW}{||W||}$$

$$y(\mathbf{x}) = \widetilde{\mathbf{w}}^\mathrm{T}\widetilde{\mathbf{x}} \qquad \widetilde{\mathbf{w}} = (w_0, \mathbf{w}) \qquad \widetilde{\mathbf{x}} = (x_0, \mathbf{x})$$

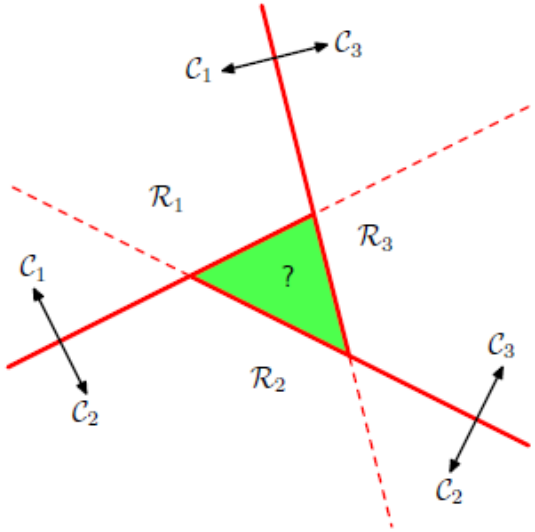


Multiple Classes

$k > 2$ Some difficulties happen when combining a number of two-class discriminant functions.



One-verus-the-rest classifier

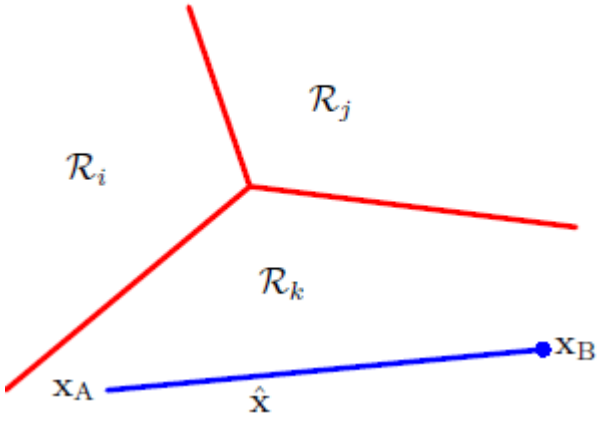


One-verus-one classifier

- We can avoid these difficulties by considering a single *K*-class discriminant comprising *K* linear functions

$$y_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0}$$

$$\hat{\mathbf{x}} = \lambda \mathbf{x}_{\mathrm{A}} + (1-\lambda)\mathbf{x}_{\mathrm{B}} \qquad y_k(\hat{\mathbf{x}}) = \lambda y_k(\mathbf{x}_{\mathrm{A}}) + (1-\lambda) y_k(\mathbf{x}_{\mathrm{B}})$$



- X is assigned to C_k if $y_k(\hat{\mathbf{x}}) > y_j(\hat{\mathbf{x}})$ for all $j \neq k$

  decision boundary C_k and C_j is given by

$$y_k(\mathbf{x}) = y_j(\mathbf{x}) \Longrightarrow (\mathbf{w}_k - \mathbf{w}_j)^{\mathrm{T}}\mathbf{x} + (w_{k0} - w_{j0}) = 0$$

  $(D-1)$-dimensional hyperplane

Three approaches to learning the parameters of linear discriminant function

Least Squares for Classification

- *k* classes Least squares approximate the conditional expectation

  $\mathbb{E}[\mathbf{t}|\mathbf{x}]$ →class posterior prob.

  Closed-form solution to regression is available.

- We have $y_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0} \qquad k = 1, \ldots, K$

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathrm{T}}\widetilde{\mathbf{x}}$$

$\mathbf{x} \in C_k$ if $y_k = \widetilde{\mathbf{w}}_k^{\mathrm{T}}\widetilde{\mathbf{x}}$ is the largest .

- Given a training set $\{X_n, t_n\}_{n=1}^N$, the sum-of-squares error function can be written by

$$\tilde{X} = \begin{bmatrix} -\tilde{X}_1^T \\ \vdots \\ -\tilde{X}_N^T \end{bmatrix} \qquad \tilde{W} = [\tilde{W}_1 \quad \cdots \quad \tilde{W}_k] \qquad \tilde{T} = \begin{bmatrix} -\tilde{t}_1^T \\ \vdots \\ -\tilde{t}_N^T \end{bmatrix}$$

$$E_D(\widetilde{\mathbf{W}}) = \frac{1}{2}\mathrm{Tr}\left\{(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})\right\}$$

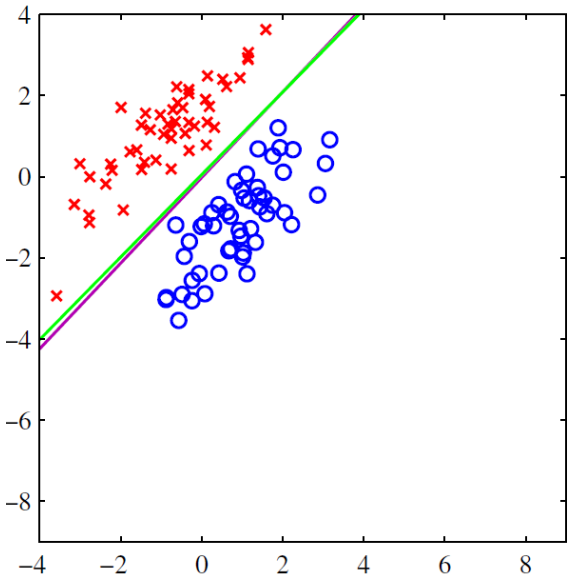
$$\nabla_{\tilde{W}} E_D = 0 \implies \widetilde{\mathbf{W}} = (\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^{\mathrm{T}}\mathbf{T} = \widetilde{\mathbf{X}}^{\dagger}\mathbf{T}$$

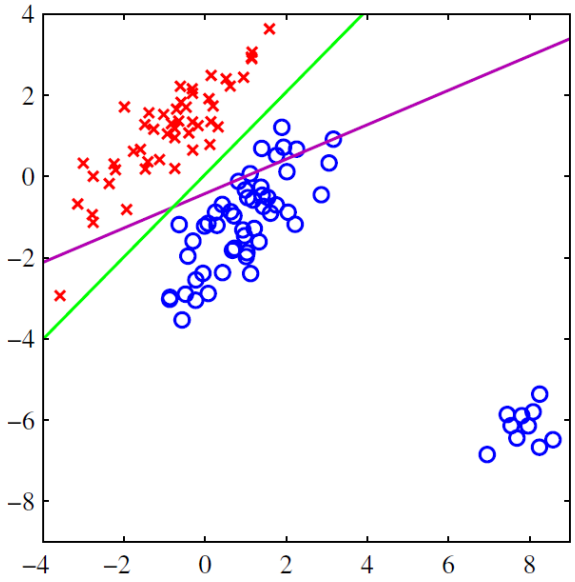
- Discriminant function is obtained by

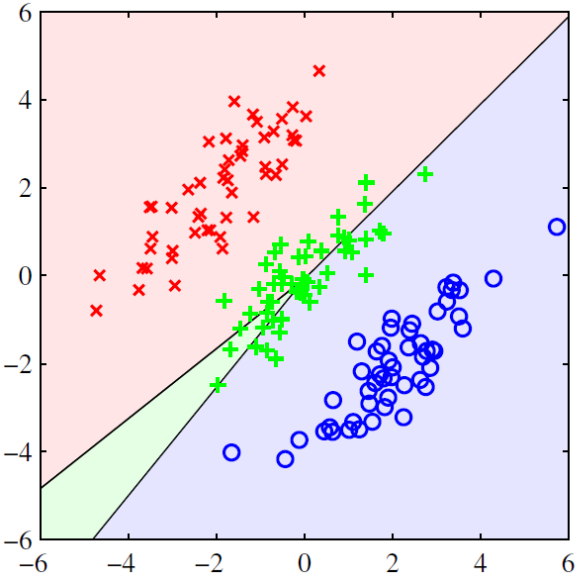
$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathrm{T}}\widetilde{\mathbf{x}} = \mathbf{T}^{\mathrm{T}}\left(\widetilde{\mathbf{X}}^{\dagger}\right)^{\mathrm{T}}\widetilde{\mathbf{x}}$$

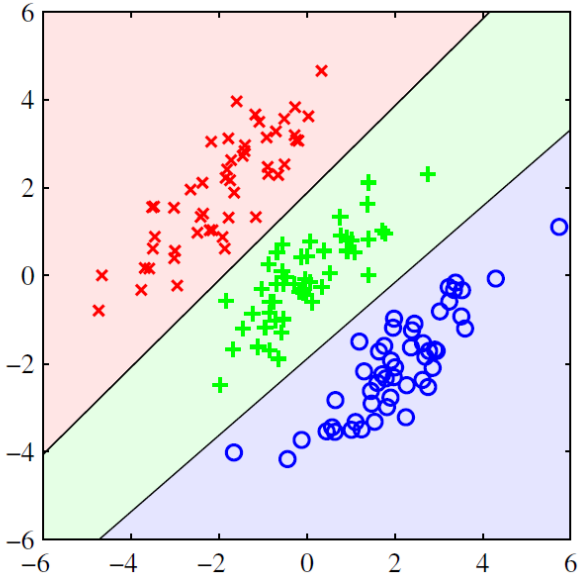
- Least squares is highly sensitive to outliers.

- ML under the assumption of a Gaussian conditional distribution whereas binary target vectors clearly have a distribution far from Gaussian. More appropriate probabilistic models are needed.









Comparison of least-squares discriminant & logistic regression model

Fisher's Linear Discriminant

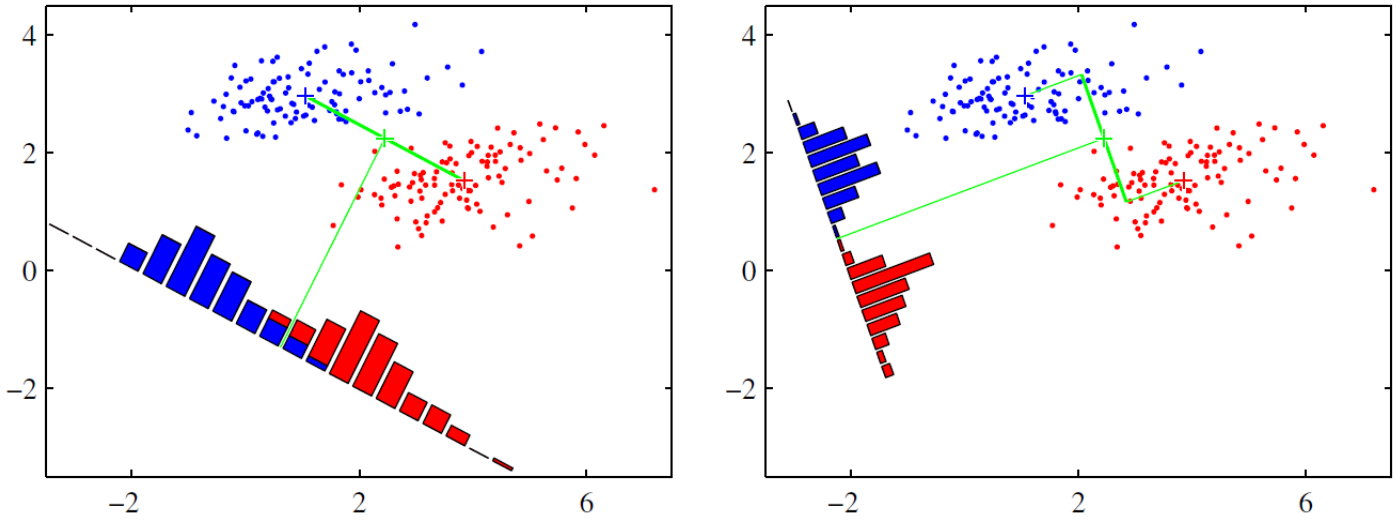
- Dimensionality reduction $y = w^T \mathbf{x} \gtrless_{C_2}^{C_1} w_0$
- We can select a projection that maximizes the class separation.

