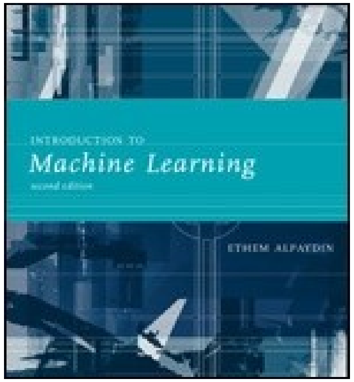
Lecture Slides for

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

This class: Ch 5: Multivariate Methods

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

CHAPTER 5:

Multivariate Methods

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

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

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

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

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

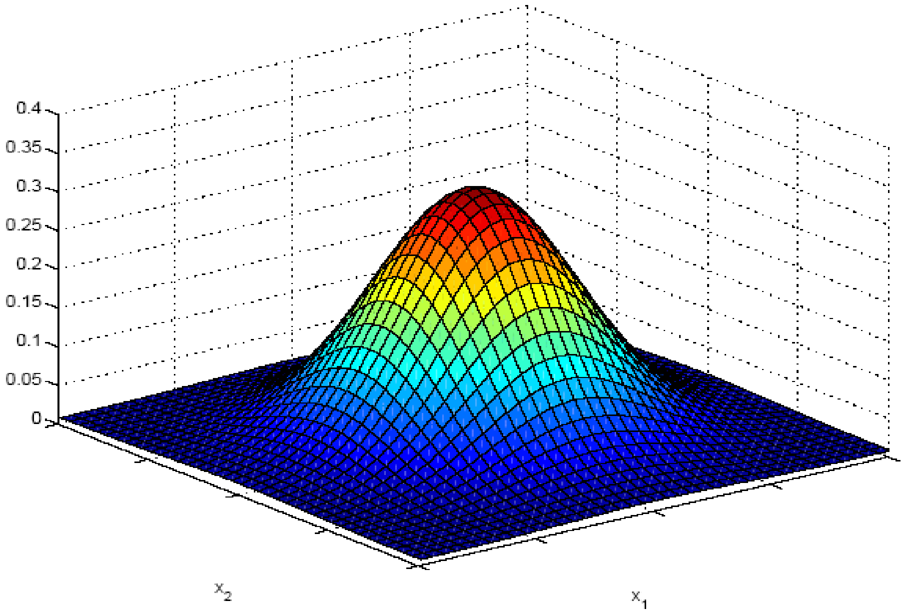
Multivariate Normal

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

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

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

Bivariate Normal



