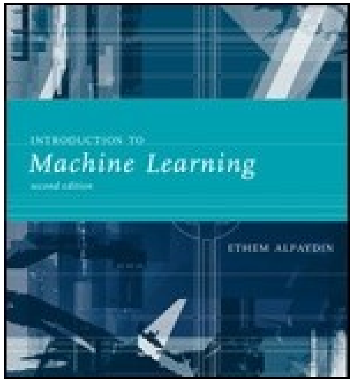
Lecture Slides for

INTRODUCTION TO

Machine Learning

2nd Edition



ETHEM ALPAYDIN, modified by Leonardo Bobadilla
and some parts from
http://www.cs.tau.ac.il/~apartzin/MachineLearning/

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2m

Outline

Last Class: Ch 4: Parametric Methods

The Bayes Estimator

Parametric Classification

Regression

Tuning Model Complexity

This class: Ch 5: Multivariate Methods

- Multivariate Data
- Parameter Estimation
- Estimation of Missing Values
- Multivariate Classification

CHAPTER 4:

Parametric Methods

Regression

$$r = f(x) + \epsilon$$

- x is independent variable, r is dependant variable
- Unknown f, want to approximate to predict future values
- Parametric approach: assume model with small number of parameters $g(x|\theta)$
- Find best parameters from data
- Also have to make assumption on noise

Regressions

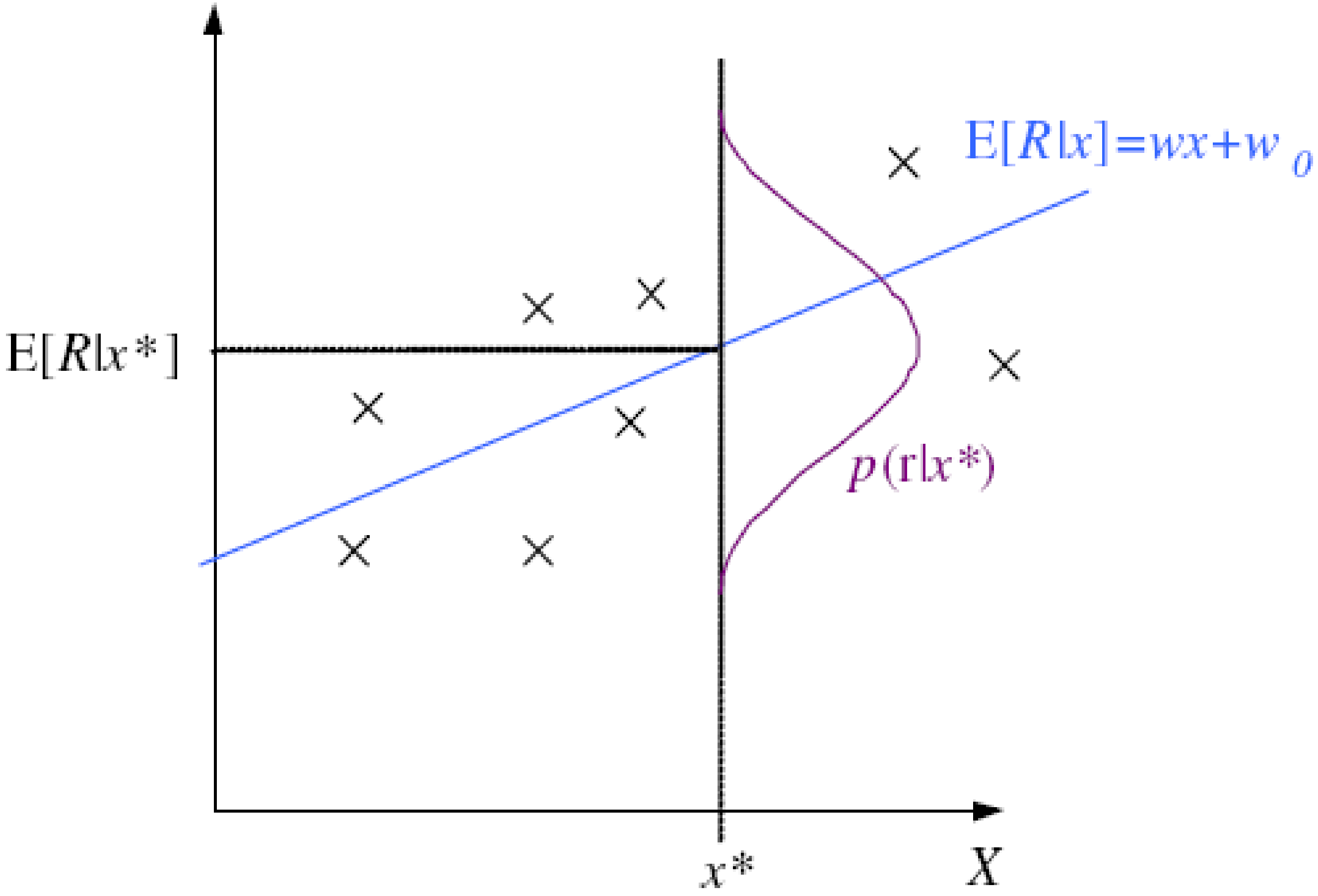
$$\epsilon \sim \mathcal{N}(0, \sigma^2) \qquad r = f(x) + \epsilon$$

$$p(r|x) \sim \mathcal{N}(g(x|\theta), \sigma^2)$$

- Have a training data (x,r)
- Find parameters to maximize likelihood
- In other words, what parameters makes data most probable

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Regressions



Regressions

$$p(x, r) = p(r|x)p(x)$$

$$\begin{aligned}\mathcal{L}(\theta|\mathcal{X}) &= \log \prod_{t=1}^{N} p(x^t, r^t) \\ &= \log \prod_{t=1}^{N} p(r^t|x^t) + \log \prod_{t=1}^{N} p(x^t)\end{aligned}$$

- Ignore the last term,(does not depend on parameters

Regression

$$\begin{aligned}
\mathcal{L}(\theta|\mathcal{X}) &= \log \prod_{t=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{[r^t - g(x^t|\theta)]^2}{2\sigma^2}\right] \\
&= \log \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^N \exp\left[-\frac{1}{2\sigma^2}\sum_{t=1}^{N}[r^t - g(x^t|\theta)]^2\right] \\
&= -N\log(\sqrt{2\pi}\sigma) - \frac{1}{2\sigma^2}\sum_{t=1}^{N}[r^t - g(x^t|\theta)]^2
\end{aligned}$$

- Minimize last term

Least Square Estimate

$$E(\theta|\mathcal{X}) = \frac{1}{2}\sum_{t=1}^{N}[r^t - g(x^t|\theta)]^2$$

- Minimize this

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Linear Regression

- Assume linear model
- Need to minimize
- Set derivatives to zero
- 2 linear equations in 2 unknowns
- Can solve easily

$$g\left(x^t|w_1,w_0\right)=w_1x^t+w_0$$

$$E(\theta|X)=\frac{1}{2}\sum_{t=1}^{N}[r^t-g(x^t|\theta)]^2$$

$$\sum_t r^t=Nw_0+w_1\sum_t x^t$$

$$\sum_t r^tx^t=w_0\sum_t x^t+w_1\sum_t\left(x^t\right)^2$$

Linear Regression

$$\mathbf{A} = \begin{bmatrix} N & \sum_t x^t \\ \sum_t x^t & \sum_t (x^t)^2 \end{bmatrix}, \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} \sum_t r^t \\ \sum_t r^t x^t \end{bmatrix}$$

and can be solved as $\boldsymbol{w} = \mathbf{A}^{-1}\boldsymbol{y}$.