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in \mathcal{C}_1} \mathbf{x}_n \qquad \mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in \mathcal{C}_2} \mathbf{x}_n$$

$m_2 - m_1 = \mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)$ separation of the projected class mean.

Optimize separation under constraint $\sum_i w_i^2 = 1$,

we obtain $\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1)$



- Fisher's criterion is to maximize a function that gives a large separation between the projected class means while also gives small variance within each class.

- within-class variance $s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$

Fisher criterion $$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{\mathbf{w}^{\mathrm{T}} \mathbf{S}_{\mathrm{B}} \mathbf{w}}{\mathbf{w}^{\mathrm{T}} \mathbf{S}_{\mathrm{W}} \mathbf{w}}$$

where $$\mathbf{S}_{\mathrm{B}} = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}}$$

$$\mathbf{S}_{\mathrm{W}} = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^{\mathrm{T}} + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^{\mathrm{T}}$$

$$\nabla_N J(\mathbf{w}) = 0 \qquad (\mathbf{w}^{\mathrm{T}} \mathbf{S}_{\mathrm{B}} \mathbf{w}) \mathbf{S}_{\mathrm{W}} \mathbf{w} = (\mathbf{w}^{\mathrm{T}} \mathbf{S}_{\mathrm{W}} \mathbf{w}) \mathbf{S}_{\mathrm{B}} \mathbf{w}$$

$\mathbf{S}_{\mathrm{B}} \mathbf{w}$ is in the direction of $\mathbf{m}_2 - \mathbf{m}_1$

Solution to Fisher's Linear Discriminant

We don't care about magnitude of **w**, only its direction. Then, we obtain

$$\mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

If within-class covariance is isotropic, $S_{\mathbf{w}} \propto I$

$$W \propto (m_2 - m_1)$$

Classification Rule $y(\mathbf{x}) \gtrless_{C_2}^{C_1} y_0$ (threshold)

we can assume $p(y|\mathcal{C}_k)$ & use Gaussian
use ML to determine y_0

Probabilistic Generative Models

$$p(\mathcal{C}_1|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)} = \frac{1}{1+\exp(-a)} = \sigma(a)$$

where $a = \ln \dfrac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)} = \ln\left[p(\mathcal{C}_1|\mathbf{x})/p(\mathcal{C}_2|\mathbf{x})\right]$

$\sigma(-a) = 1 - \sigma(a)$ symmetric

$a = \ln\left(\dfrac{\sigma}{1-\sigma}\right)$ logistic sigmoid

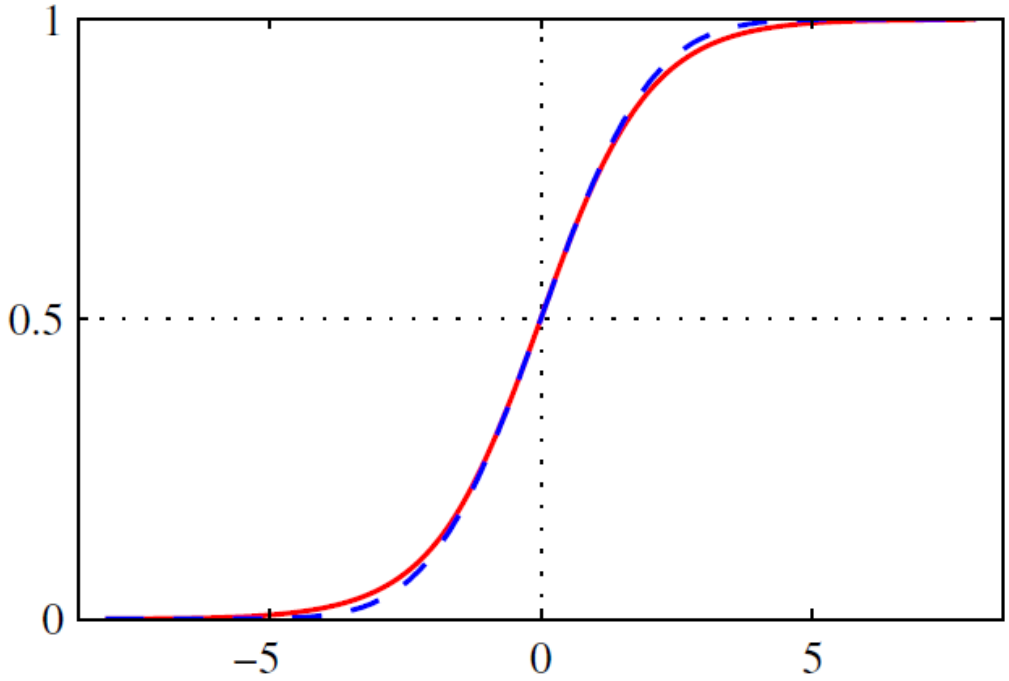
For $K > 2$

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)} = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

normalized exponential

or softmax function

$a_k = \ln p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)$



Plot of the logistic sigmoid function $\sigma(a)$ defined by (4.59), shown in red, together with the scaled probit function $\Phi(\lambda a)$, for $\lambda^2 = \pi/8$, shown in dashed blue, where $\Phi(a)$ is defined by (4.114). The scaling factor $\pi/8$ is chosen so that the derivatives of the two curves are equal for $a = 0$.

$$\Phi(a) = \int_{-\infty}^{a} \mathcal{N}(\theta|0,1)\,\mathrm{d}\theta$$

Continuous Inputs

- Class-conditional densities

$$p(\mathbf{x}|\mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right\}$$

- Considering two classes

$$p(\mathcal{C}_1|\mathbf{x}) = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0)$$

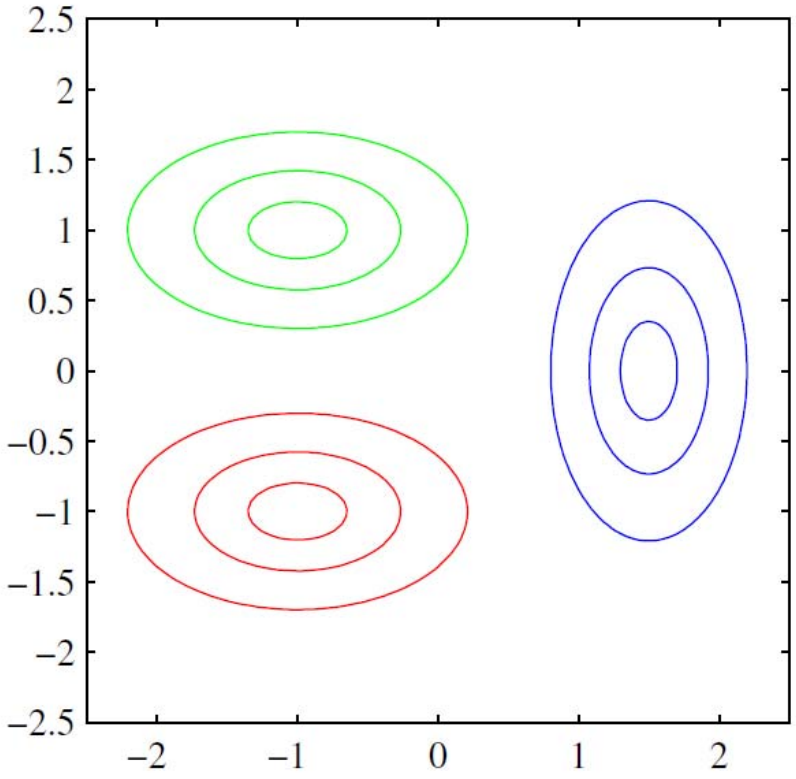
$$\begin{aligned} \mathbf{w} &= \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \\ w_0 &= -\frac{1}{2}\boldsymbol{\mu}_1^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 + \ln\frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)} \end{aligned}$$

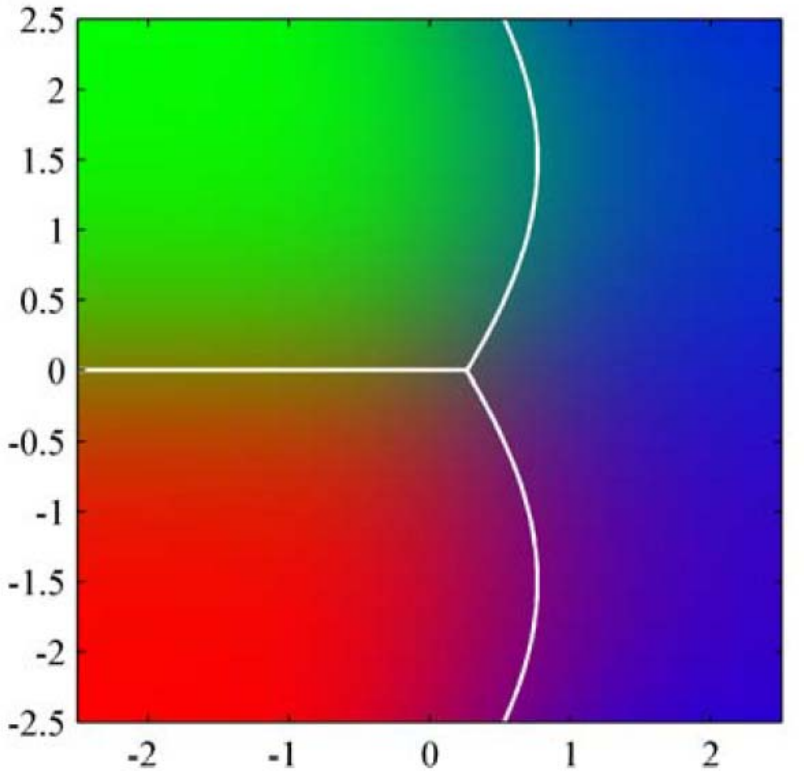
- Class-conditional density ↔posterior probability
- For the general case of *K* classes, we have

$$\begin{aligned} a_k(\mathbf{x}) &= \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0} \\ \mathbf{w}_k &= \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k \\ w_{k0} &= -\frac{1}{2}\boldsymbol{\mu}_k^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \ln p(\mathcal{C}_k) \end{aligned}$$

Decision boundaries when two posterior are equal.