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

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

Multivariate Normal Distribution

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

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

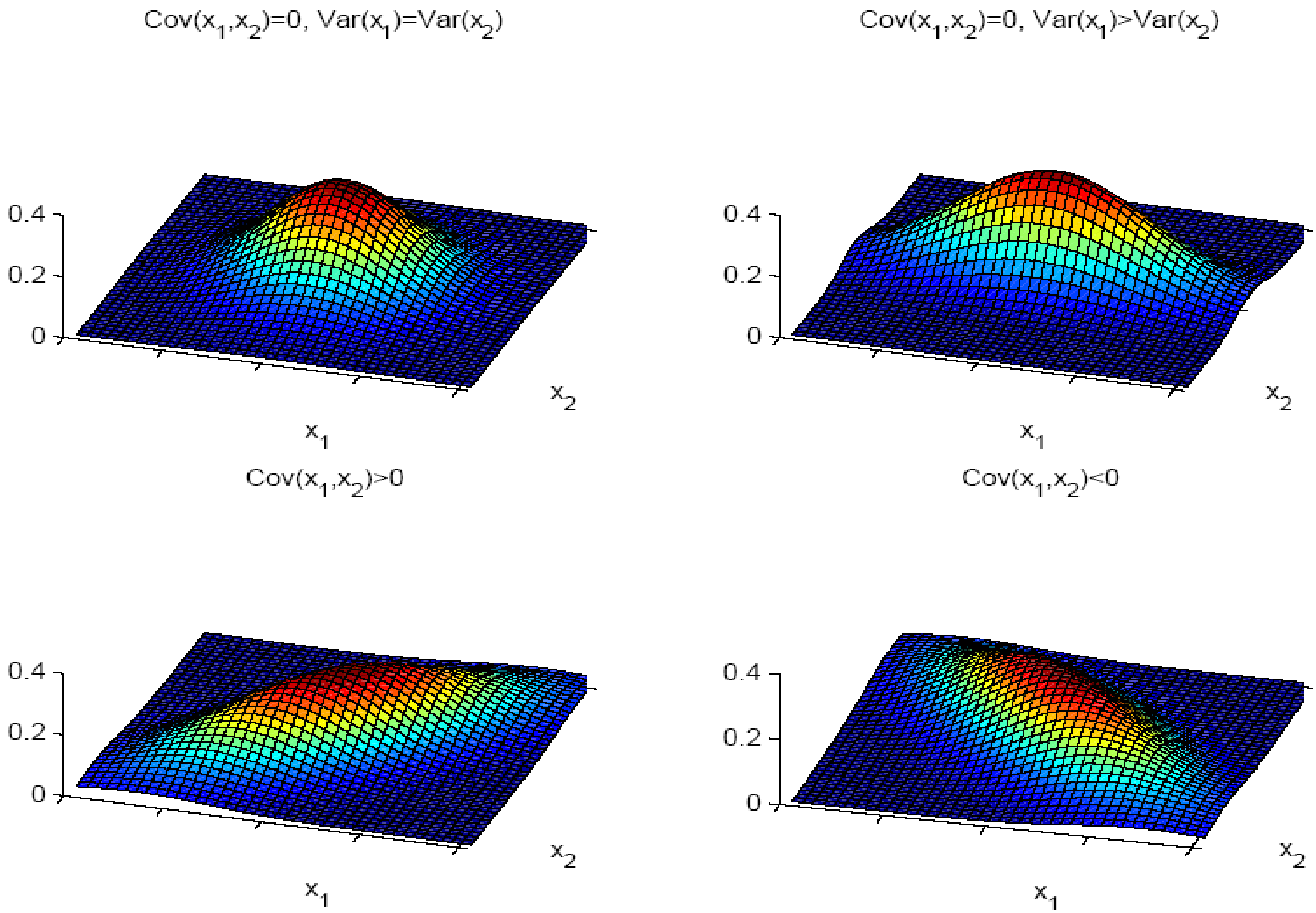
- Bivariate: $d = 2$

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

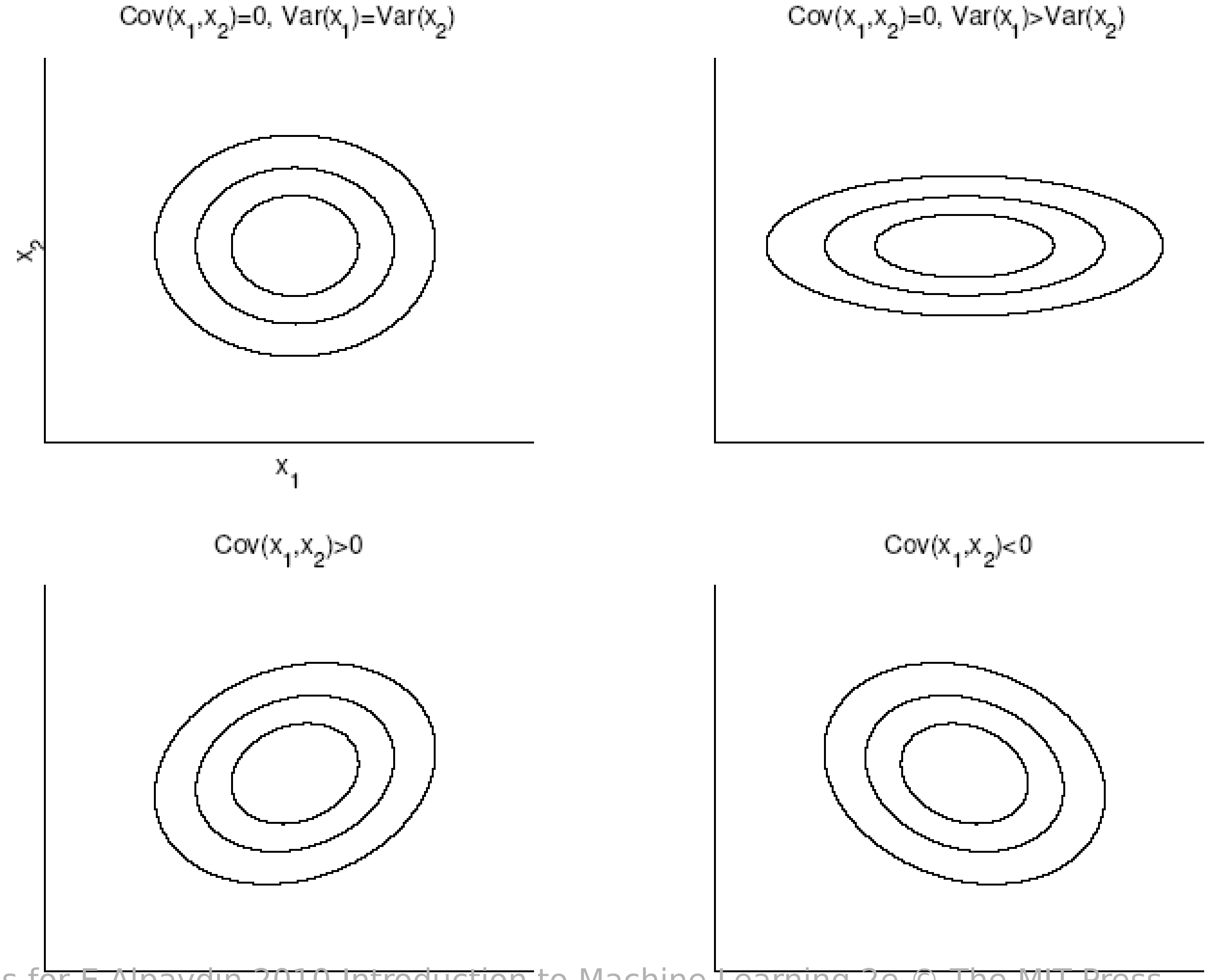
$$p(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left[-\frac{1}{2(1-\rho^2)}\left(z_1^2 - 2\rho z_1 z_2 + z_2^2\right)\right]$$

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

Bivariate Normal



Bivariate Normal



Independent Inputs: Naive Bayes

- If x_i are independent, offdiagonals of Σ 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