Polynomial Regression

$$g\left(x^t \mid w_k,\ldots,w_2,w_1,w_0\right)=w_k\left(x^t\right)^k+\cdots+w_2\left(x^t\right)^2+w_1x^t+w_0$$

$$\mathbf{A}\boldsymbol{w}=\boldsymbol{y}$$

$$\mathbf{A} = \begin{bmatrix} N & \sum_t x^t & \sum_t (x^t)^2 & \cdots & \sum_t (x^t)^k \\ \sum_t x^t & \sum_t (x^t)^2 & \sum_t (x^t)^3 & \cdots & \sum_t (x^t)^{k+1} \\ \vdots & & & & \\ \sum_t (x^t)^k & \sum_t (x^t)^{k+1} & \sum_t (x^t)^{k+2} & \cdots & \sum_t (x^t)^{2k} \end{bmatrix}$$

$$\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_k \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} \sum_t r^t \\ \sum_t r^t x^t \\ \sum_t r^t (x^t)^2 \\ \vdots \\ \sum_t r^t (x^t)^k \end{bmatrix}$$

Polynomial Regression

$$g\left(x^t \mid w_k, \ldots, w_2, w_1, w_0\right) = w_k \left(x^t\right)^k + \cdots + w_2 \left(x^t\right)^2 + w_1 x^t + w_0$$

$$\boldsymbol{w} = \left(\mathbf{D}^T \mathbf{D}\right)^{-1} \mathbf{D}^T \boldsymbol{r}$$

Polynomial Regression

$$g\left(x^t \mid w_k,\ldots,w_2,w_1,w_0\right) = w_k\left(x^t\right)^k + \cdots + w_2\left(x^t\right)^2 + w_1 x^t + w_0$$

$$\mathbf{D} = \begin{bmatrix} 1 & x^1 & \left(x^1\right)^2 & \cdots & \left(x^1\right)^k \\ 1 & x^2 & \left(x^2\right)^2 & \cdots & \left(x^2\right)^k \\ \vdots & & & & \\ 1 & x^N & \left(x^N\right)^2 & \cdots & \left(x^N\right)^2 \end{bmatrix} \quad \boldsymbol{r} = \begin{bmatrix} r^1 \\ r^2 \\ \vdots \\ r^N \end{bmatrix}$$

$$\boldsymbol{w} = \left(\mathbf{D}^T\mathbf{D}\right)^{-1}\mathbf{D}^T\boldsymbol{r}$$

Tuning Model Complexity: Bias and Variance

- Given single sample (x,r), what is the expected error
- Variations are due to noise and training

$$E[(r-g(x))^2|x] = \underbrace{E[(r-E[r|x])^2|x]}_{noise} + \underbrace{(E[r|x]-g(x))^2}_{squared\ error}$$

- First term is due to noise
  - Does not depend on the estimate
  - Can’t be removed

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Variance

$$E[(r-g(x))^2|x] = \underbrace{E[(r-E[r|x])^2|x]}_{noise} + \underbrace{(E[r|x]-g(x))^2}_{squared\ error}$$

- Second term
  - Deviation of estimator from regression function
  - Depends on estimator and training set
  - Average over all possible training samples

$$E_{\mathcal{X}}[(E[r|x]-g(x))^2|x] = \underbrace{(E[r|x]-E_{\mathcal{X}}[g(x)])^2}_{bias} + \underbrace{E_{\mathcal{X}}[(g(x)-E_{\mathcal{X}}[g(x)])^2]}_{variance}$$

Bias and Variance

$$E\left[(r - g(x))^2 \mid x\right] = E\left[(r - E[r \mid x])^2 \mid x\right] + \left(E[r \mid x] - g(x)\right)^2$$

noise *squared error*

$$E_X\left[(E[r \mid x] - g(x))^2 \mid x\right] = \left(E[r \mid x] - E_X[g(x)]\right)^2 + E_X\left[(g(x) - E_X[g(x)])^2\right]$$

bias *variance*

Bias/Variance Dilemma

- Example: $g_i(x)$=2 has no variance and high bias

$g_i(x)= \sum_t r^t_i/N$ has lower bias with variance

- As we increase complexity,

bias decreases (a better fit to data) and

variance increases (fit varies more with data)

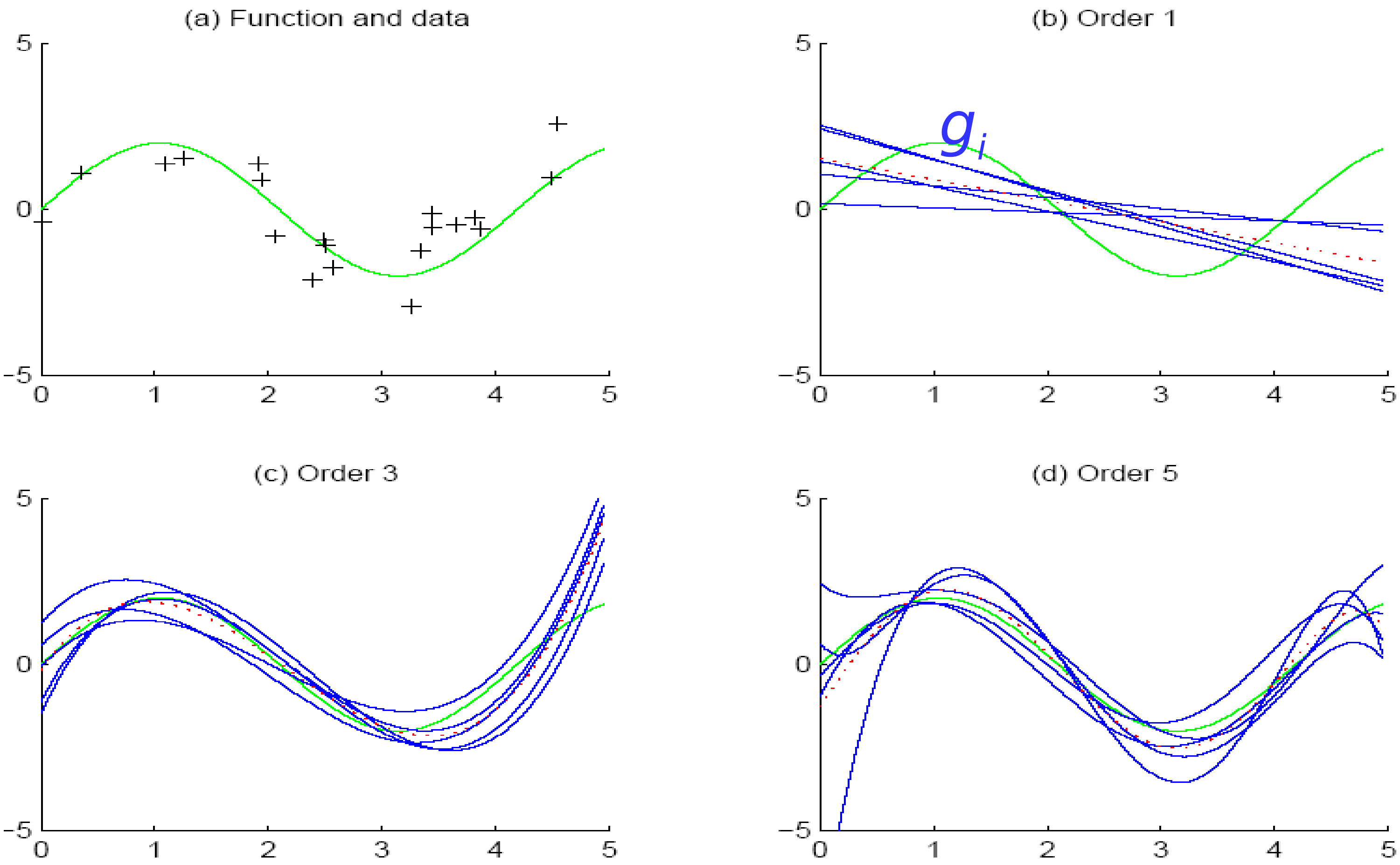
- Bias/Variance dilemma: (Geman et al., 1992)