→minimum misclassification rate





Maximum Likelihood Solution

Gaussian class-conditional density $\{\mathbf{x}_n, t_n\}_{n=1}^N$ $t_n = 1$ $\mathcal{C}_1$, $t_n = 0$ $\mathcal{C}_2$

$$p(\mathcal{C}_1) = \pi \qquad p(\mathcal{C}_2) = 1 - \pi$$

$$p(\mathbf{x}_n, \mathcal{C}_1) = p(\mathcal{C}_1)p(\mathbf{x}_n|\mathcal{C}_1) = \pi\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}) \qquad t_n = 1$$

$$p(\mathbf{x}_n, \mathcal{C}_2) = p(\mathcal{C}_2)p(\mathbf{x}_n|\mathcal{C}_2) = (1 - \pi)\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}) \qquad t_n = 0$$

Likelihood function $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$

$$p(\mathbf{t}, \mathbf{X}|\pi, \mu_1, \mu_2, \boldsymbol{\Sigma}) = \prod_{n=1}^{N} \left[\pi\mathcal{N}(\mathbf{x_n}|\mu_1, \boldsymbol{\Sigma})\right]^{t_n} \left[(1 - \pi)\mathcal{N}(\mathbf{x_n}|\mu_2, \boldsymbol{\Sigma})\right]^{1-t_n}$$

$$\frac{\partial p(\mathbf{t}, \mathbf{X}|\pi, \mu_1, \mu_2, \boldsymbol{\Sigma})}{\partial \pi} = 0$$

The functions depending on π are $\sum_{n=1}^{N} \{t_n \ln \pi + (1 - t_n) \ln(1 - \pi)\}$

We obtain $$\pi = \frac{1}{N}\sum_{n=1}^{N} t_n = \frac{N_1}{N} = \frac{N_1}{N_1 + N_2}$$

$\boldsymbol{\mu}_1$: $\boldsymbol{\mu}_2$ $$\sum_{n=1}^{N} t_n \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}) = -\frac{1}{2}\sum_{n=1}^{N} t_n(\mathbf{x}_n - \boldsymbol{\mu}_1)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_1) + \text{const}$$

$\nabla_{\mu_1} p(\mathbf{t}|\pi, \mu_1, \mu_2, \Sigma) = 0$ We obtain $\boxed{\boldsymbol{\mu}_1 = \frac{1}{N_1}\sum_{n=1}^{N} t_n \mathbf{x}_n}$ sample mean of $\mathcal{C}_1$

$\nabla_{\mu_2} p(\mathbf{t}|\pi, \mu_1, \mu_2, \Sigma) = 0$ We obtain $\boxed{\boldsymbol{\mu}_2 = \frac{1}{N_2}\sum_{n=1}^{N} (1 - t_n)\mathbf{x}_n}$ sample mean of $\mathcal{C}_2$

$\boldsymbol{\Sigma}$:

$$-\frac{1}{2}\sum_{n=1}^{N} t_n \ln|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{n=1}^{N} t_n(\mathbf{x}_n - \boldsymbol{\mu}_1)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_1)$$

$$-\frac{1}{2}\sum_{n=1}^{N} (1 - t_n) \ln|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{n=1}^{N} (1 - t_n)(\mathbf{x}_n - \boldsymbol{\mu}_2)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_2)$$

$$= -\frac{N}{2}\ln|\boldsymbol{\Sigma}| - \frac{N}{2}\mathrm{Tr}\left\{\boldsymbol{\Sigma}^{-1}\mathbf{S}\right\}$$

$$\mathbf{S} = \frac{N_1}{N}\mathbf{S}_1 + \frac{N_2}{N}\mathbf{S}_2 \qquad \mathbf{S}_1 = \frac{1}{N_1}\sum_{n\in\mathcal{C}_1}(\mathbf{x}_n - \boldsymbol{\mu}_1)(\mathbf{x}_n - \boldsymbol{\mu}_1)^{\mathrm{T}}$$

$$\mathbf{S}_2 = \frac{1}{N_2}\sum_{n\in\mathcal{C}_2}(\mathbf{x}_n - \boldsymbol{\mu}_2)(\mathbf{x}_n - \boldsymbol{\mu}_2)^{\mathrm{T}}$$

weighted average of covariance matrices

$\nabla_{\Sigma^{-1}} p(\mathbf{t}|\pi, \mu_1, \mu_2, \Sigma) = 0$ $\boxed{\boldsymbol{\Sigma} = \mathbf{S}}$

Maximum likelihood solution is not robust to outliers.

Probabilistic Discriminative Models

- Indirect : $p(\mathbf{x}|\mathcal{C}_k)$ → Maximum likelihood → $p(\mathcal{C}_k|\mathbf{x})$ → Decision $p(\mathcal{C}_k)$
- Direct: $p(\mathcal{C}_k|\mathbf{x})$ → *D*iscriminative training → Fewer adaptive parameters → Predictive performance is improved

Logistic regression

- Generalized linear model, using fixed basis functions

$$p(\mathcal{C}_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\right) \qquad p(\mathcal{C}_2|\boldsymbol{\phi}) = 1 - p(\mathcal{C}_1|\boldsymbol{\phi})$$

- In case of Gaussian class conditional density, mean $2M$ shared covariance Matrix $M(M+1)/2$
- Total number of parameters $M(M+5)/2+1$ quadratic function of M
- Using maximum likelihood, we should calculate

$$\frac{d\sigma}{da} = \sigma(1-\sigma)$$

- Given $\{\phi_n, t_n\}_{n=1}^N, t_n \in \{0,1\}, \phi_n = \phi(x_n)$, likelihood function has the form

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n} \qquad y_n = p(\mathcal{C}_1|\boldsymbol{\phi}_n)$$

$$y_n = \sigma(a_n) \qquad a_n = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n \qquad \mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$$

- We generate $E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\boldsymbol{\phi}_n$$

the same form as the gradient of the sum-of-squares error function for the linear regression in

$\Longrightarrow$ sequential learning algorithm can be presented.

Iterative Reweighted Least Squares

- No closed-form solution is available due to the nonlinearity of the logistic sigmoid function.
- Error function can be minimized by *Newton-Raphson* iterative optimization

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \mathbf{H}^{-1}\nabla E(\mathbf{w})$$

local quadratic approximation to log likelihood function

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n - t_n)\boldsymbol{\phi}_n = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\mathbf{w} - \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N}\boldsymbol{\phi}_n\boldsymbol{\phi}_n^{\mathrm{T}} = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$$

- Newton-Raphson updating:

$$\begin{aligned}\mathbf{w}^{(\text{new})} &= \mathbf{w}^{(\text{old})} - (\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi})^{-1}\left\{\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\mathbf{w}^{(\text{old})} - \mathbf{\Phi}^{\mathrm{T}}\mathbf{t}\right\} \\ &= (\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}\end{aligned}$$

- In case of cross-entropy error function,

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}(y_n - t_n)\phi_n = \mathbf{\Phi}^{\mathrm{T}}(\mathbf{y} - \mathbf{t})$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n(1 - y_n)\phi_n\phi_n^{\mathrm{T}} = \mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi}$$

- Weighting matrix

$$\mathbf{R}_{N\times N} = [R_{nn}] \quad \text{where} \quad R_{nn} = y_n(1 - y_n)$$

$$\begin{aligned}\mathbf{w}^{(\text{new})} &= \mathbf{w}^{(\text{old})} - (\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathrm{T}}(\mathbf{y} - \mathbf{t}) \\ &= (\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi})^{-1}\left\{\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi}\mathbf{w}^{(\text{old})} - \mathbf{\Phi}^{\mathrm{T}}(\mathbf{y} - \mathbf{t})\right\} \\ &= (\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{z}\end{aligned}$$

where $\mathbf{z} = \mathbf{\Phi}\mathbf{w}^{(\text{old})} - \mathbf{R}^{-1}(\mathbf{y} - \mathbf{t})$

$\mathbf{w} \rightarrow \mathbf{R} \rightarrow \mathbf{w} \rightarrow \mathbf{R} \rightarrow \ldots \Longrightarrow$ iterative reweighted least squares

Laplace Approximation

- Bayesian model comparison → evidence $Z = \int f(\mathbf{z})\,\mathrm{d}\mathbf{z}$
- Data set D a set of models $\{\mathcal{M}_i\}$ parameter $\{\boldsymbol{\theta}_i\}$
- Model evidence $p(\mathcal{D}|\mathcal{M}_i)$

  or $p(\mathcal{D}) = \int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}$ $\qquad \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} = p(\theta|\mathcal{D}) = \frac{f(\theta)}{[Z=\int f(\theta)d(\theta)]}$

$$\left.\begin{aligned} f(\boldsymbol{\theta}) &= p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \\ Z &= p(\mathcal{D}) \end{aligned}\right\} \quad Z \simeq f(\mathbf{z}_0)\frac{(2\pi)^{M/2}}{|\mathbf{A}|^{1/2}}$$

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\boldsymbol{\theta}_{\mathrm{MAP}}) + \underbrace{\ln p(\boldsymbol{\theta}_{\mathrm{MAP}}) + \frac{M}{2}\ln(2\pi) - \frac{1}{2}\ln|\mathbf{A}|}_{\text{Occam factor}}$$

$$\mathbf{A} = -\nabla\nabla \ln p(\mathcal{D}|\boldsymbol{\theta}_{\mathrm{MAP}})p(\boldsymbol{\theta}_{\mathrm{MAP}}) = -\nabla\nabla \ln p(\boldsymbol{\theta}_{\mathrm{MAP}}|\mathcal{D})$$

- Consider the Gaussian prior distribution is broad, and the Hessian has full rank, we have

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\boldsymbol{\theta}_{\mathrm{MAP}}) - \frac{1}{2}M\ln N$$

- *Bayesian Information Criterion* (BIC) (Schwarz, 1978)

Bayesian Logistic Regression

- Exact Bayesian inference for logistic regression is intractable including evaluation of the posterior distribution.