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

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

1D projection

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

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

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

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

- 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

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

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

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

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

Covariance Matrix per Class

- Quadratic discriminant

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

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

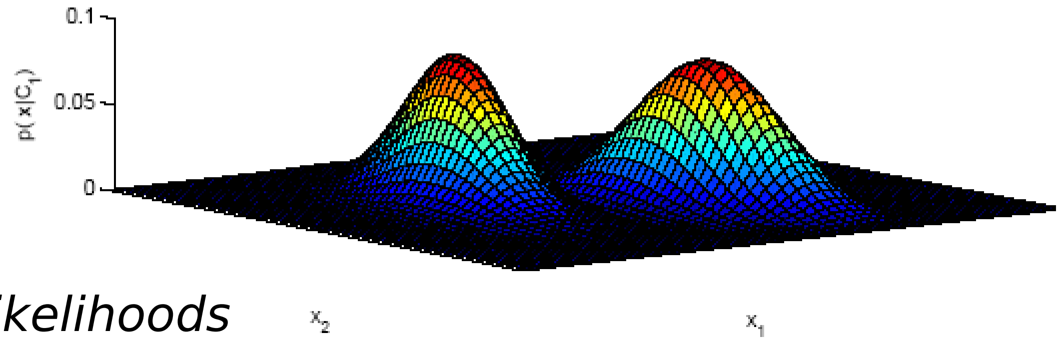
where

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

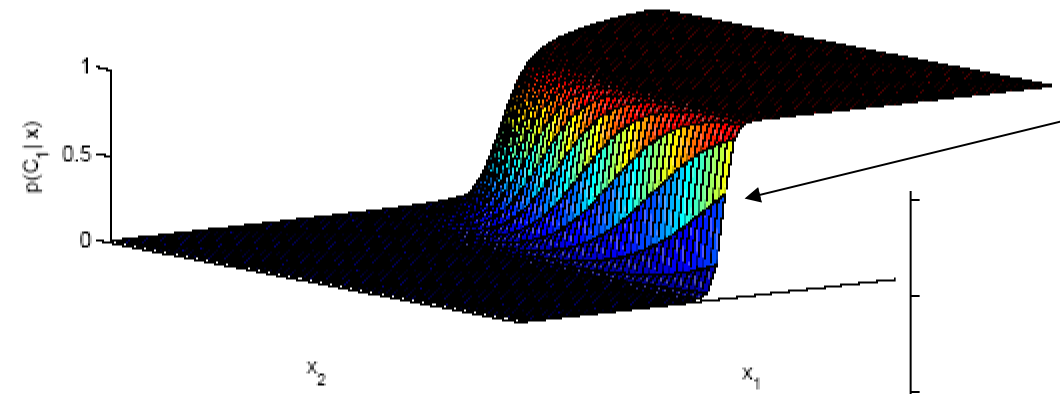
$$\boldsymbol{w}_i = \mathbf{S}_i^{-1}\boldsymbol{m}_i$$

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

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

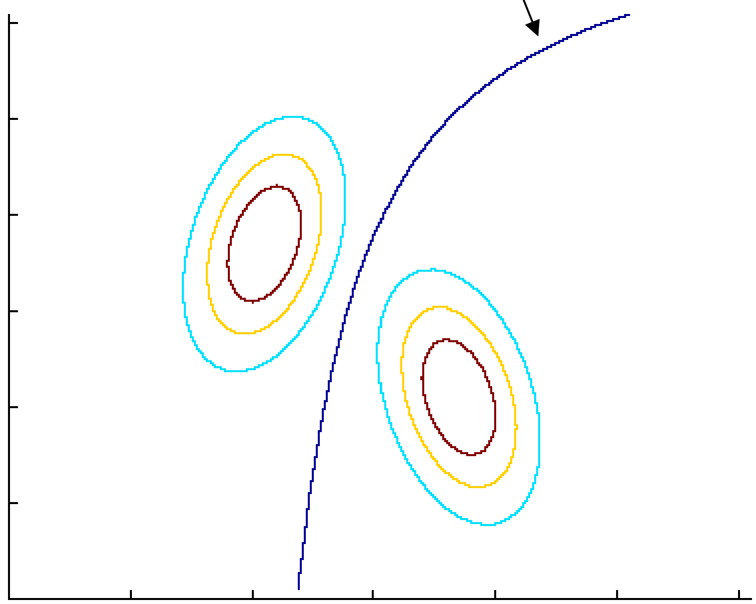


likelihoods



posterior for C_1

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



Common Covariance Matrix **S**

- If not enough data can assume all classes have same common sample covariance matrix **S** $$\mathbf{S} = \sum_i \hat{P}(C_i)\mathbf{S}_i$$

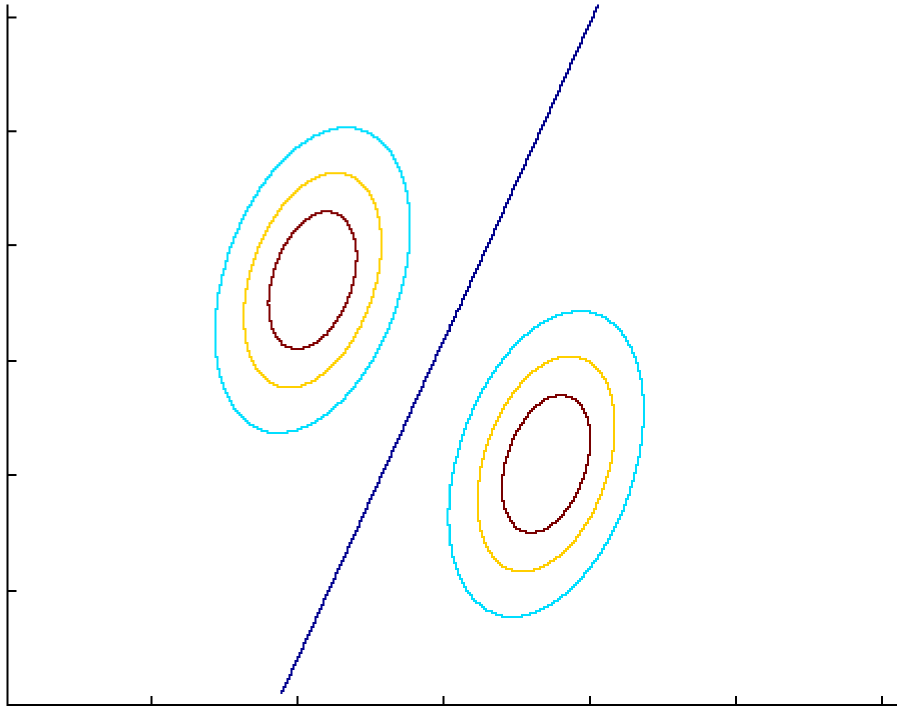
  Discriminant reduces to a linear discriminant ($x^T S^{-1} x$ is common to all discriminant and can be removed)