Example: polynomial regression

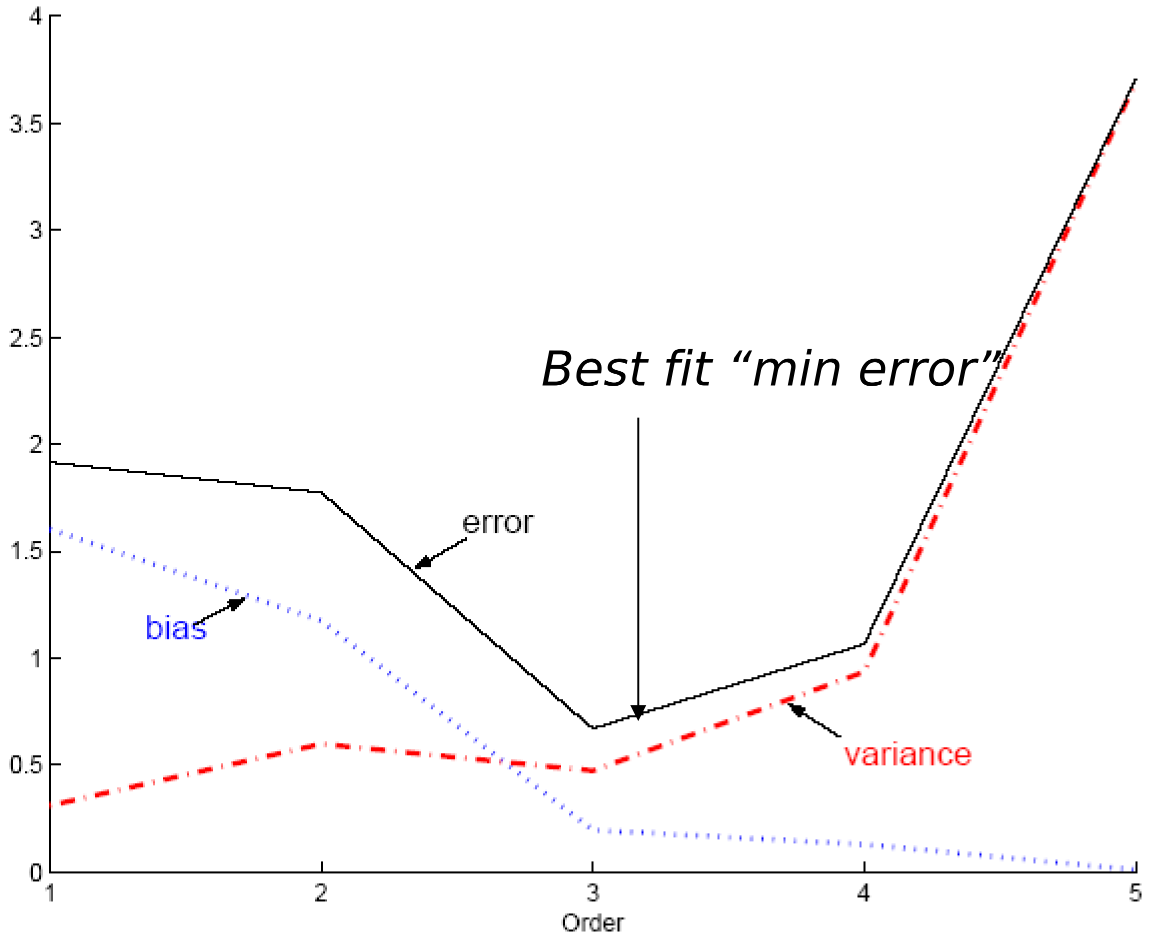
- As we increase degree of the polynomial
  - Bias decreases as allow better fit to points
  - Variance increases as small deviation in training sample might result in large deviation in model parameters
- Bias/variance dilemma true for any machine learning systems
- Need a way to find optimal model complexity to balance between bias and variance

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Bias/Variance Dilemma



Polynomial Regression



Best fit “min error”

Model Selection

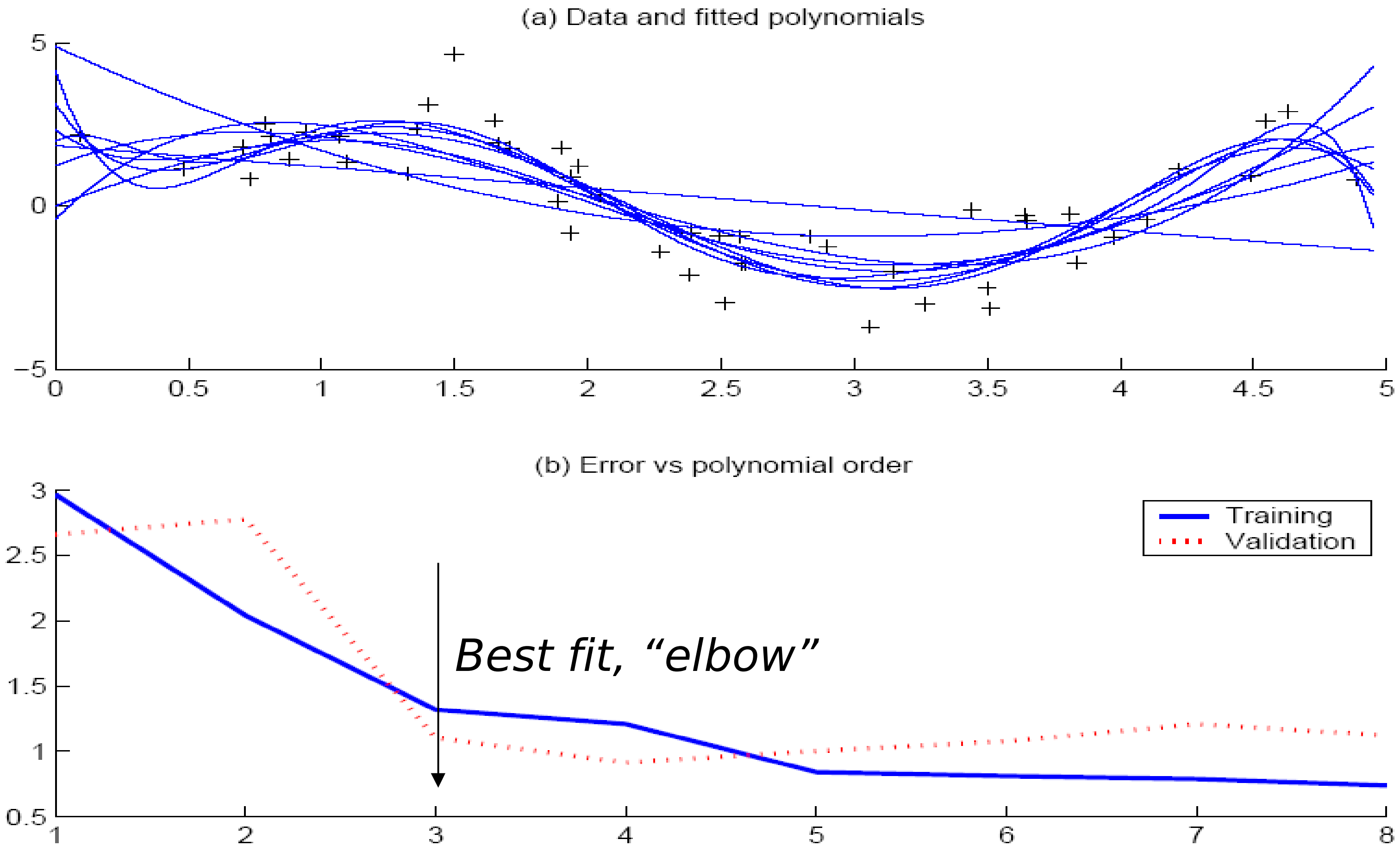
- How to select right model complexity?
- Different from estimating model parameters
- There are several procedures

Cross-Validation

- Can’t calculate bias and variance as don’t know true model
- But can estimate total generalization error
- Set aside portion of data (validation set)
- Increase model complexity, find parameters
- Calculate error on validation set
- Stop when error cease to decrease or even start increasing

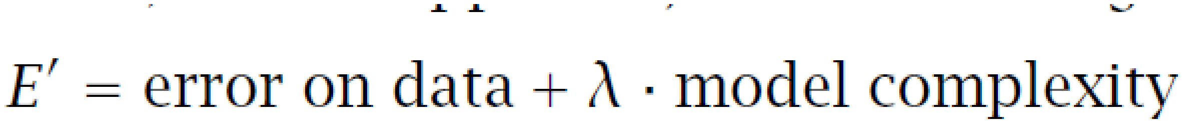
Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Cross-Validation



Best fit, "elbow"

Regularization

- Introduce penalty for model complexity into an error function
- 

  $E' = \text{error on data} + \lambda \cdot \text{model complexity}$
- Find optimal model complexity (e.g. degree of polynomial) and optimal parameters (coefficients) which minimize this function
- Lambda is penalty for model complexity
- If lambda is too large only very simple models will be admitted