Gaussian Approximation

- Gaussian prior $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0)$
- Posterior $p(\mathbf{w}|\mathbf{t}) \propto p(\mathbf{w})p(\mathbf{t}|\mathbf{w})$

$$\ln p(\mathbf{w}|\mathbf{t}) = -\frac{1}{2}(\mathbf{w}-\mathbf{m}_0)^{\mathrm{T}}\mathbf{S}_0^{-1}(\mathbf{w}-\mathbf{m}_0) + \sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} + \text{const}$$

where $y_n = \sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n)$

- Gaussian approximation $q(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{w}_{\mathrm{MAP}}, \mathbf{S}_N) \rightarrow p(\mathbf{w}|\mathbf{t})$

$$\frac{d}{d\mathbf{w}} \ln p(\mathbf{w}|\mathbf{t})\Big|_{\mathbf{w}_{\mathrm{MAP}}} = 0 \qquad \mathbf{S}_N = -\nabla\nabla \ln p(\mathbf{w}|\mathbf{t}) = \mathbf{S}_0^{-1} + \sum_{n=1}^{N} y_n(1-y_n)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^{\mathrm{T}}$$

Predictive Distribution

$$p(\mathcal{C}_1|\boldsymbol{\phi},\mathbf{t}) = \int p(\mathcal{C}_1|\boldsymbol{\phi},\mathbf{w})p(\mathbf{w}|\mathbf{t})\,\mathrm{d}\mathbf{w} \simeq \int \sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})q(\mathbf{w})\,\mathrm{d}\mathbf{w}$$

$$\sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}) = \int \delta(a - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})\sigma(a)\,\mathrm{d}a \qquad \int \sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})q(\mathbf{w})\,\mathrm{d}\mathbf{w} = \int \sigma(a)p(a)\,\mathrm{d}a$$

where $$p(a) = \int \delta(a - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})q(\mathbf{w})\,\mathrm{d}\mathbf{w}$$

$$\mu_a = \mathbb{E}[a] = \int p(a)a\,\mathrm{d}a = \int q(\mathbf{w})\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\,\mathrm{d}\mathbf{w} = \mathbf{w}_{\mathrm{MAP}}^{\mathrm{T}}\boldsymbol{\phi}$$

$$\sigma_a^2 = \mathrm{var}[a] = \int p(a)\left\{a^2 - \mathbb{E}[a]^2\right\}\,\mathrm{d}a$$

$$= \int q(\mathbf{w})\left\{(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})^2 - (\mathbf{m}_N^{\mathrm{T}}\boldsymbol{\phi})^2\right\}\,\mathrm{d}\mathbf{w} = \boldsymbol{\phi}^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}$$

- Variational approximation to the predictive distribution

$$p(\mathcal{C}_1|\mathbf{t}) = \int \sigma(a)p(a)\,\mathrm{d}a = \int \sigma(a)\mathcal{N}(a|\mu_a,\sigma_a^2)\,\mathrm{d}a$$

CONTENTS

Feed-forward Network Functions

- Multilayer Perceptron (MLP)
- Mathematical representations of information processing in biological systems
- Linear models for regression and classification

$$y(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathbf{x})\right)$$

- Neural network model

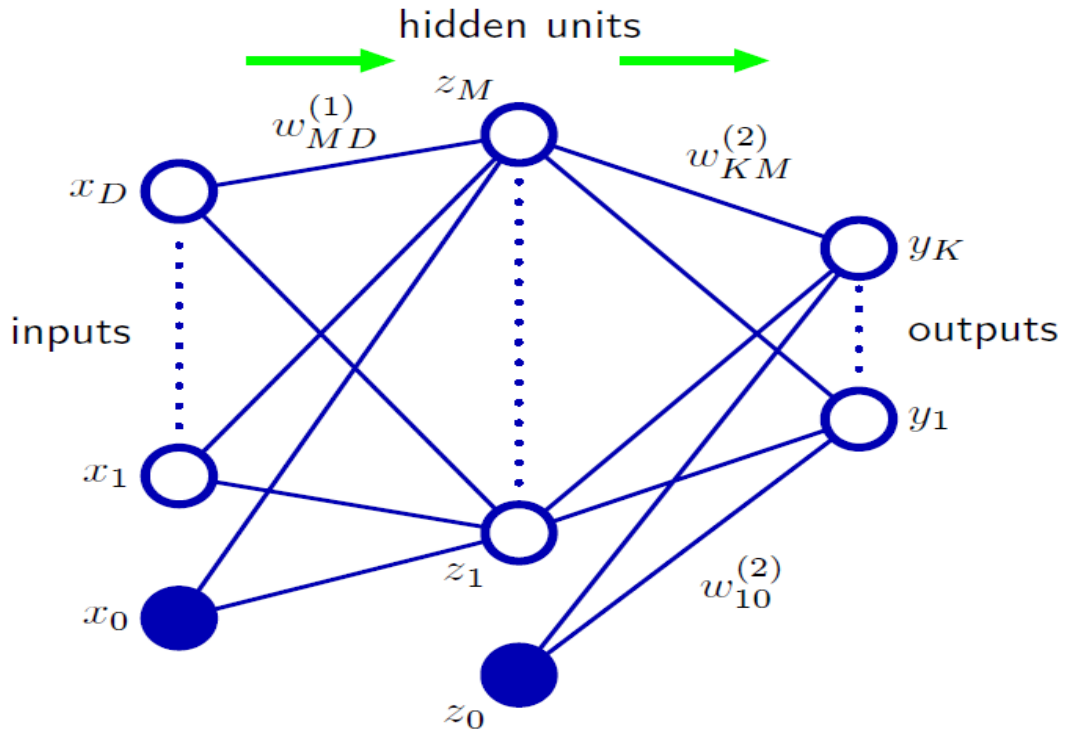
$$x = (x_1, x_2, ..., x_D)^T$$

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}$$

$$z_j = h(a_j)$$

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)}$$

where $j = 1, \cdots, M \quad , \quad k = 1, \cdots, K$



$$y_k = \sigma(a_k)$$

where $\sigma(a) = \dfrac{1}{1+\exp(-a)}$, softmax activation function

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$$

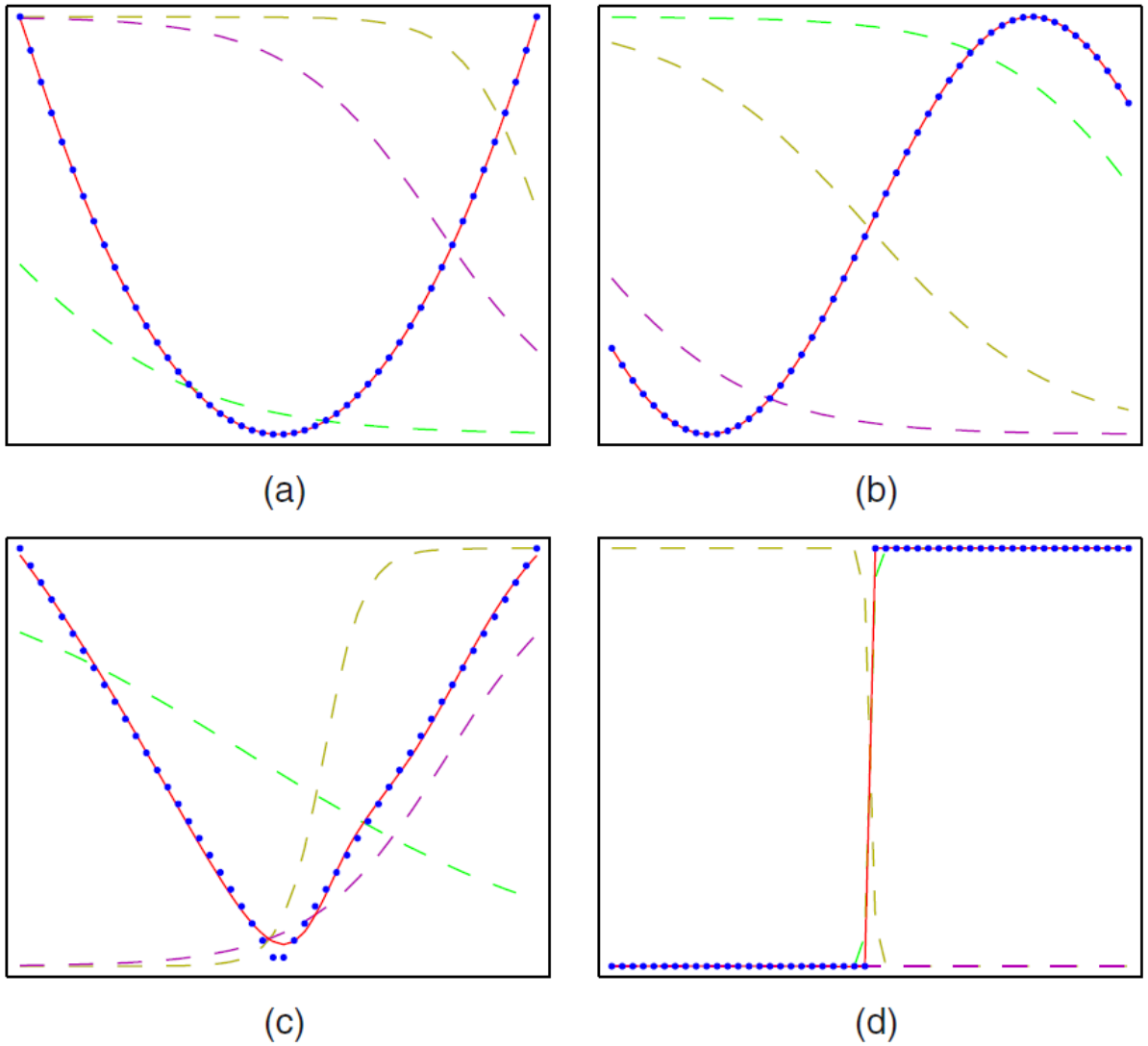
- Neural network model is simply a nonlinear function from a set of inputs $\{x_i\}$ to a set of outputs $\{y_k\}$ controlled by $\mathbf{w} = \{\{w_{ji}^{(1)}\}, \{w_{kj}^{(2)}\}\}$.

  $\Longrightarrow$ deterministic variables for neural network

- We can give the additional input variables $x_0 = 1$, $z_0 = 1$

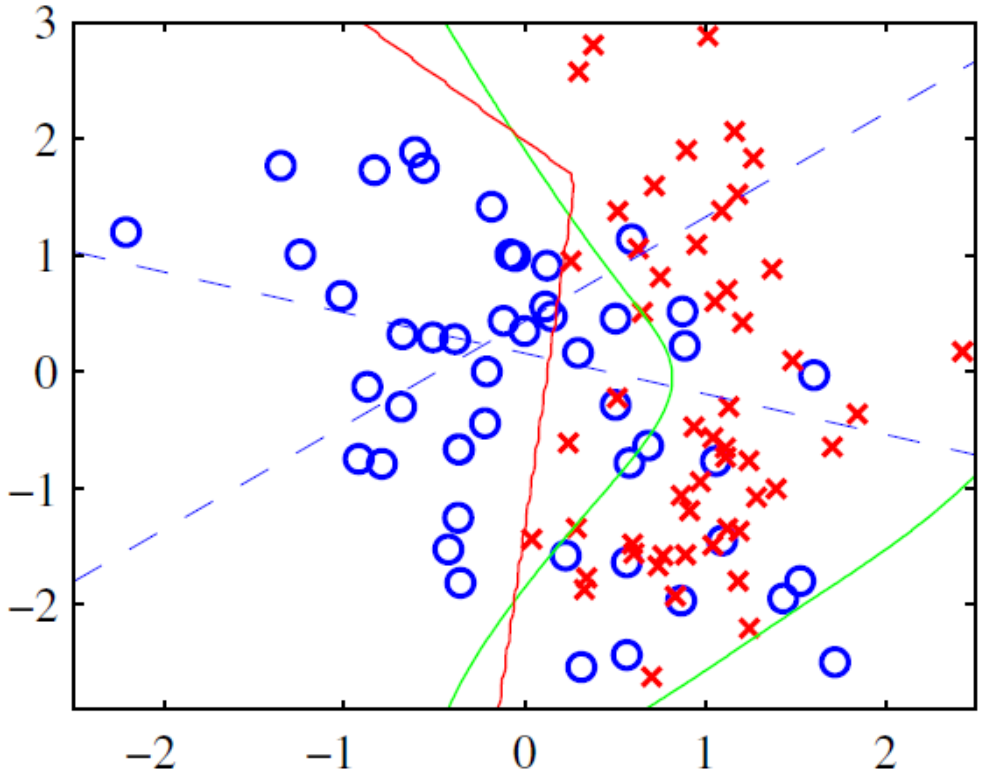
$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i\right)\right)$$

Figure 5.3 Illustration of the capability of a multilayer perceptron to approximate four different functions comprising (a) $f(x) = x^2$, (b) $f(x) = \sin(x)$, (c), $f(x) = |x|$, and (d) $f(x) = H(x)$ where $H(x)$ is the Heaviside step function. In each case, $N = 50$ data points, shown as blue dots, have been sampled uniformly in x over the interval $(-1, 1)$ and the corresponding values of $f(x)$ evaluated. These data points are then used to train a two-layer network having 3 hidden units with 'tanh' activation functions and linear output units. The resulting network functions are shown by the red curves, and the outputs of the three hidden units are shown by the three dashed curves.