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

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

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

Common Covariance Matrix **S**



Diagonal **S**

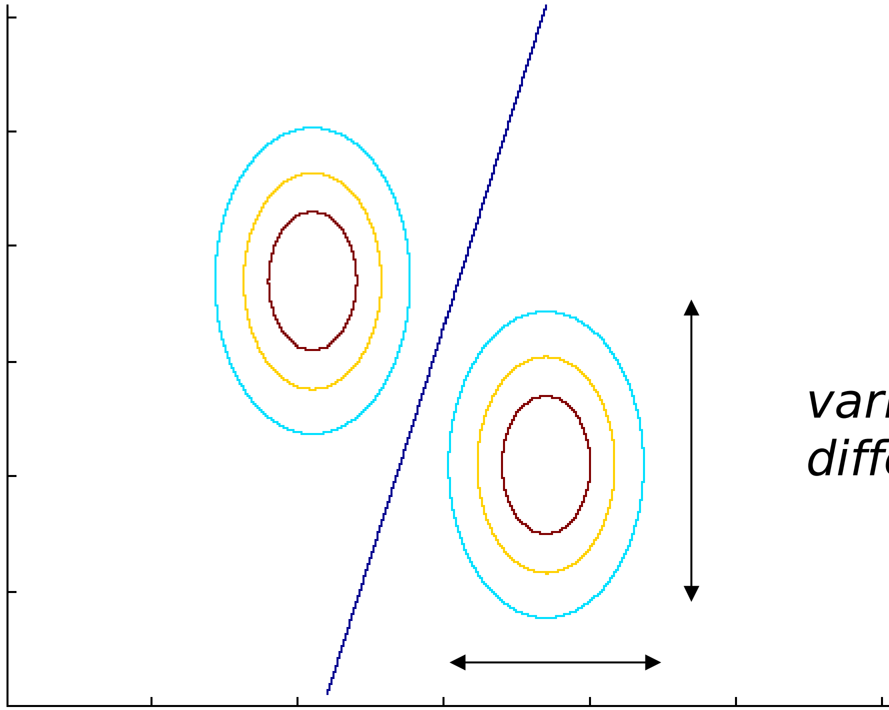
- When x_j $j = 1,..d$, are independent, $\sum$ is diagonal

  $p(\mathbf{x}|C_i) = \prod_j p(x_j|C_i)$ (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

Diagonal **S**, equal variances

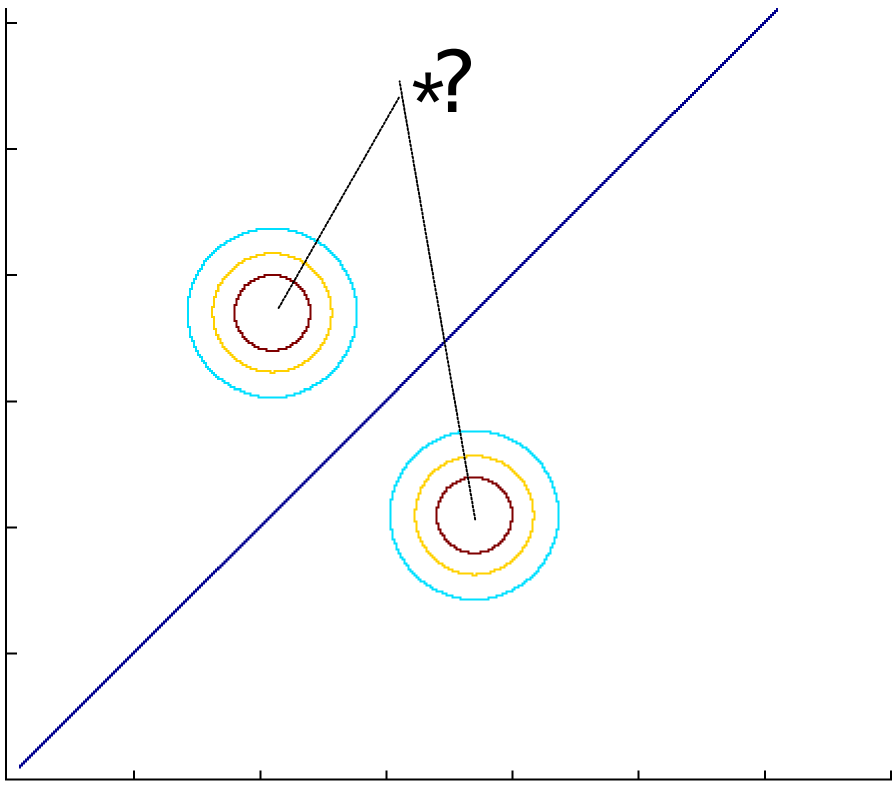
- Nearest mean classifier: Classify based on Euclidean distance to the nearest mean

$$g_i(x) = -\frac{\|x - m_i\|^2}{2s^2} + \log \hat{P}(C_i)$$

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

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

Diagonal **S**, equal variances



Model Selection

Assumption	*Covariance matrix*	*No of parameters*
Shared, Hyperspheric	$\mathbf{S}_i=\mathbf{S}=s^2\mathbf{I}$	1
Shared, Axis-aligned	$\mathbf{S}_i=\mathbf{S}$, with $s_{ij}=0$	d
Shared, Hyperellipsoidal	$\mathbf{S}_i=\mathbf{S}$	$d(d+1)/2$
Different, Hyperellipsoidal	$\mathbf{S}_i$	$K\ d(d+1)/2$

- As we increase complexity (less restricted **S**), bias decreases and variance increases
- Assume simple models (allow some bias) to control variance (regularization)

Model Selection

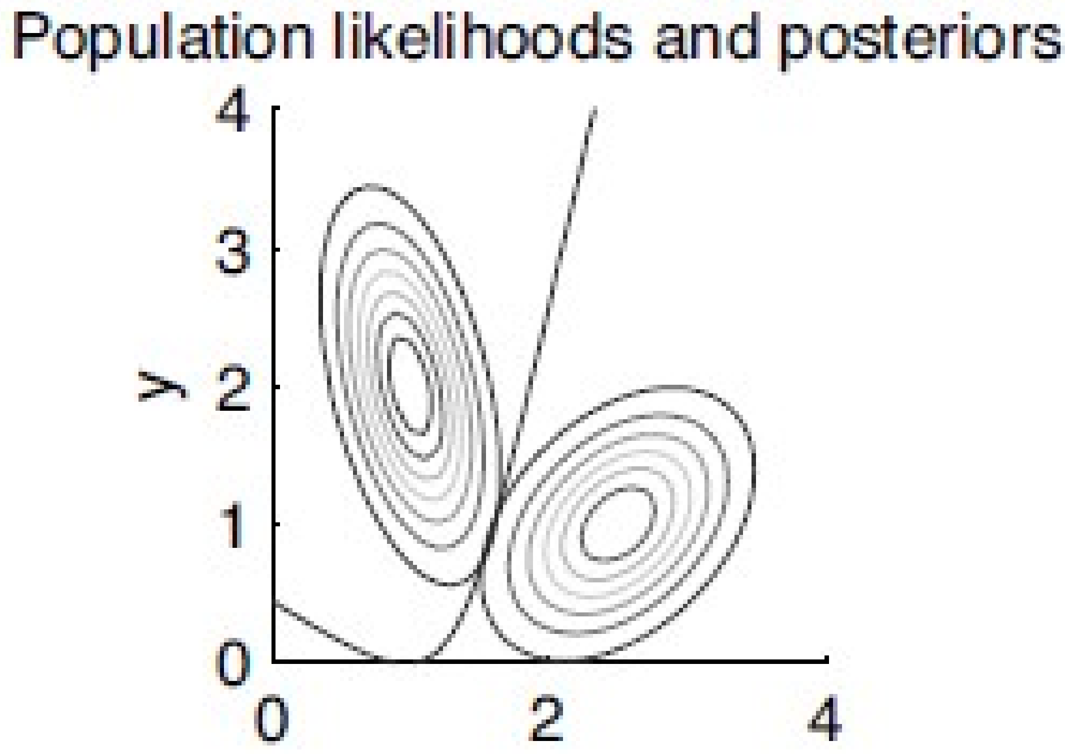
- Different covariance matrix for each class
- Have to estimate many parameters
- Small bias , large variance
- Common covariance matrices, diagonal covariance etc. reduce number of parameters
- Increase bias but control variance
- In-between states?