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

CHAPTER 5:

Multivariate Methods

Motivating Example

- Loan Application
- Observation Vector: Information About Customer
  - Age
  - Marital Status
  - Yearly Income
  - Savings
- Inputs/Attribute/Features associated with a customer
- The variables are correlated (savings vs. age)

Correlation

- Suppose we have two random variables X and Y.
- We want to estimate the degree of “correlation” among them
  - Positive Correlation: If one happens to be large so the probability that another one will be large is significant
  - Negative Correlation: If one happens to be large so the probability that another one will be small is significant
  - Zero correlation: Value of one tells nothing about the value of other

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Correlation

- Some reasonable assumptions
  - The “correlation” between X and Y is the same as between X+a and X+b where a,b constant
  - The “correlation” between X and Y is the same as between aX and bY
  - a,b are constant
- Example
  - If there is a connection between temperature inside the building and outside the building , it’s does not mater what scale is used

Correlation

- Let’s do a “normalization”

$$X_1 = \frac{X - EX}{\sigma_X}, Y_1 = \frac{Y - EY}{\sigma_Y}$$

- Both these variables have zero mean and unit variance
- Filtered out the individual differences
- Let’s check mean (expected) square differences between them $E(X_1 - Y_1)^2$

Correlation

$$E(X_1 - Y_1)^2$$

- The result should be
  - Small when positively “correlated”
  - Large when negatively correlated
  - Medium when “uncorrelated”

Correlation

$$E(X_1 - Y_1)^2 = E(X_1^{\,2} + Y_1^2 - 2X_1Y_1) =$$

$$= EX_1^{\,2} + EY_1^2 - 2EX_1Y_1 = 2 - 2\rho$$

$$\rho = EX_1Y_1 = \frac{E(X - EY\ )(X\ \ - EY)}{\sigma_1\sigma_2} = \frac{Cov(X,Y)}{\sigma_1\sigma_2}$$

- Larger covariance means larger correlation coefficient means smaller average square differences

Correlation vs. Dependance

- Not the same thing
- Independent=>Have zero correlation
- Have zero correlation=> May not be independent
- We look at square differences between two variables

$$E(X_1 - Y_1)^2$$

- Two variables might have “unpredictable” square differences but still be dependant

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Correlation vs. Independence

- Random variable X from {-1,0,1} with p=1/3
- Random variable Y=X^2
- Clearly dependant but
- COV(X,Y)=E((X-0)(Y-EY))=EXY-EY*EX=EXY=EX^3=0
- Correlation only measures “linear” independence

Multivariate Distribution

- Assume all members of class came from join distribution
- Can learn distributions from data P(x|C)
- Assign new instance for most probable class P(C|x) using Bayes rule
- An instance described by a vector of correlated parameters
- Realm of multivariate distributions
- Multivariate normal

Multivariate Data

- Multiple measurements (sensors)
- *d* inputs/features/attributes: *d*-variate
- *N* instances/observations/examples

$$\mathbf{X} = \begin{bmatrix} X_1^1 & X_2^1 & \cdots & X_d^1 \\ X_1^2 & X_2^2 & \cdots & X_d^2 \\ \vdots & & & \\ X_1^N & X_2^N & \cdots & X_d^N \end{bmatrix}$$

Multivariate Parameters

$$\text{Mean}: E[\boldsymbol{x}] = \boldsymbol{\mu} = [\mu_1, \ldots, \mu_d]^T$$

$$\text{Covariance}: \sigma_{ij} \equiv \text{Cov}(X_i, X_j)$$

$$\text{Correlation}: \text{Corr}(X_i, X_j) \equiv \rho_{ij} = \frac{\sigma_{ij}}{\sigma_i \sigma_j}$$

$$\Sigma \equiv \text{Cov}(\boldsymbol{X}) = E\left[(\boldsymbol{X} - \mu)(\boldsymbol{X} - \mu)^T\right] = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1d} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2d} \\ \vdots & & & \\ \sigma_{d1} & \sigma_{d2} & \cdots & \sigma_d^2 \end{bmatrix}$$

Parameter Estimation

Samplemean $\mathbf{m}: m_i = \frac{\sum_{t=1}^{N} x_i^t}{N}, i = 1, \ldots, d$

Covariancematrix $\mathbf{S}: s_{ij} = \frac{\sum_{t=1}^{N} \left(x_i^t - m_i\right)\left(x_j^t - m_j\right)}{N}$

Correlation matrix $\mathbf{R}: r_{ij} = \frac{s_{ij}}{s_i s_j}$

Estimation of Missing Values

- What to do if certain instances have missing attributes?
- Ignore those instances: not a good idea if the sample is small
- Use ‘missing’ as an attribute: may give information
- **Imputation**: Fill in the missing value
  - Mean imputation: Use the most likely value (e.g., mean)
  - Imputation by regression: Predict based on other attributes

Multivariate Normal

- Have d-attributes
- Often can assume each one distributed normally
- Attributes might be dependant/correlated
- Joint distribution of correlated several variables
  - $P(X_1=x_1, X_2=x_2, \ldots X_d=x_d)=?$
  - X_1 is normally distributed with mean μ_i and variance σ_i

Multivariate Normal

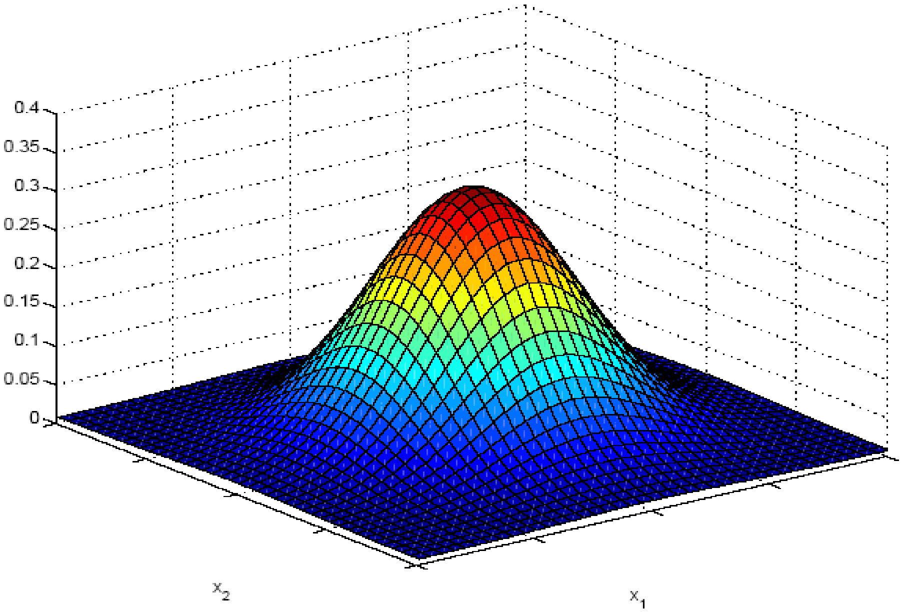
$$\boldsymbol{x} \sim \mathrm{N}_d(\boldsymbol{\mu}, \Sigma)$$

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu})\right]$$

- Mahalanobis distance: $(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\sum}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})$
- 2 variables are correlated
- Divided by inverse of covariance (large)
- Contribute less to Mahalanobis distance
- Contribute more to the probability

Bivariate Normal

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$



$$p(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left[-\frac{1}{2(1-\rho^2)}\left(z_1^2 - 2\rho z_1 z_2 + z_2^2\right)\right]$$

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Multivariate Normal Distribution

- Mahalanobis distance: $(\boldsymbol{x} - \boldsymbol{\mu})^T \sum^{-1} (\boldsymbol{x} - \boldsymbol{\mu})$

  measures the distance from $\boldsymbol{x}$ to $\boldsymbol{\mu}$ in terms of $\sum$ (normalizes for difference in variances and correlations)