Neural networks are said to be "universal approximators"

Figure 5.4 Example of the solution of a simple two-class classification problem involving synthetic data using a neural network having two inputs, two hidden units with 'tanh' activation functions, and a single output having a logistic sigmoid activation function. The dashed blue lines show the $z = 0.5$ contours for each of the hidden units, and the red line shows the $y = 0.5$ decision surface for the network. For comparison, the green line denotes the optimal decision boundary computed from the distributions used to generate the data.



Objective for Network Training

- Given input vectors $\{\mathbf{x}_n\}_{n=1}^N$ and target vector $\{\mathbf{t}_n\}_{n=1}^N$, the minimization of a sum-of-squares error function

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} \|\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n\|^2$$

Relation to maximum likelihood criterion

$$p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right)$$
$$= \mathcal{N}(t - y(\mathbf{x}, \mathbf{w})|0, \beta^{-1})$$

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n|\mathbf{x}_n, \mathbf{w}, \beta)$$

$$\mathbf{w}_{\mathrm{ML}} = \mathrm{argmax}_{\mathbf{w}}\, p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta)$$

Taking the negative logarithm, we have

$$\frac{\beta}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n,\mathbf{w})-t_n\}^2-\frac{N}{2}\ln\beta+\frac{N}{2}\ln(2\pi)$$

ML ⇔ Minimization of sum-of-squares error function

$$E(\mathbf{w})=\frac{1}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n,\mathbf{w})-t_n\}^2$$

$$\frac{1}{\beta_{\mathrm{ML}}}=\frac{1}{N}\sum_{n=1}^{N}\{y(\mathbf{x}_n,\mathbf{w}_{\mathrm{ML}})-t_n\}^2$$

- For the case of multiple target variables

$$p(\mathbf{t}|\mathbf{x},\mathbf{w})=\mathcal{N}\left(\mathbf{t}|\mathbf{y}(\mathbf{x},\mathbf{w}),\beta^{-1}\mathbf{I}\right)$$

$$\frac{1}{\beta_{\mathrm{ML}}}=\frac{1}{NK}\sum_{n=1}^{N}\|\mathbf{y}(\mathbf{x}_n,\mathbf{w}_{\mathrm{ML}})-\mathbf{t}_n\|^2$$

Cross Entropy Error Function

- Consider a binary classification, let

$$y(\mathbf{x}, \mathbf{w}) = p(C_1|\mathbf{x})$$

$$1 - y(\mathbf{x}, \mathbf{w}) = p(C_2|\mathbf{x})$$

$$p(t|\mathbf{x}, \mathbf{w}) = y(\mathbf{x}, \mathbf{w})^t \{1 - y(\mathbf{x}, \mathbf{w})\}^{1-t} \qquad \begin{matrix} t = 1 \rightarrow C_1 \\ t = 0 \rightarrow C_2 \end{matrix}$$

- Cross entropy error function

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

- In case of *K* binary classifications,

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \prod_{k=1}^{K} y_k(\mathbf{x}, \mathbf{w})^{t_k} \left[1 - y_k(\mathbf{x}, \mathbf{w})\right]^{1-t_k}$$

- Taking the negative logarithm of the likelihood function then gives the cross-entropy error function

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K}\{t_{nk}\ln y_{nk} + (1-t_{nk})\ln(1-y_{nk})\}$$

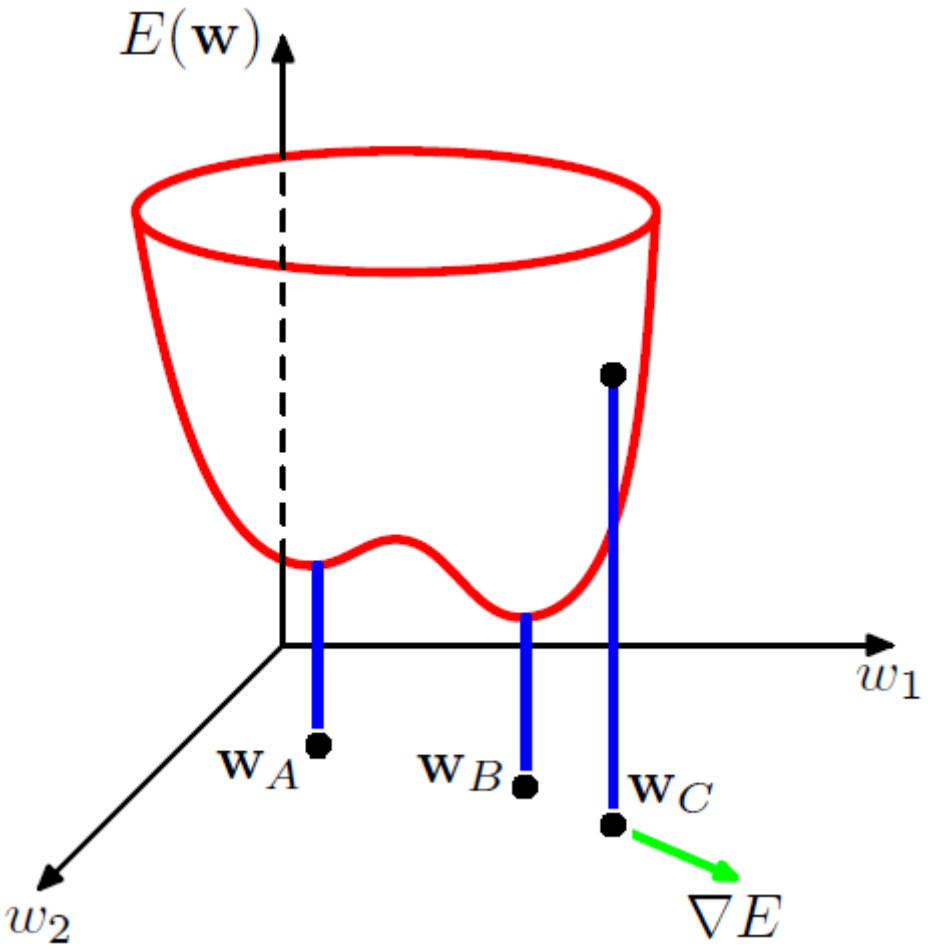
where $y_{nk} = y_k(\mathbf{x}_n, \mathbf{w})$

- In case of standard multiclass classification, 1-of-*K* coding scheme $y_k(\mathbf{x}, \mathbf{w}) = p(t_k = 1|\mathbf{x})$, $\sum_k y_k = 1$

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{kn}\ln y_k(\mathbf{x}_n, \mathbf{w})$$

$$y_k(\mathbf{x}, \mathbf{w}) = \frac{\exp(a_k(\mathbf{x}, \mathbf{w}))}{\sum_j \exp(a_j(\mathbf{x}, \mathbf{w}))}$$

Geometrical view of the error function $E(\mathbf{w})$ as a surface sitting over weight space. Point $\mathbf{w}_A$ is a local minimum and $\mathbf{w}_B$ is the global minimum. At any point $\mathbf{w}_C$, the local gradient of the error surface is given by the vector ∇E.



Parameter optimization

$$\mathbf{w} \rightarrow \mathbf{w} + \delta\mathbf{w}$$

$$\delta E \simeq \delta\mathbf{w}^T \nabla E(\mathbf{w})$$

- Smallest value will occur at a point which $\nabla E(\mathbf{w}) = 0$

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}$$

Local quadratic approximation

$\hat{\mathbf{w}}$ is some point around $\mathbf{w}$, the Taylor expansion

$$E(\mathbf{w}) \simeq E(\widehat{\mathbf{w}}) + (\mathbf{w} - \widehat{\mathbf{w}})^{\mathrm{T}}\mathbf{b} + \frac{1}{2}(\mathbf{w} - \widehat{\mathbf{w}})^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \widehat{\mathbf{w}})$$

$$\mathbf{b} \equiv \nabla E|_{\mathbf{w}=\widehat{\mathbf{w}}}$$

$$\mathbf{H} = \nabla\nabla E \qquad (\mathbf{H})_{ij} \equiv \left.\frac{\partial^2 E}{\partial w_i \partial w_j}\right|_{\mathbf{w}=\widehat{\mathbf{w}}}$$

- We obtain $\nabla E \simeq \mathbf{b} + \mathbf{H}(\mathbf{w} - \widehat{\mathbf{w}})$
- Let $\mathbf{w}^*$ be a local minimum and $(\nabla E)_{\mathbf{w}^*} = 0$

$$E(\mathbf{w}) = E(\mathbf{w}^\star) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^\star)^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \mathbf{w}^\star)$$

- For geometric interpretation, we can perform eigen-analysis

$$\mathbf{H}\mathbf{u}_i = \lambda_i \mathbf{u_i}$$

where $\{\mathbf{u}_i\}$ are orthonormal vectors, i.e., $\mathbf{u}_i^T\mathbf{u}_j = \delta_{ij}$