Regularized Discriminant Analysis(RDA)

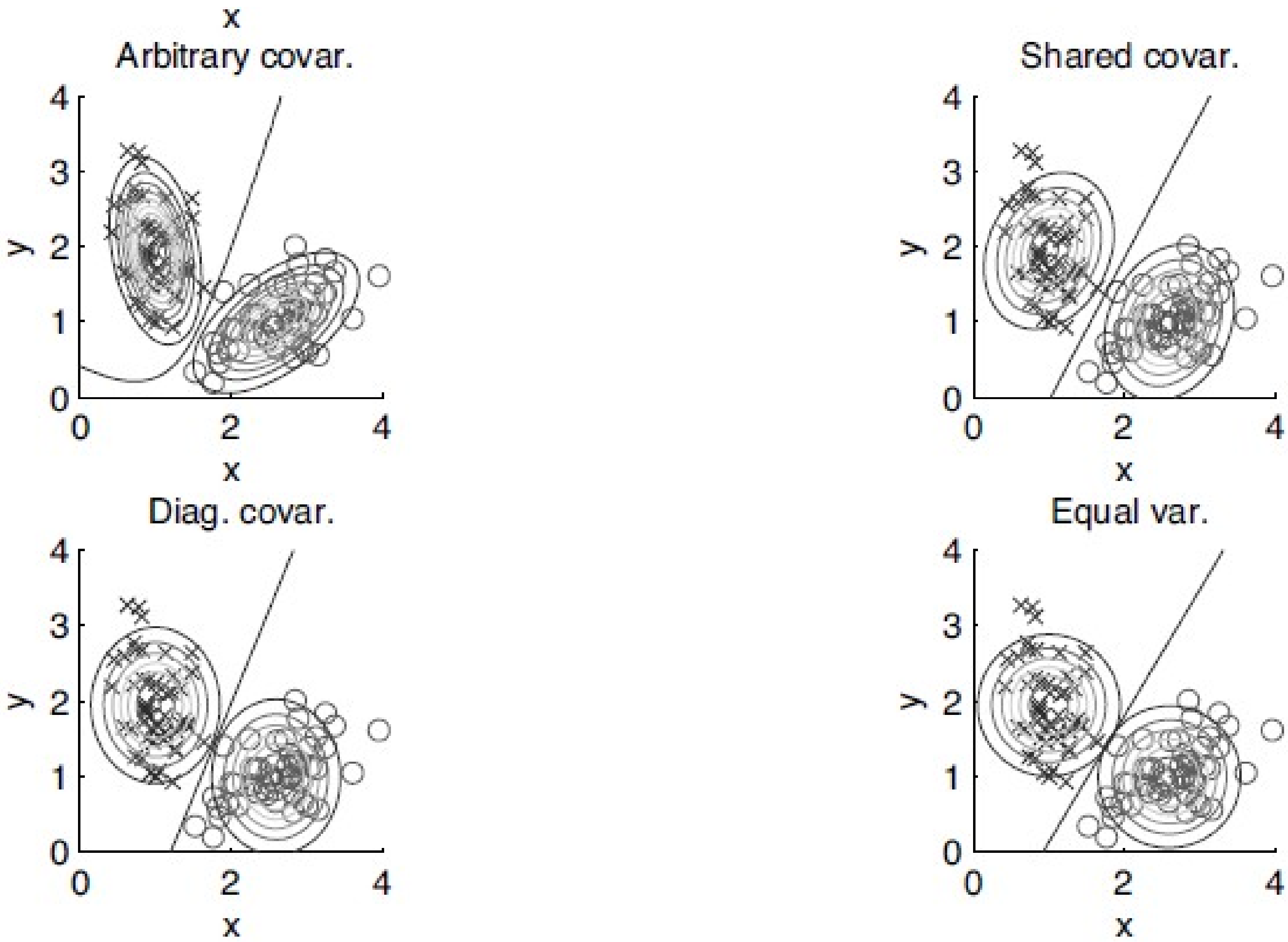
$$\mathbf{S}_i' = \alpha\sigma^2\mathbf{I} + \beta\mathbf{S} + (1-\alpha-\beta)\mathbf{S}_i$$

- 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



Discrete Features

- Binary features: $p_{ij} \equiv p(x_j = 1 | C_i)$

  if x_j are independent (Naive Bayes')

  $$p(x|C_i) = \prod_{j=1}^{d} p_{ij}^{x_j} (1 - p_{ij})^{(1-x_j)}$$

  the discriminant is linear

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

  $$= \sum_j \left[x_j \log p_{ij} + (1 - x_j)\log(1 - p_{ij})\right] + \log P(C_i)$$

  Estimated parameters $\hat{p}_{ij} = \frac{\sum_t x_j^t r_i^t}{\sum_t r_i^t}$

Multivariate Regression

$$r^t = g\left(x^t \mid w_0, w_1, \ldots, w_d\right) + \varepsilon$$

- Multivariate linear model

$$w_0 + w_1 x_1^t + w_2 x_2^t + \cdots + w_d x_d^t$$

- $$E\left(w_0, w_1, \ldots, w_d \mid \mathcal{X}\right) = \frac{1}{2}\sum_t \left[r^t - w_0 - w_1 x_1^t - \cdots - w_d x_d^t\right]^2$$