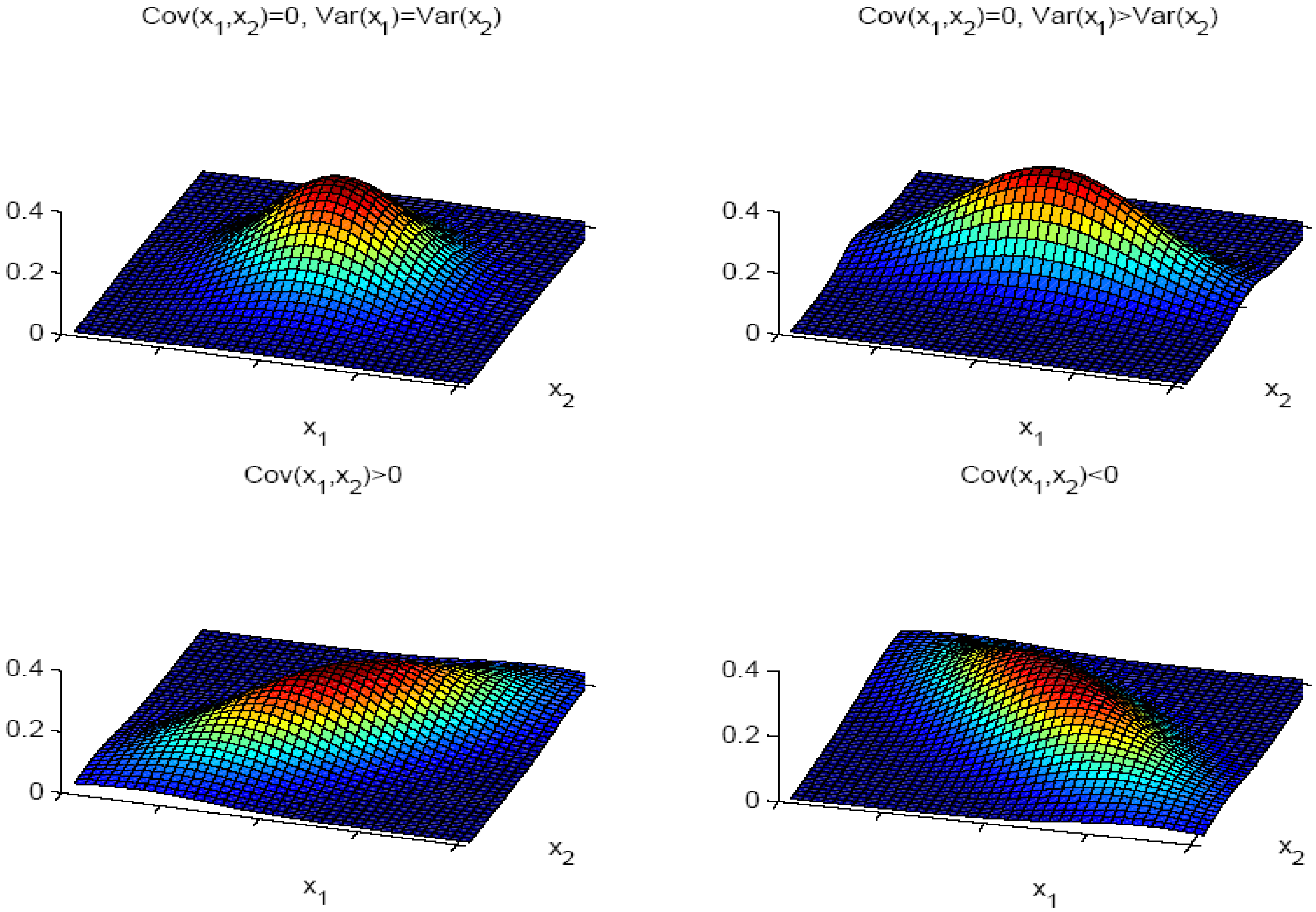
- Bivariate: $d = 2$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$

$$p(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left[-\frac{1}{2(1-\rho^2)} \left(z_1^2 - 2\rho z_1 z_2 + z_2^2 \right) \right]$$

$$z_i = (x_i - \mu_i) / \sigma_i$$

Bivariate Normal



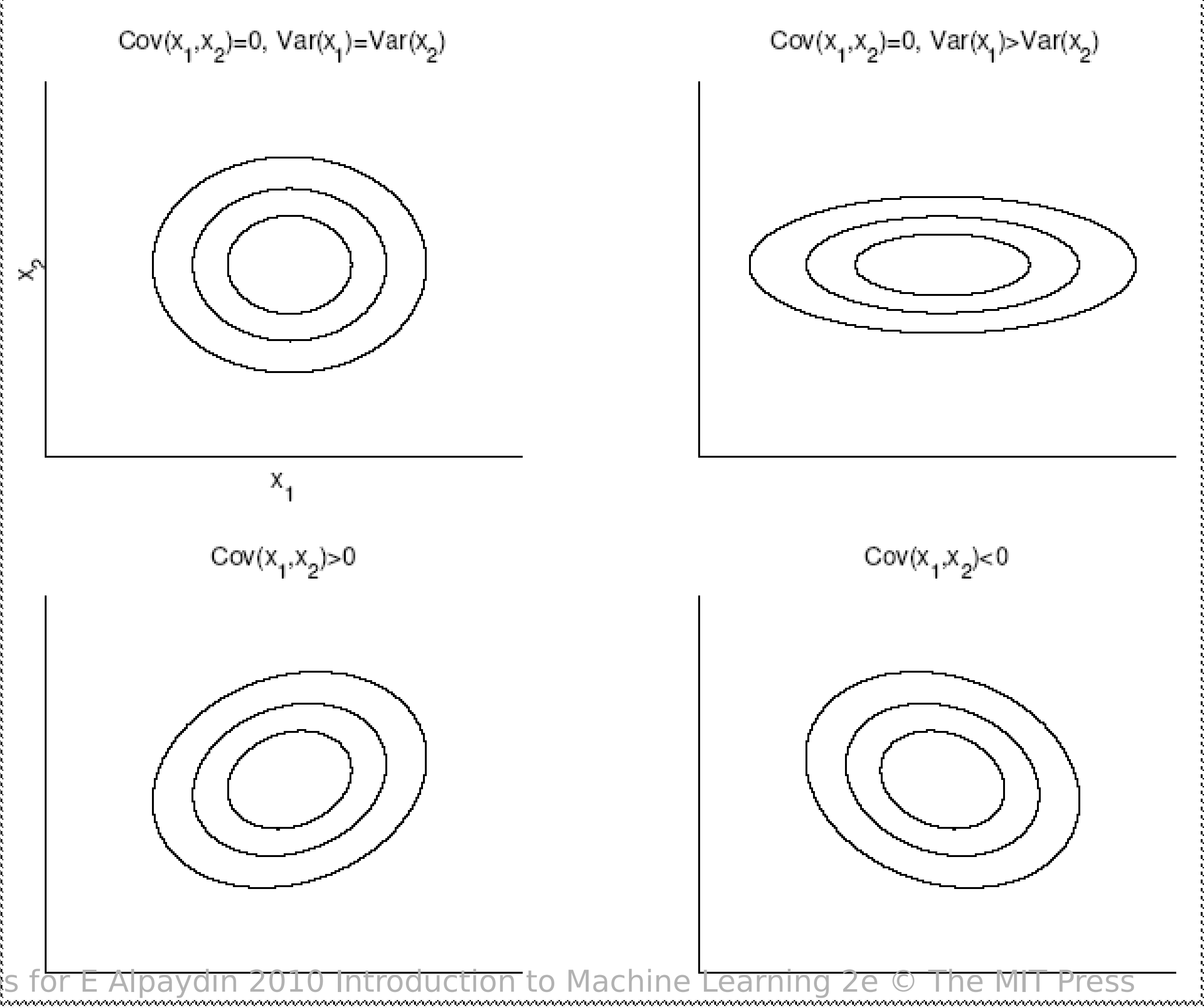
Cov(x_1,x_2)=0, Var(x_1)=Var(x_2)

Cov(x_1,x_2)=0, Var(x_1)>Var(x_2)

Cov(x_1,x_2)>0

Cov(x_1,x_2)<0

Bivariate Normal



Independent Inputs: Naive Bayes

- If x_i are independent, offdiagonals of ∑ are 0, Mahalanobis distance reduces to weighted (by $1/\sigma_i$) Euclidean distance:

$$p(x) = \prod_{i=1}^{d} p_i(x_i) = \frac{1}{(2\pi)^{d/2} \prod_{i=1}^{d} \sigma_i} \exp\left[-\frac{1}{2} \sum_{i=1}^{d} \left(\frac{x_i - \mu_i}{\sigma_i} \right)^2 \right]$$