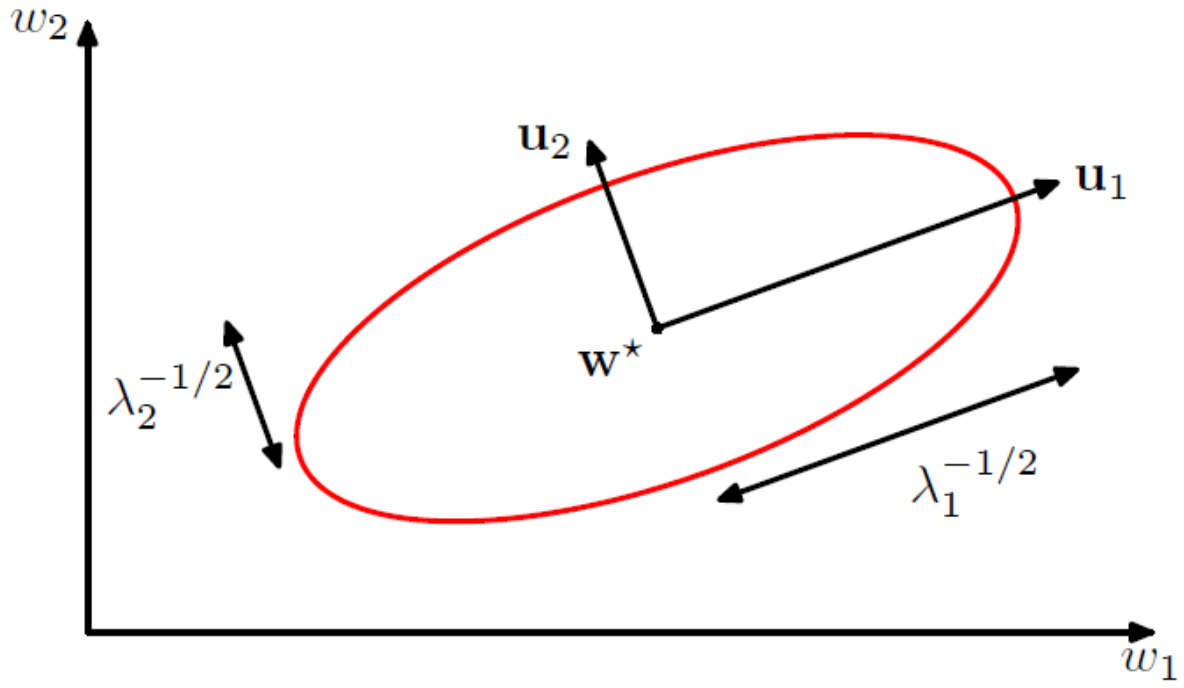
- We expand $\mathbf{w} - \mathbf{w}^\star = \sum_i \alpha_i \mathbf{u}_i$, then

$$E(\mathbf{w}) = E(\mathbf{w}^*) + \tfrac{1}{2}(\textstyle\sum_i \alpha_i \mathbf{u}_i)^{\mathrm{T}}\mathbf{H}(\sum_i \alpha_i \mathbf{u}_i)$$
$$= E(\mathbf{w}^\star) + \frac{1}{2}\sum_i \lambda_i \alpha_i^2.$$

- **H** is positive definite if, and only if,

$$\mathbf{v}^{\mathbf{T}}\mathbf{H}\mathbf{v} > 0 \quad \text{for all} \quad \mathbf{v} \neq 0$$

or if $\{\lambda_i\}$ are all positive.



Gradient Descent Optimization

- Batch gradient descent

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E(\mathbf{w}^{(\tau)})$$

where the parameter η is the learning rate.

- Sequential gradient descent method

If $$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w})$$

then $$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n(\mathbf{w}^{(\tau)})$$

Error Backpropagation Algorithm

- We try to find an efficient way of evaluating $\nabla E(\mathbf{w})$ for a feed-forward neural network.

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w})$$

$$E_n = \frac{1}{2} \sum_k (y_{nk} - t_{nk})^2$$

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j} \frac{\partial a_j}{\partial w_{ji}}$$

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i$$

- In a general feed-forward network, each unit computes a

$$a_j = \sum_i w_{ji} z_i$$

Evaluation of Error-Function Derivatives

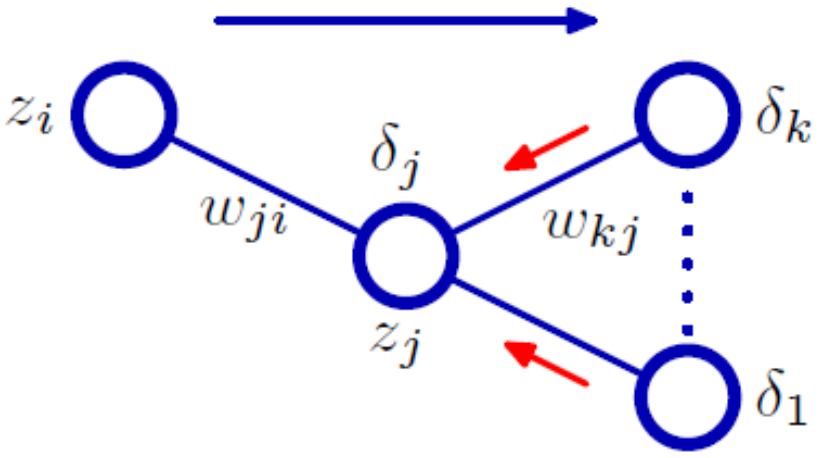
send a connection to j , $z_j = h(a_j)$

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial a_j}$$

$$= h'(a_j) \sum_k w_{kj} \delta_k$$

$$\delta_k = y_k - t_k$$

- Finally, $$\frac{\partial E}{\partial w_{ji}} = \sum_n \frac{\partial E_n}{\partial w_{ji}}$$



Error Backpropagation Procedure

1. Apply $\mathbf{x}_n$ to network by general computation

$$a_j = \sum_i w_{ji} z_i \quad \& \quad z_j = h(a_j)$$

   Find all activations of all hidden and output units

2. Evaluate $\delta_k = y_k - t_k$ for all output nodes

3. Backpropagate the δ's using

$$\delta_j = h'(a_j) \sum_k w_{kj} \delta_k$$

4. Find the required derivatives $\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i$

   for all weights in first layer $\{w_{ji}^{(1)}\}$ and second layer $\{w_{kj}^{(2)}\}$

Logistic Sigmoid Activation Function

$$h(a) \equiv \tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$$

$$h'(a) = \frac{(e^a+e^{-a})(e^a+e^{-a})-(e^a-e^a)(e^a-e^{-a})}{(e^a+e^{-a})^2}$$

$$= 1 - h(a)^2$$

The Hessian Matrix

- Backpropagation algorithm for the first derivatives can be extended to that for the second derivatives of the error, given by

$$\frac{\partial^2 E}{\partial w_{ji} \partial w_{lk}}$$

$$\mathbf{H} = [\mathbf{H}_{ij}] = [\frac{\partial^2 E}{\partial w_i \partial w_j}]_{\mathbf{w} \times \mathbf{w}}$$

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \mathbf{H}^{-1} \nabla E(\mathbf{w}^{(\tau)})$$

- Hessian matrix plays an important role in many aspects of neural computing. Various approximation schemes have been used to evaluate the Hessian matrix for a neural network.
- Hessian exact computation is also available with cost $O(W^2)$ where W is the number of weights.

Diagonal Approximation

- For this case, the diagonal elements are computed by

$$\frac{\partial^2 E_n}{\partial w_{ji}^2} = \frac{\partial^2 E_n}{\partial a_j^2} z_i^2 \qquad \left(\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j} \frac{\partial a_j}{\partial w_{ji}} \right)$$

$$\delta_j = \frac{\partial E_n}{\partial a_j} = h'(a_j) \sum_k w_{kj} \delta_k$$

$$\frac{\partial^2 E_n}{\partial a_j^2} = h'(a_j)^2 \sum_k \sum_{k'} w_{kj} w_{k'j} \frac{\partial^2 E_n}{\partial a_k \partial a_{k'}} + h''(a_j) \sum_k w_{kj} \frac{\partial E_n}{\partial a_k}$$

- If we neglect off-diagonal elements in the second-derivative terms, we obtain

$$\frac{\partial^2 E_n}{\partial a_j^2} = h'(a_j)^2 \sum_k w_{kj}^2 \frac{\partial^2 E_n}{\partial a_k^2} + h''(a_j) \sum_k w_{kj} \frac{\partial E_n}{\partial a_k}$$

Outer Product Approximation

- In case of a single output

$$E = \frac{1}{2}\sum_{n=1}^{N}(y_n - t_n)^2$$

$$\mathbf{H} = \nabla\nabla E = \sum_{n=1}^{N}\nabla y_n \nabla^{\mathrm{T}} y_n + \sum_{n=1}^{N}(y_n - t_n)\nabla\nabla y_n$$

$$(\nabla E = \textstyle\sum_{n=1}^{N}(y_n - t_n)\nabla y_n)$$

- If the network is well trained, $y_n \to t_n$, we have the outer-product approximation

$$\mathbf{H} \cong \textstyle\sum_{n=1}^{N}(\nabla y_n \nabla^{\mathrm{T}} y_n) = \sum_{n=1}^{N}\mathbf{b}_n\mathbf{b}_n^{\mathrm{T}}$$

Inverse Hessian

- Using outer-product approximation,

$$\mathbf{H}_N = \sum_{n=1}^{N} \mathbf{b}_n \mathbf{b}_n^{\mathrm{T}}$$

where $\mathbf{b}_n \equiv \nabla_{\mathbf{w}} a_n$

- Sequential procedure for building Hessian, for the first L data points,

$$\mathbf{H}_{L+1} = \mathbf{H}_L + \mathbf{b}_{L+1}\mathbf{b}_{L+1}^{\mathrm{T}}$$

- Using the Woodbury identity

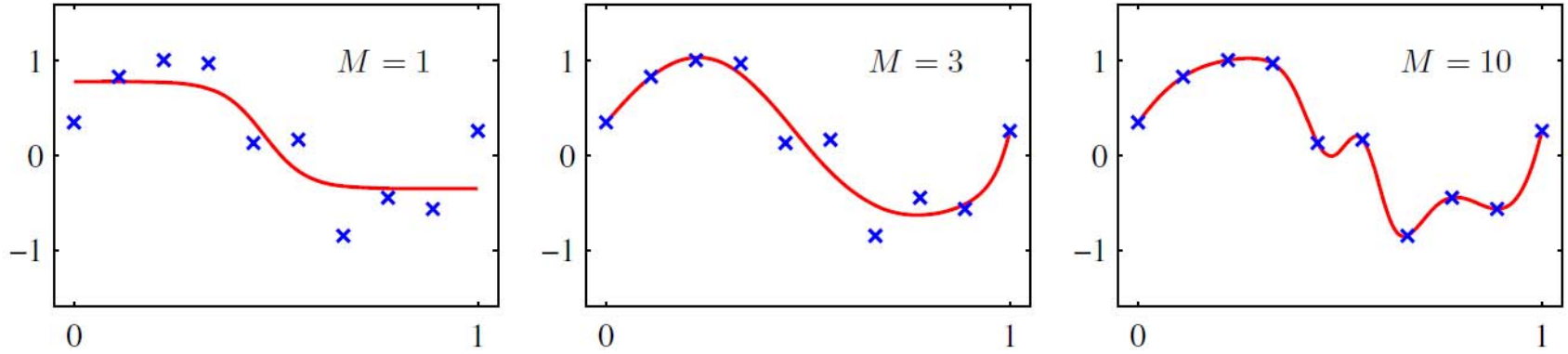
$$\left(\mathbf{M} + \mathbf{v}\mathbf{v}^{\mathrm{T}}\right)^{-1} = \mathbf{M}^{-1} - \frac{\left(\mathbf{M}^{-1}\mathbf{v}\right)\left(\mathbf{v}^{\mathrm{T}}\mathbf{M}^{-1}\right)}{1 + \mathbf{v}^{\mathrm{T}}\mathbf{M}^{-1}\mathbf{v}}$$

we obtain

$$\mathbf{H}_{L+1}^{-1} = \mathbf{H}_L^{-1} - \frac{\mathbf{H}_L^{-1}\mathbf{b}_{L+1}\mathbf{b}_{L+1}^{\mathrm{T}}\mathbf{H}_L^{-1}}{1 + \mathbf{b}_{L+1}^{\mathrm{T}}\mathbf{H}_L^{-1}\mathbf{b}_{L+1}}$$

5.5 Regularization in Neural Networks

- Optimum value of M, number of hidden neurons, gives the best generalization performance or optimum balance between under-fitting and over-fitting.
- Figure 5.9



$$\widetilde{E}(\mathbf{w}) = E(\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

- The above regularizer is also known as weight decay and λ is the regularization coefficient.