Multivariate Regression

$$w_0 + w_1 x_1^t + w_2 x_2^t + \cdots + w_d x_d^t$$

- $$E(w_0, w_1, \ldots, w_d \mid \mathcal{X}) = \frac{1}{2} \sum_t \left[r^t - w_0 - w_1 x_1^t - \cdots - w_d x_d^t \right]^2$$

- $$\begin{aligned}
\sum_t r^t &= N w_0 + w_1 \sum_t x_1^t + w_2 \sum_t x_2^t + \cdots + w_d \sum_t x_d^t \\
\sum_t x_1^t r^t &= w_0 \sum_t x_1^t + w_1 \sum_t (x_1^t)^2 + w_2 \sum_t x_1^t x_2^t + \cdots + w_d \sum_t x_1^t x_d^t \\
\sum_t x_2^t r^t &= w_0 \sum_t x_2^t + w_1 \sum_t x_1^t x_2^t + w_2 \sum_t (x_2^t)^2 + \cdots + w_d \sum_t x_2^t x_d^t \\
&\vdots \\
\sum_t x_d^t r^t &= w_0 \sum_t x_d^t + w_1 \sum_t x_d^t x_1^t + w_2 \sum_t x_d^t x_2^t + \cdots + w_d \sum_t (x_d^t)^2
\end{aligned}$$

CHAPTER 6:

Dimensionality Reduction

Dimensionality of input

- Number of Observables (e.g. age and income)
- If number of observables is increased
  - More time to compute
  - More memory to store inputs and intermediate results
  - More complicated explanations (knowledge from learning)
    - Regression from 100 vs. 2 parameters
  - No simple visualization
    - 2D vs. 10D graph
  - **Need much more data (curse of dimensionality)**
    - 1M of 1-d inputs is not equal to 1 input of dimension 1M

Dimensionality reduction

- Some features (dimensions) bear little or nor useful information (e.g. color of hair for a car selection)
  - Can drop some features
  - Have to estimate which features can be dropped from data

- Several features can be combined together without loss or even with gain of information (e.g. income of all family members for loan application)
  - Some features can be combined together
  - Have to estimate which features to combine from

Feature Selection vs Extraction

- Feature selection: Choosing $k<d$ important features, ignoring the remaining $d - k$
  - Subset selection algorithms
- Feature extraction: Project the original x_i , i =1,...,d dimensions to new $k<d$ dimensions, z_j , j =1,...,k
  - Principal Components Analysis (PCA)
  - Linear Discriminant Analysis (LDA)
  - Factor Analysis (FA)

Usage

- Have data of dimension d
- Reduce dimensionality to k<d
  - Discard unimportant features
  - Combine several features in one
- Use resulting k-dimensional data set for
  - Learning for classification problem (e.g. parameters of probabilities P(x|C)
  - Learning for regression problem (e.g. parameters for model y=g(x|Thetha)

Subset selection

- Have initial set of features of size d
- There are 2^d possible subsets
- Need a criteria to decide which subset is the best
- A way to search over the possible subsets
- Can’t go over all 2^d possibilities
- Need some heuristics

“Goodness” of feature set

- Supervised
  - Train using selected subset
  - Estimate error on validation data set
- Unsupervised
  - Look at input only(e.g. age, income and savings)
  - Select subset of 2 that bear most of the information about the person

Mutual Information

- Have a 3 random variables(features) X,Y,Z and have to select 2 which gives most information
- If X and Y are "correlated" then much of the information about of Y is already in X
- Make sense to select features which are "uncorrelated"
- Mutual Information (Kullback–Leibler Divergence) is more general measure of "mutual information"
- Can be extended to n variables (information variables x_1,.. x_n have about variable x_{n+1})

Subset-selection

- Forward search
  - Start from empty set of features
  - Try each of remaining features
  - Estimate classification/regression error for adding specific feature
  - Select feature that gives maximum improvement in validation error
  - Stop when no significant improvement

- Backward search
  - Start with original set of size d
  - Drop features with smallest impact on error

Subset Selection

- There are 2^d subsets of d features
- Forward search: Add the best feature at each step
  - Set of features F initially Ø.
  - At each iteration, find the best new feature

    $j = \text{argmin}_i \, E (F \cup x_i)$

    - Add x_j to F if $E (F \cup x_j) < E (F)$

- Hill-climbing $O(d^2)$ algorithm
- Backward search: Start with all features and remove one at a time, if possible.
- Floating search (Add k, remove l)

Floating Search

- Forward and backward search are “greedy” algorithms
  - Select best options at single step
  - Do not always achieve optimum value

- Floating search
  - Two types of steps: Add *k*, remove *l*
  - *More computations*

Feature Extraction

- Face recognition problem
  - Training data input: pairs of Image + Label(name)
  - Classifier input: Image
  - Classifier output: Label(Name)
- Image: Matrix of 256X256=65536 values in range 0..256
- Each pixels bear little information so can’t select 100 best ones
- Average of pixels around specific positions may give an indication about an eye color.