- If variances are also equal, reduces to Euclidean distance

Based on Introduction to Machine Learning © The MIT Press (V1.1)

Projection Distribution

- Example: vector of 3 features
- Multivariate normal distribution
- Projection to 2 dimensional space (e.g. XY plane) Vectors of 2 features
- Projection are also multivariate normal distribution
- Projection of d-dimensional normal to k-dimensional space is k-dimensional normal

$\mathrm{W}^T \boldsymbol{x} \sim \mathcal{N}_k(\mathrm{W}^T \boldsymbol{\mu}, \mathrm{W}^T \Sigma \mathrm{W})$ W is a $d \times k$ matrix

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

1D projection

$$\boldsymbol{w}^T\boldsymbol{x} = w_1x_1 + w_2x_2 + \cdots + w_dx_d \sim \mathcal{N}(\boldsymbol{w}^T\boldsymbol{\mu}, \boldsymbol{w}^T\Sigma\boldsymbol{w})$$

$$\begin{aligned}
E[\boldsymbol{w}^T\boldsymbol{x}] &= \boldsymbol{w}^T E[\boldsymbol{x}] = \boldsymbol{w}^T\boldsymbol{\mu} \\
\mathrm{Var}(\boldsymbol{w}^T\boldsymbol{x}) &= E[(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})^2] = E[(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})] \\
&= E[\boldsymbol{w}^T(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T\boldsymbol{w}] = \boldsymbol{w}^T E[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T]\boldsymbol{w} \\
&= \boldsymbol{w}^T\Sigma\boldsymbol{w}
\end{aligned}$$

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Multivariate Classification

- Assume members of class from a single multivariate distribution
- Multivariate normal is a good choice
  - Easy to analyze
  - Model many natural phenomena
  - Model a class as having single prototype source (mean) slightly randomly changed

Example

- Matching cars to customers
- Each cat defines a class of matching customers
- Customers described by (age, income)
- There is a correlation between age and income
- Assume each class is multivariate normal
- Need to learn P(x|C) from data
- Use Bayes to compute P(C|x)

Parametric Classification

- If $p\,(\boldsymbol{x} \mid C_i) \sim \mathrm{N}\,(\boldsymbol{\mu}_i , \sum_i)$

$$p(\boldsymbol{x}\,|C_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right]$$

- Discriminant functions are

$$g_i(\mathbf{x}) = \log P(C_i \mid x) = \log \frac{P(\mathbf{x}|C_i)\,P(C_i)}{P(x)} = \log p(\mathbf{x}|C_i) + \log P(C_i) - \log P(x)$$

$$= -\frac{d}{2}\log 2\pi - \frac{1}{2}\log|\Sigma_i| - \frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1}(\mathbf{x}-\mu_i) + \log P(C_i) - Log P(x)$$

- Need to know Covariance Matrix and mean to compute discriminant functions.
- Can ignore P(x) as the same for all classes

Estimation of Parameters

$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N}$$

$$\boldsymbol{m}_i = \frac{\sum_t r_i^t \boldsymbol{x}^t}{\sum_t r_i^t}$$

$$\mathbf{S}_i = \frac{\sum_t r_i^t (\boldsymbol{x}^t - \boldsymbol{m}_i)(\boldsymbol{x}^t - \boldsymbol{m}_i)^T}{\sum_t r_i^t}$$

$$g_i(\boldsymbol{x}) = -\frac{1}{2}\log|\mathbf{S}_i| - \frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_i)^T \mathbf{S}_i^{-1}(\boldsymbol{x} - \boldsymbol{m}_i) + \log \hat{P}(C_i)$$

Covariance Matrix per Class

- Quadratic discriminant

$$g_i(\boldsymbol{x}) = -\frac{1}{2}\log|\mathbf{S}_i| - \frac{1}{2}\left(\boldsymbol{x}^T\mathbf{S}_i^{-1}\boldsymbol{x} - 2\boldsymbol{x}^T\mathbf{S}_i^{-1}\boldsymbol{m}_i + \boldsymbol{m}_i^T\mathbf{S}_i^{-1}\boldsymbol{m}_i\right) + \log\hat{P}(C_i)$$

$$= \boldsymbol{x}^T\mathbf{W}_i\boldsymbol{x} + \boldsymbol{w}_i^T\boldsymbol{x} + w_{i0}$$

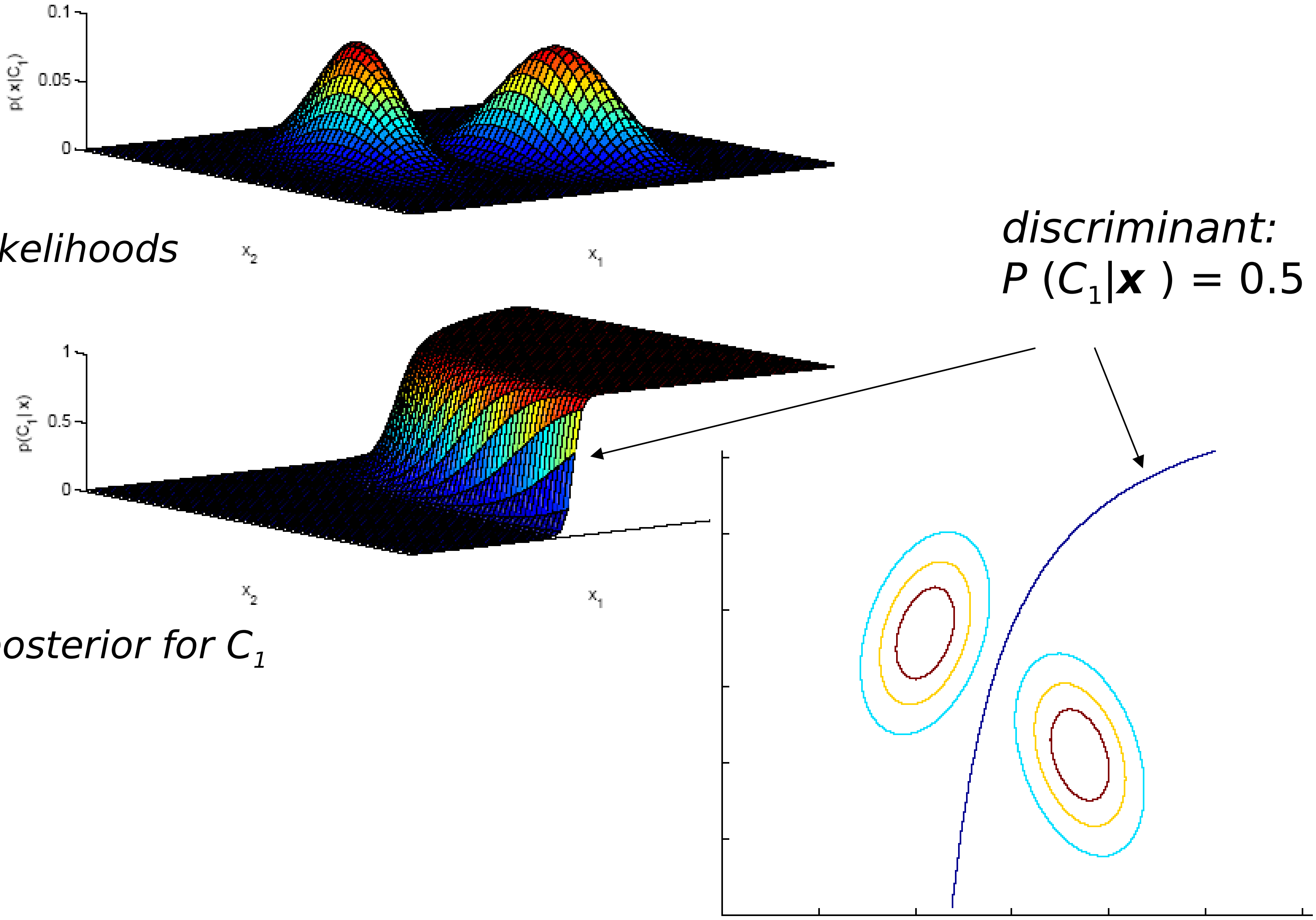
where

$$\mathbf{W}_i = -\frac{1}{2}\mathbf{S}_i^{-1}$$

$$\boldsymbol{w}_i = \mathbf{S}_i^{-1}\boldsymbol{m}_i$$

$$w_{i0} = -\frac{1}{2}\boldsymbol{m}_i^T\mathbf{S}_i^{-1}\boldsymbol{m}_i - \frac{1}{2}\log|\mathbf{S}_i| + \log\hat{P}(C_i)$$