5.5.1 Consistent Gaussian priors

- This weight decay is inconsistent with certain scaling properties of network mapping due to it treats all weights and biases on an equal footing, does not satisfy the property of consistency of linear transformation of obtaining equivalent network.
- Regularizer is modified by

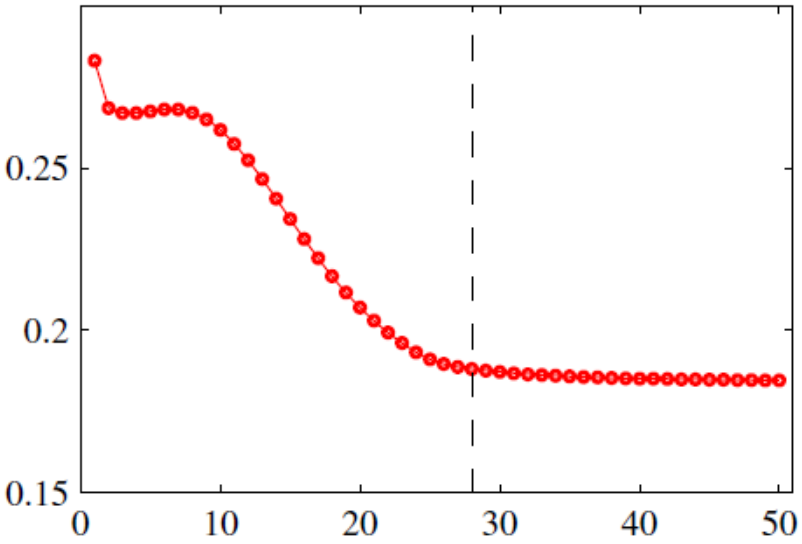
$$\frac{\lambda_1}{2}\sum_{w\in\mathcal{W}_1} w^2 + \frac{\lambda_2}{2}\sum_{w\in\mathcal{W}_2} w^2$$

- This corresponds to merge a prior of the form

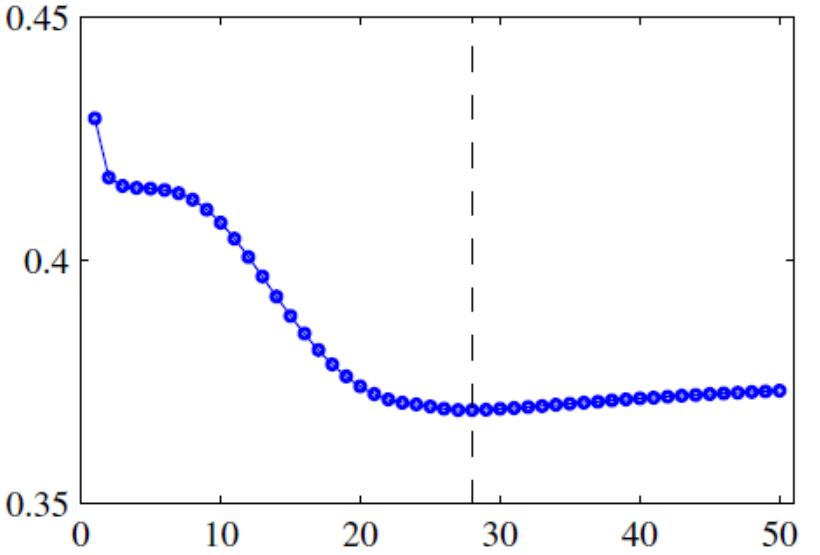
$$p(\mathbf{w}|\alpha_1,\alpha_2) \propto \exp\left(-\frac{\alpha_1}{2}\sum_{w\in\mathcal{W}_1} w^2 - \frac{\alpha_2}{2}\sum_{w\in\mathcal{W}_2} w^2\right)$$

5.5.2 Early stopping

- Early stopping is a way of controlling the effective complexity of a network.
- Our goal is to achieve the best generalization.
- Figure 5.12

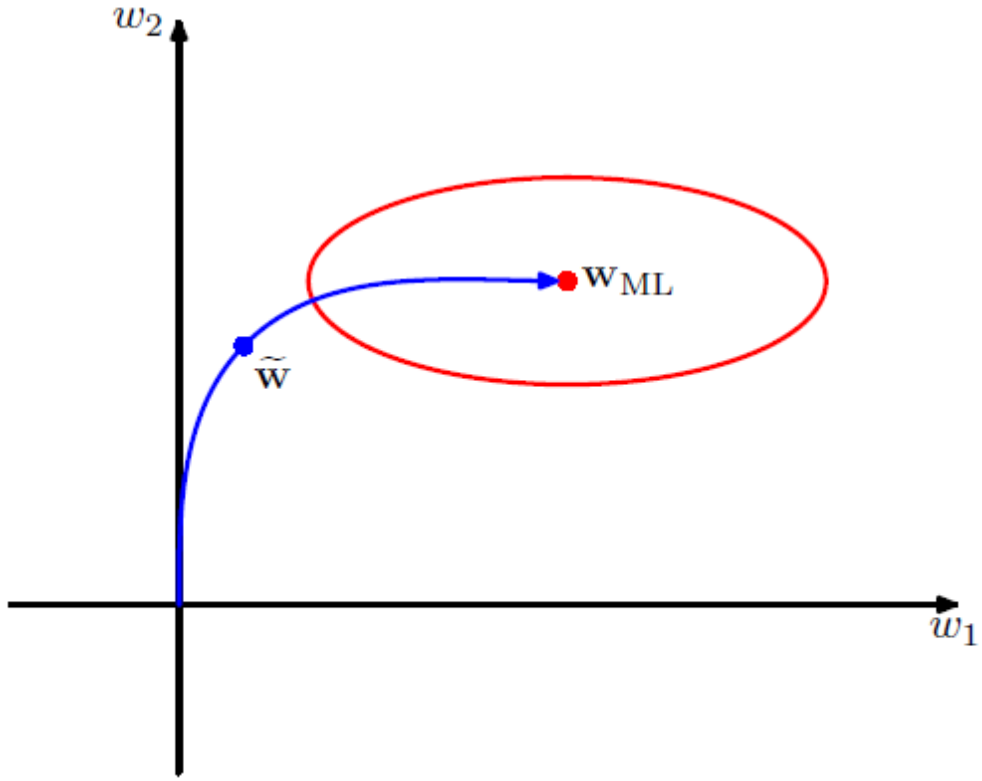


(Training set error)



(Validation set error)

- **Figure 5.13** A schematic illustration of why early stopping can give similar results to weight decay in the case of a quadratic error function.

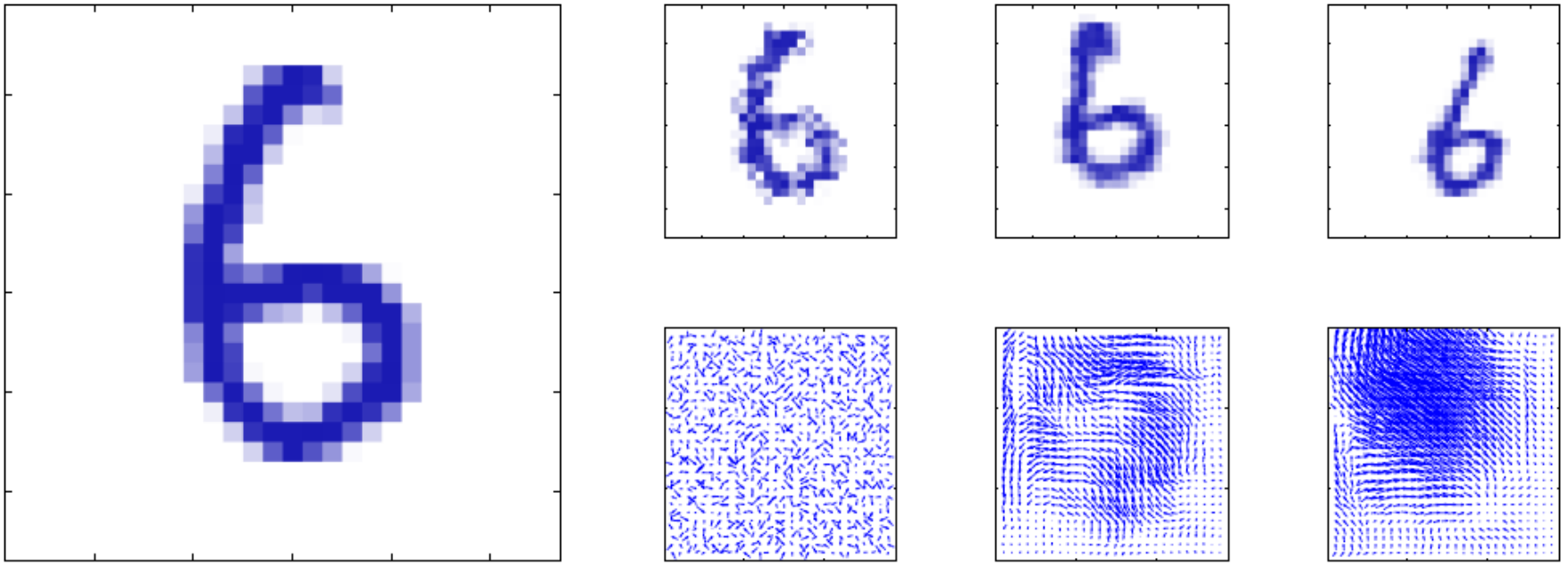


- Early stopping exhibits similar behavior to regularization using a weight-decay term.

5.5.3 Invariances

- The training set is augmented using replicas of the training patterns, transformed according to the desired invariances (e.g., translation or scale invariance).

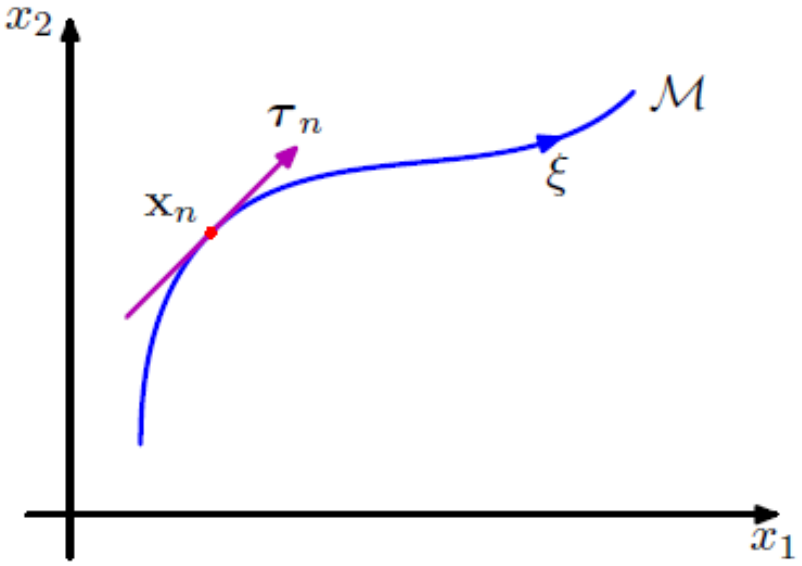
- Figure 5.14



5.5.4 Tangent propagation

- We can use regularization to encourage models to be invariant to transformations of the input through the technique of "*tangent propagation*".