Projection

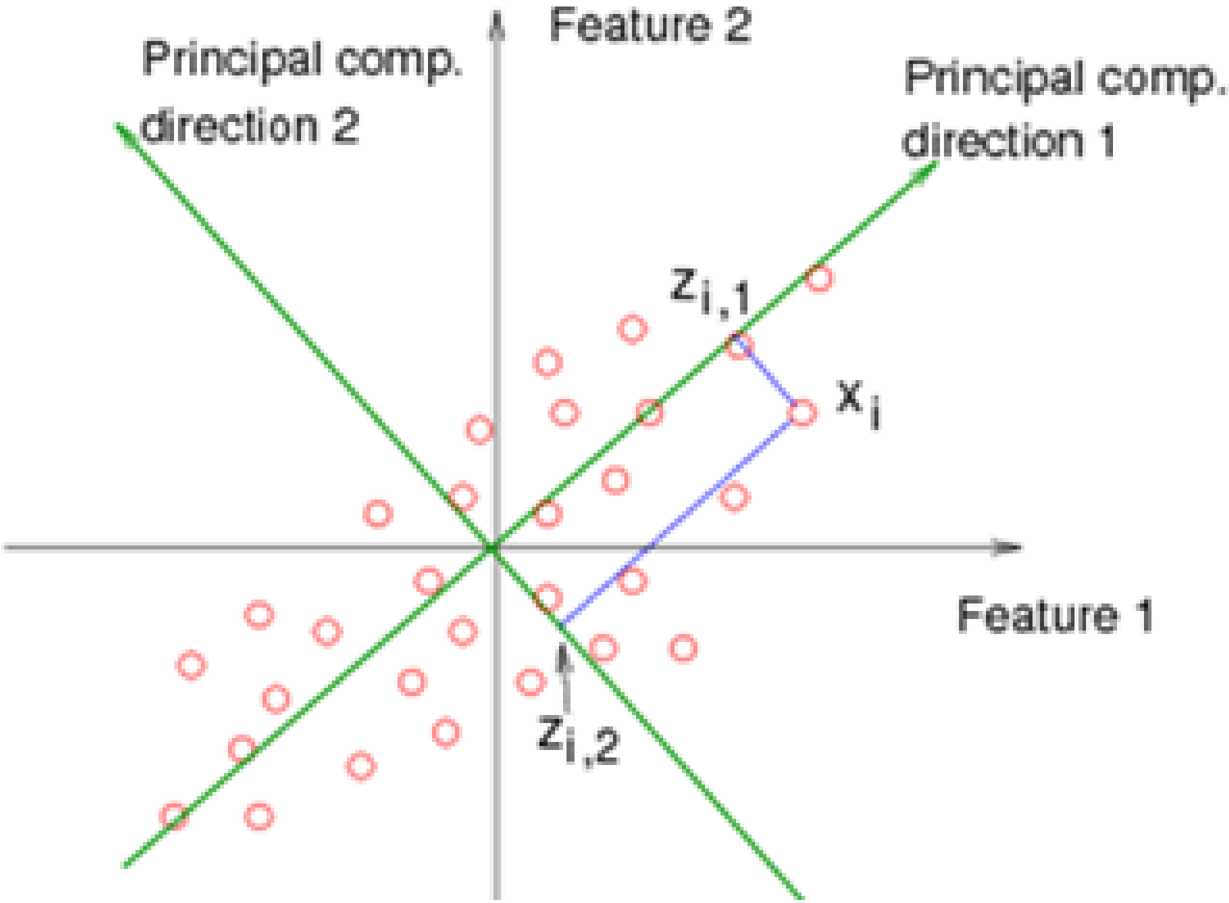
- Find a projection matrix w from d-dimensional to k-dimensional vectors that keeps error low

$$z = \mathbf{w}^T \mathbf{x}$$

PCA: Motivation

- Assume that d observables are linear combination of k<d vectors
- $z_i = w_{i1}x_{i1} + \ldots + w_{ik}x_{id}$
- We would like to work with basis as it has lesser dimension and have all(almost) required information
- What we expect from such basis
  - Uncorrelated or otherwise can be reduced further
  - Have large variance (e.g. w_{i1} have large variation) or otherwise bear no information

PCA: Motivation



PCA: Motivation

- Choose directions such that a total variance of data will be maximum
  - Maximize Total Variance

- Choose directions that are orthogonal
  - Minimize correlation

- Choose $k<d$ orthogonal directions which maximize total variance

PCA

- Choosing only directions: $\|\boldsymbol{w}_1\| = 1$
- $z_1 = \boldsymbol{w}_1^T \boldsymbol{x}$ $\quad \mathrm{Cov}(\boldsymbol{x}) = \boldsymbol{\Sigma},$ $\quad \mathrm{Var}(z_1) = \boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1$
- Maximize variance subject to a constrain using Lagrange Multipliers

$$\max_{\boldsymbol{w}_1} \boldsymbol{w}_1^T \boldsymbol{\Sigma} \boldsymbol{w}_1 - \alpha(\boldsymbol{w}_1^T \boldsymbol{w}_1 - 1)$$

- Taking Derivatives

$$2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\alpha\boldsymbol{w}_1 = 0 \qquad \boldsymbol{\Sigma}\boldsymbol{w}_1 = \alpha\boldsymbol{w}_1$$

- Eigenvector. Since want to maximize $w_1^T \Sigma w_1 = \alpha w_1^T w_1 = \alpha$ we should choose an eigenvector with largest eigenvalue

PCA

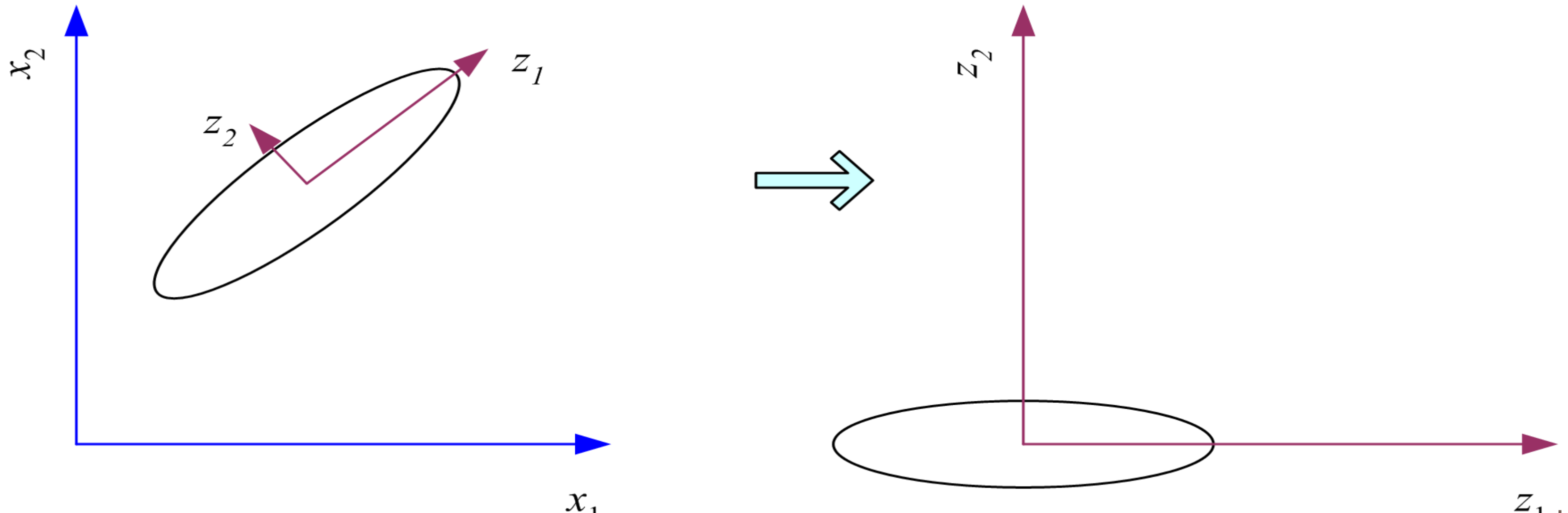
- d-dimensional feature space
- d by d symmetric covariance matrix estimated from samples $\text{Cov}(\boldsymbol{x}) = \Sigma,$
- Select k largest eigenvalue of the covariance matrix and associated k eigenvectors
- The first eigenvector will be a direction with largest variance