- Requires estimation of K*d*(d+1)/2 parameters for covariance matrix



likelihoods

posterior for C_1

discriminant:
$P\ (C_1|\boldsymbol{x}\ ) = 0.5$

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Common Covariance Matrix **S**

- If not enough data can assume all classes have same common sample covariance matrix **S**

  $$\mathbf{S} = \sum_i \hat{P}(C_i)\mathbf{S}_i$$

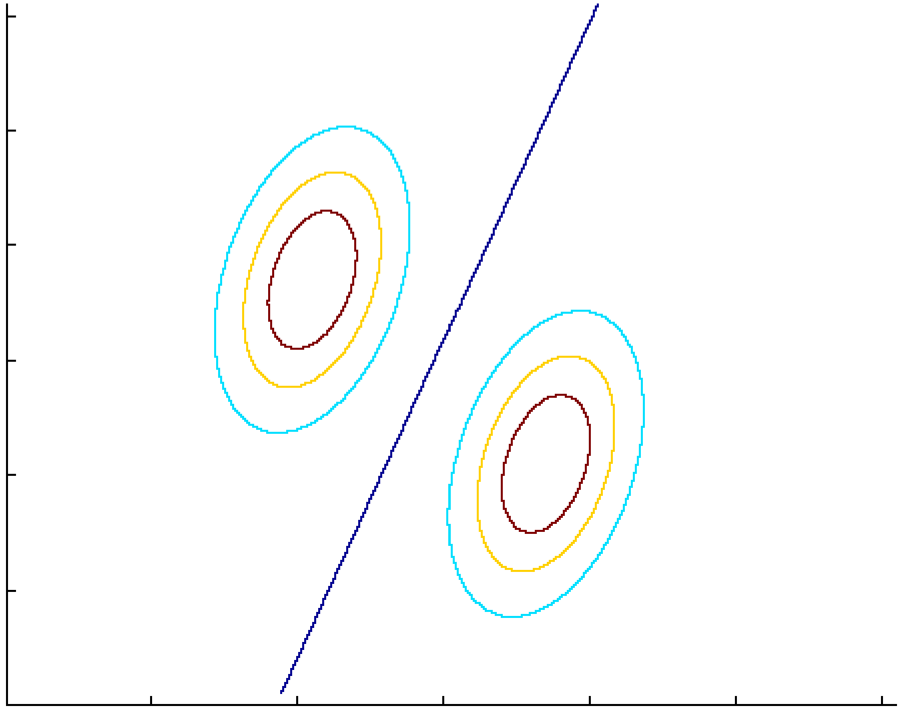
  Discriminant reduces to a linear discriminant ($x^T S^{-1} x$ is common to all discriminant and can be removed)

$$g_i(\boldsymbol{x}) = -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_i)^T \mathbf{S}^{-1}(\boldsymbol{x} - \boldsymbol{m}_i) + \log \hat{P}(C_i)$$

$$g_i(\mathbf{x}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

$$\text{where } \mathbf{w}_i = \mathbf{S}^{-1}\mathbf{m}_i \quad w_{i0} = -\frac{1}{2}\mathbf{m}_i^T \mathbf{S}^{-1}\mathbf{m}_i + \log \hat{P}(C_i)$$

Common Covariance Matrix **S**



Diagonal **S**

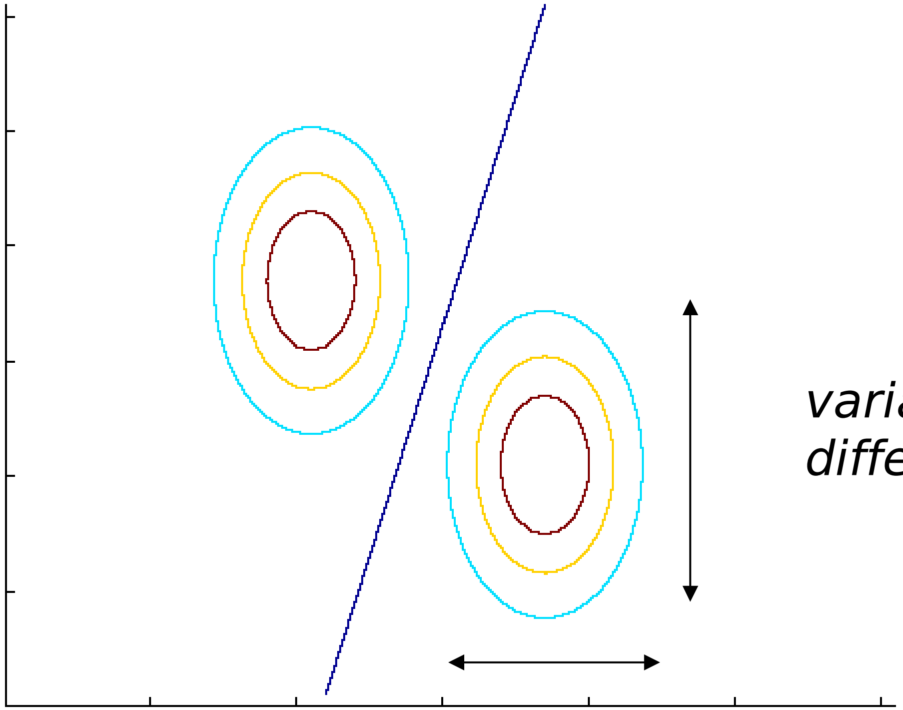
- When x_j $j = 1,..d$, are independent, ∑ is diagonal

$p(\mathbf{x}|C_i) = \prod_j p(x_j|C_i)$ (Naive Bayes' assumption)

$$g_i(\mathbf{x}) = -\frac{1}{2}\sum_{j=1}^{d}\left(\frac{x_j^t - m_{ij}}{s_j}\right)^2 + \log \hat{P}(C_i)$$

Classify based on weighted Euclidean distance (in s_j units) to the nearest mean

Diagonal **S**



variances may be different

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Diagonal **S**, equal variances

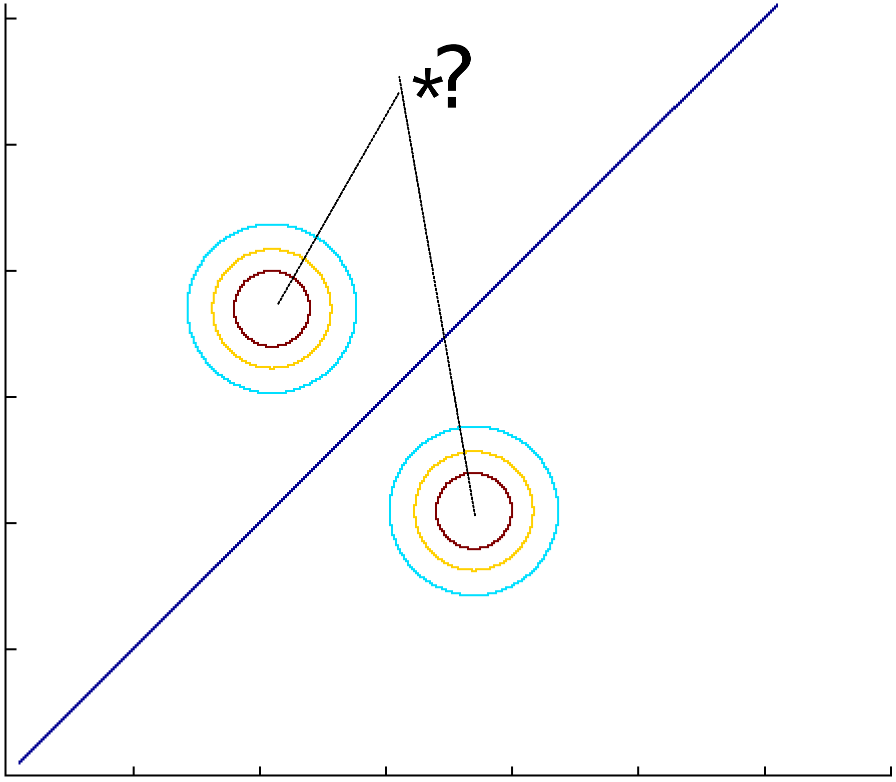
- Nearest mean classifier: Classify based on Euclidean distance to the nearest mean

$$g_i(\boldsymbol{x}) = -\frac{\left\|\boldsymbol{x} - \boldsymbol{m}_i\right\|^2}{2s^2} + \log \hat{P}(C_i)$$

$$= -\frac{1}{2s^2}\sum_{j=1}^{d}\left(x_j^t - m_{ij}\right)^2 + \log \hat{P}(C_i)$$