- Figure 5.15



5.5.4 Tangent propagation

- Let the vector that results from acting on $\mathbf{x}_n$ by the transformation $\mathbf{s}(\mathbf{x}_n, \xi)$ with $\mathbf{s}(\mathbf{x}, 0) = \mathbf{x}$ and ξ is a parameter, the tangent vector at the point $\mathbf{x}_n$ is given by

$$\boldsymbol{\tau}_n = \left. \frac{\partial \mathbf{s}(\mathbf{x}_n, \xi)}{\partial \xi} \right|_{\xi=0}$$

- Under a transformation of the input vector, the network output vector will, in general, change.

$$\left. \frac{\partial y_k}{\partial \xi} \right|_{\xi=0} = \left. \sum_{i=1}^{D} \frac{\partial y_k}{\partial x_i} \frac{\partial x_i}{\partial \xi} \right|_{\xi=0} = \sum_{i=1}^{D} J_{ki} \tau_i$$

5.5.4 Tangent propagation

- Encourage local invariance in the neighborhood of the data points by the addition of a regularization function Ω

$$\widetilde{E} = E + \lambda\Omega$$

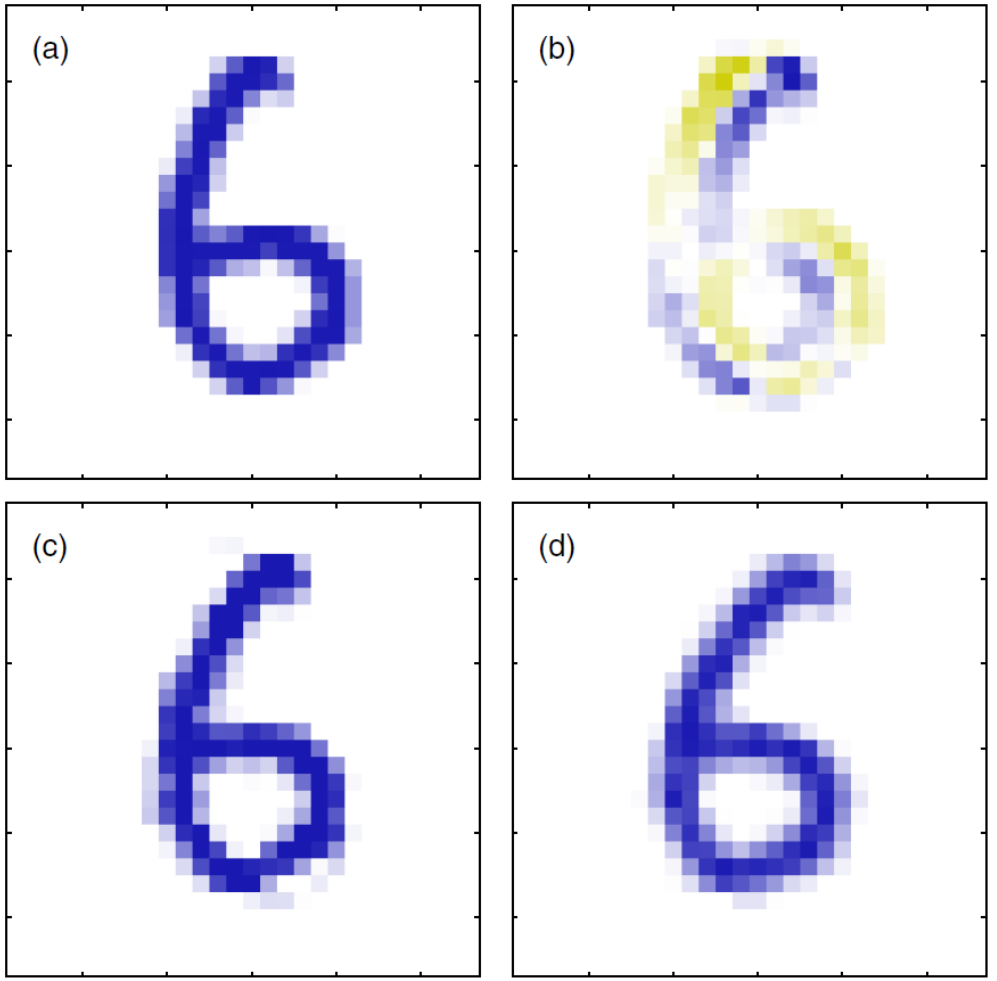
where λ is a regularization coefficient and

$$\Omega = \frac{1}{2}\sum_n\sum_k\left(\left.\frac{\partial y_{nk}}{\partial\xi}\right|_{\xi=0}\right)^2 = \frac{1}{2}\sum_n\sum_k\left(\sum_{i=1}^{D} J_{nki}\tau_{ni}\right)^2$$

- Notes :
  1. $\Omega = 0$ in case of the network mapping function is invariant under the transformation in the neighborhood of each pattern vector.
  2. The value of λ determines the balance between fitting the training data and learning the invariance property.

- Figure 5.16

(a) the original image $\mathbf{x}$ of a hand-written digit, (b) the tangent vector τ corresponding to an infinitesimal clockwise rotation, (c) the result of adding a small contribution from the tangent vector to the original image giving $\mathbf{x} + \epsilon\tau$ with $\epsilon = 15$ degrees, and (d) the true image rotated for comparison.



5.5.5 Training with transformed data

$$\widetilde{E} = E + \lambda\Omega$$

$$E = \frac{1}{2}\iint \{y(\mathbf{x}) - t\}^2 p(t|\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}\,\mathrm{d}t$$

- In case of perturbation by the transformation $\mathbf{s}(\mathbf{x}, \xi)$ with density $p(\xi)$ zero mean and small variance

$$\widetilde{E} = \frac{1}{2}\iiint \{y(\mathbf{s}(\mathbf{x}, \xi)) - t\}^2 p(t|\mathbf{x})p(\mathbf{x})p(\xi)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t\,\mathrm{d}\xi$$

- Expand the transformation by Taylor series

$$\begin{aligned}\mathbf{s}(\mathbf{x}, \xi) &= \mathbf{s}(\mathbf{x}, 0) + \xi \left.\frac{\partial}{\partial\xi}\mathbf{s}(\mathbf{x}, \xi)\right|_{\xi=0} + \frac{\xi^2}{2}\left.\frac{\partial^2}{\partial\xi^2}\mathbf{s}(\mathbf{x}, \xi)\right|_{\xi=0} + O(\xi^3) \\ &= \mathbf{x} + \xi\boldsymbol{\tau} + \frac{1}{2}\xi^2\boldsymbol{\tau}' + O(\xi^3)\end{aligned}$$

5.5.5 Training with transformed data

- Expand the model function to give

$$y(\mathbf{s}(\mathbf{x},\xi)) = y(\mathbf{x}) + \xi\boldsymbol{\tau}^{\mathrm{T}}\nabla y(\mathbf{x}) + \frac{\xi^2}{2}\left[(\boldsymbol{\tau}')^{\mathrm{T}}\nabla y(\mathbf{x}) + \boldsymbol{\tau}^{\mathrm{T}}\nabla\nabla y(\mathbf{x})\boldsymbol{\tau}\right] + O(\xi^3)$$

- Taylor series in terms of $y(\mathbf{s}(\mathbf{x},\xi))$

$$\begin{aligned}
\widetilde{E} &= \frac{1}{2}\iint \{y(\mathbf{x}) - t\}^2 p(t|\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}\,\mathrm{d}t \\
&+ \mathbb{E}[\xi]\iint \{y(\mathbf{x}) - t\}\boldsymbol{\tau}^{\mathrm{T}}\nabla y(\mathbf{x})p(t|\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}\,\mathrm{d}t \\
&+ \mathbb{E}[\xi^2]\iint \left[\{y(\mathbf{x}) - t\}\frac{1}{2}\left\{(\boldsymbol{\tau}')^{\mathrm{T}}\nabla y(\mathbf{x}) + \boldsymbol{\tau}^{\mathrm{T}}\nabla\nabla y(\mathbf{x})\boldsymbol{\tau}\right\}\right. \\
&\quad \left. + \left(\boldsymbol{\tau}^{\mathrm{T}}\nabla y(\mathbf{x})\right)^2\right] p(t|\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}\,\mathrm{d}t + O(\xi^3).
\end{aligned}$$

5.5.5 Training with transformed data

- Omitting terms of $O(\xi^3)$, the average error function is

$$\widetilde{E} = E + \lambda\Omega$$

where the regularization term takes the form

$$\begin{aligned}\Omega = \int \Big[\{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\} \frac{1}{2} \Big\{ (\boldsymbol{\tau}')^{\mathrm{T}} \nabla y(\mathbf{x}) + \boldsymbol{\tau}^{\mathrm{T}} \nabla\nabla y(\mathbf{x}) \boldsymbol{\tau} \Big\} \\ + \big(\boldsymbol{\tau}^T \nabla y(\mathbf{x})\big)^2 \Big] p(\mathbf{x})\, \mathrm{d}\mathbf{x}\end{aligned}$$

in which we have performed the integration over t.

5.5.5 Training with transformed data

- To simplify the regularization term, we express

$$y(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}] + O(\xi^2)$$

which minimizes the total error.

- Thus, to order ξ^2, the first term in the regularizer vanishes and left with

$$\Omega = \frac{1}{2}\int \left(\boldsymbol{\tau}^T \nabla y(\mathbf{x})\right)^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x}$$

- In case of $\mathbf{x} \to \mathbf{x} + \boldsymbol{\xi}$, then $\frac{\partial S}{\partial \xi} = 1 = \tau$ and

$$\Omega = \frac{1}{2}\int \|\nabla y(\mathbf{x})\|^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x}$$

which is known as *Tikhonov* regularization. We can build MLP by considering this regularization function.

5.5.7 Soft weight sharing

- One way to reduce the effective complexity is to constrain weights within certain groups to be equal.
- We can encourage the weight values to form several groups, rather than just one group, by considering a mixture of Gaussians

$$p(\mathbf{w}) = \prod_i p(w_i)$$

where

$$p(w_i) = \sum_{j=1}^{M} \pi_j \mathcal{N}(w_i | \mu_j, \sigma_j^2)$$

and π_j are the mixing coefficients.

5.5.7 Soft weight sharing

- Taking the negative logarithm then leads to a regularization term

$$\Omega(\mathbf{w}) = -\sum_i \ln\left(\sum_{j=1}^{M} \pi_j \mathcal{N}(w_i|\mu_j, \sigma_j^2)\right)$$

and the total error is given by

$$\widetilde{E}(\mathbf{w}) = E(\mathbf{w}) + \lambda\Omega(\mathbf{w})$$

- Parameters $\{\pi_j, \mu_j, \sigma_j\}$ can be determined by using EM algorithm if the weights were constant.
- As the distribution of weights is itself evolving during the learning process, a joint optimization can be performed simultaneously over the weights and the mixture-model parameters.