What PCA does

$$\boldsymbol{z} = \mathbf{W}^T(\boldsymbol{x} - \boldsymbol{m})$$

where the columns of **W** are the eigenvectors of $\sum$, and ***m*** is sample mean

Centers the data at the origin and rotates the axes



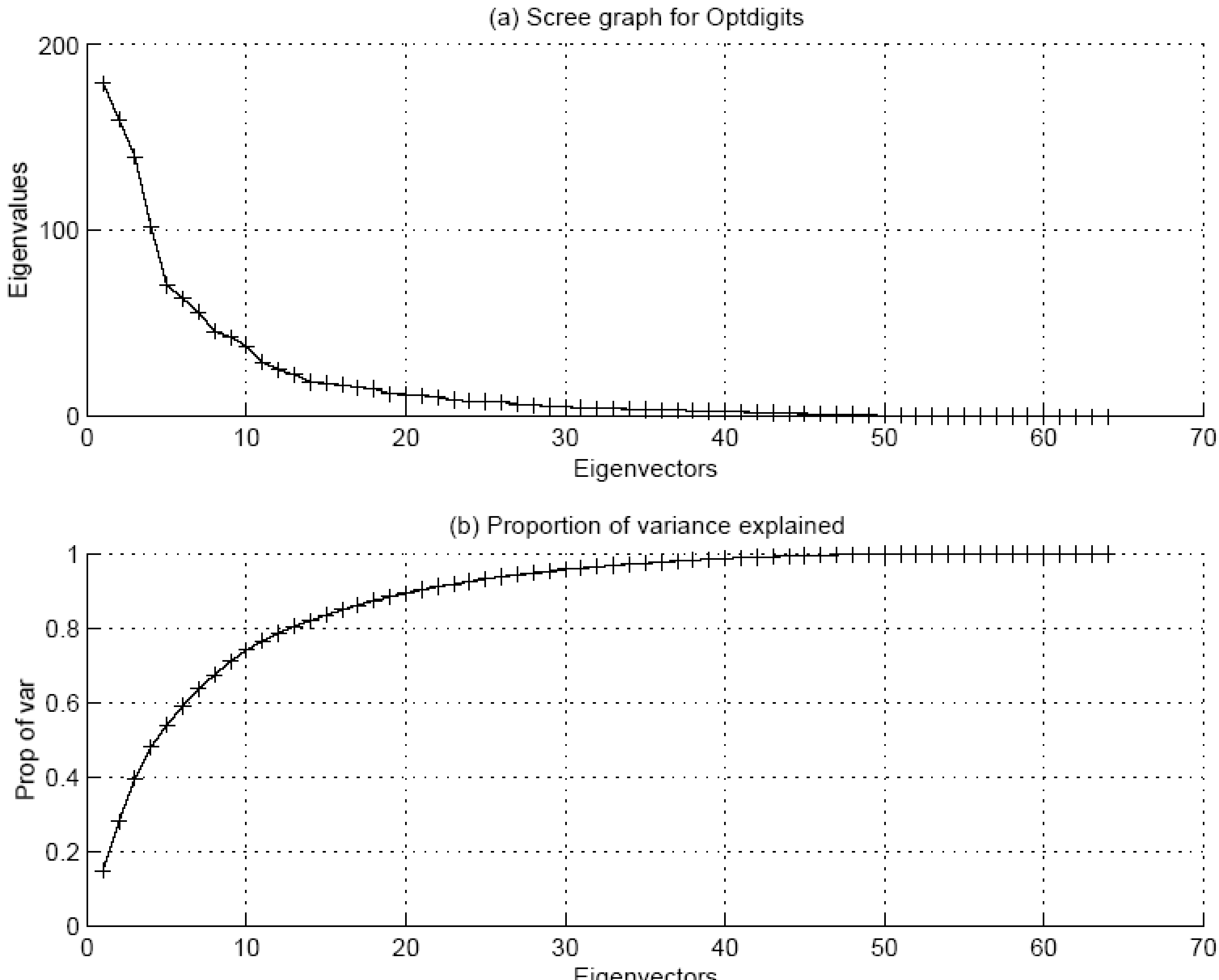
How to choose k ?

- Proportion of Variance (PoV) explained

$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

when λ_i are sorted in descending order

- Typically, stop at PoV>0.9
- Scree graph plots of PoV vs *k*, stop at "elbow"


(a) Scree graph for Optdigits
Eigenvalues
Eigenvectors
200
100
0
0
10
20
30
40
50
60
70
(b) Proportion of variance explained
Prop of var
Eigenvectors
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60
70

PCA

- PCA is unsupervised (does not take into account class information)
- Can take into account classes : Karhuned-Loeve Expansion
  - Estimate Covariance Per Class
  - Take average weighted by prior
- Common Principle Components
  - Assume all classes have same eigenvectors (directions) but different variances

PCA

- Does not try to explain noise
  - Large noise can become new dimension/largest PC
- Interested in resulting uncorrelated variables which explain large portion of **total** sample variance
- Sometimes interested in explained shared variance (common factors) that affect data