- Each mean can be considered a prototype or template and this is template matching

Diagonal **S**, equal variances



Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Model Selection

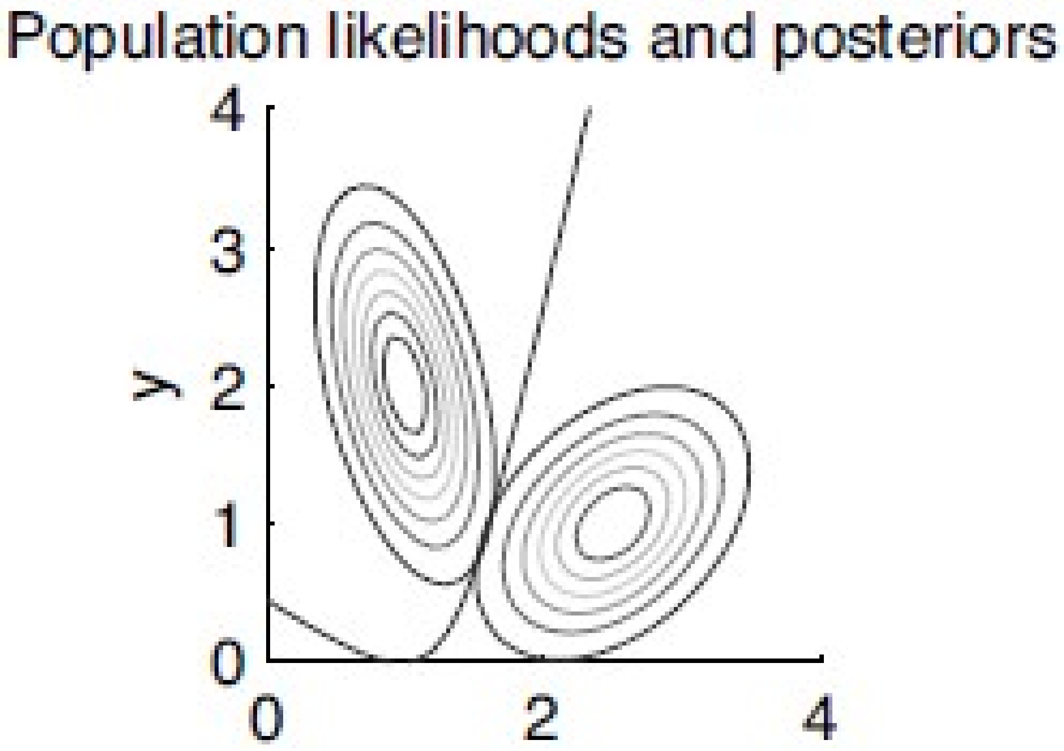
- Different covariance matrix for each class
- Have to estimate many parameters
- Small bias , large variance
- Common covariance matrices, diagonal covariance etc. reduce number of parameters
- Increase bias but control variance
- In-between states?

Regularized Discriminant Analysis(RDA)

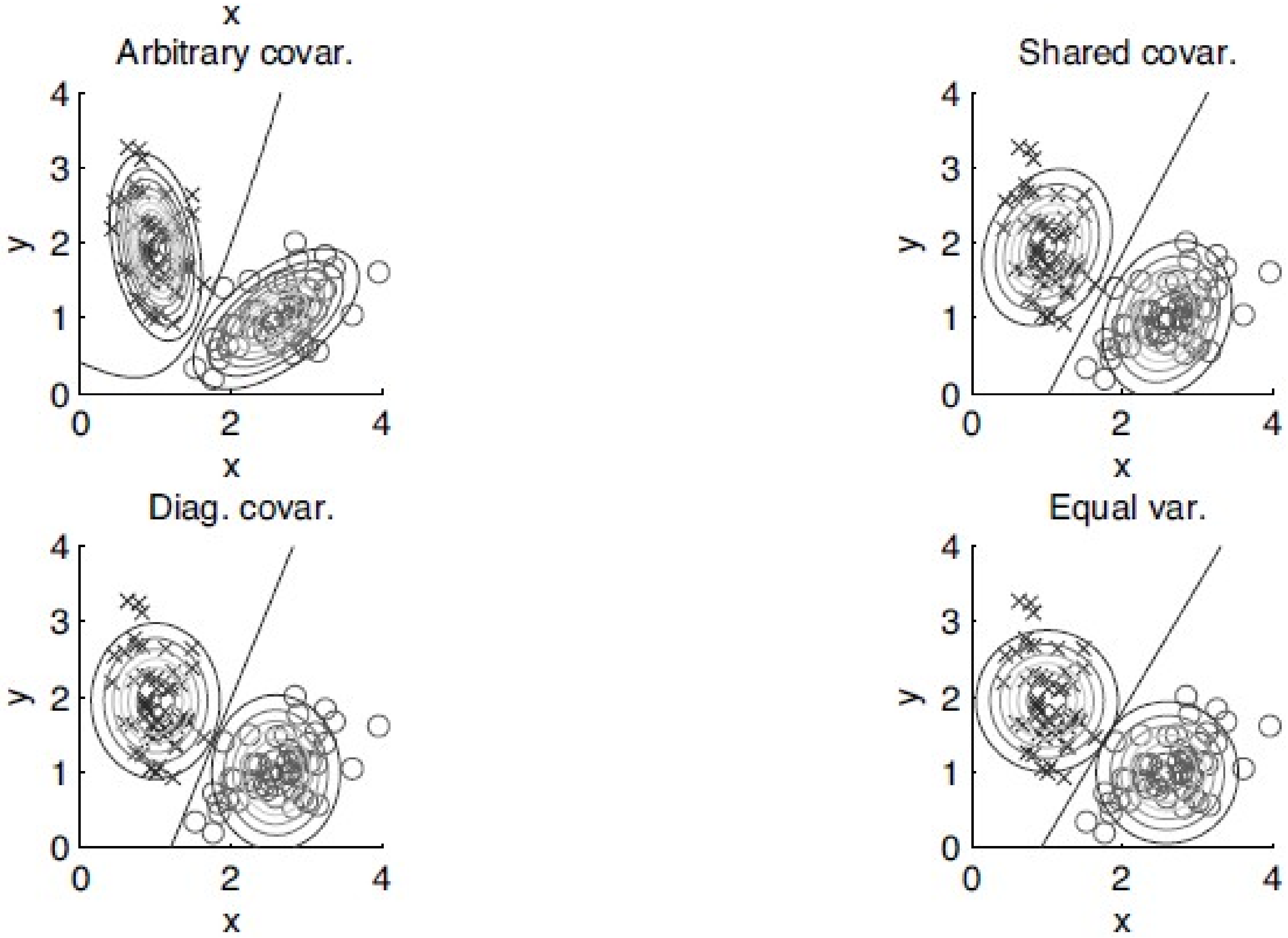
$$\mathbf{S}_i' = \alpha\sigma^2\mathbf{I} + \beta\mathbf{S} + (1 - \alpha - \beta)\mathbf{S}_i$$

- a=b=0: Quadratic classifier
- a=0, b=1:Shared Covariance, linear classifier
- a=1,b=0: Diagonal Covariance
- Choose best a,b by cross validation

Model Selection: Example



Model Selection



Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)