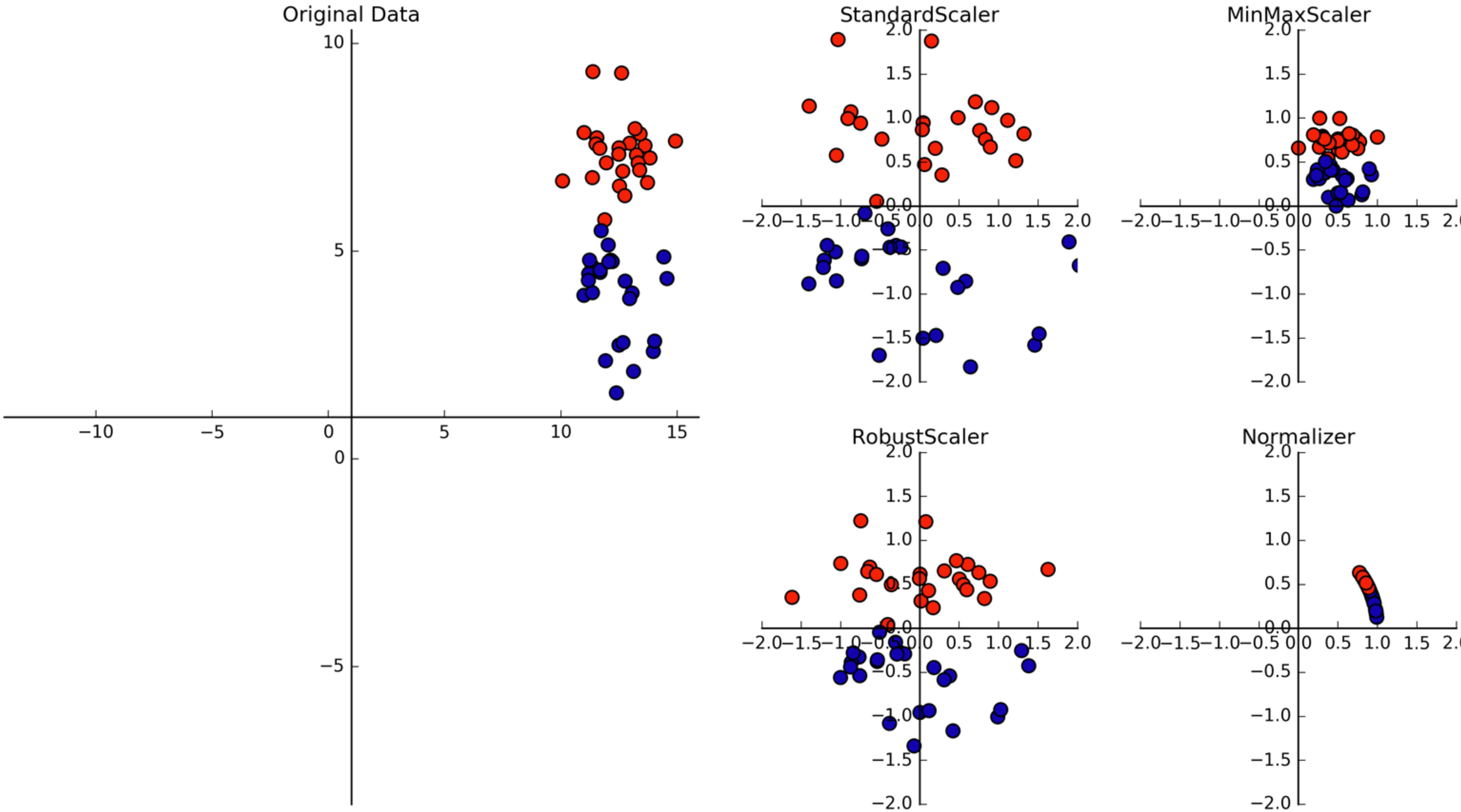
DIMENSIONALITY REDUCTION

Machine Learning Problems

	Supervised Learning	***Unsupervised Learning***
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction


Original Data
StandardScaler
MinMaxScaler
RobustScaler
Normalizer

Why Reduce Dimensionality?

- Reduces time complexity: Less computation
- Reduces space complexity: Fewer parameters
- Saves the cost of observing the feature
- Simpler models are more robust on small datasets
- More interpretable; simpler explanation
- Data visualization (structure, groups, outliers, etc) if plotted in 2 or 3 dimensions (just have to)

Feature Selection vs Extraction

- Feature selection: Choosing $k<d$ important features, ignoring the remaining $d - k$

  Subset selection algorithms

- Feature extraction: Project the

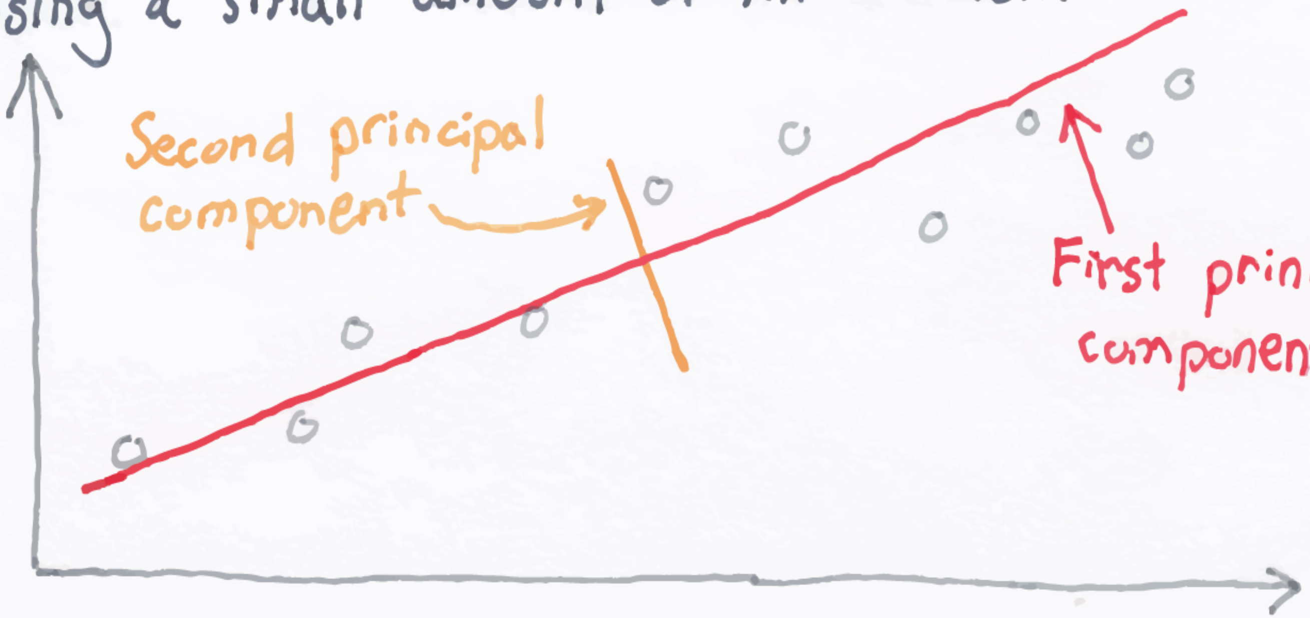
  original x_i , $i = 1,...,d$ dimensions to

  new $k<d$ dimensions, z_j , $j = 1,...,k$

PCA

PRINCIPAL COMPONENT ANALYSIS

PCA projects the features onto the principal components. The motivation is to reduce the features dimensionality while only losing a small amount of information.



Principal Components Analysis

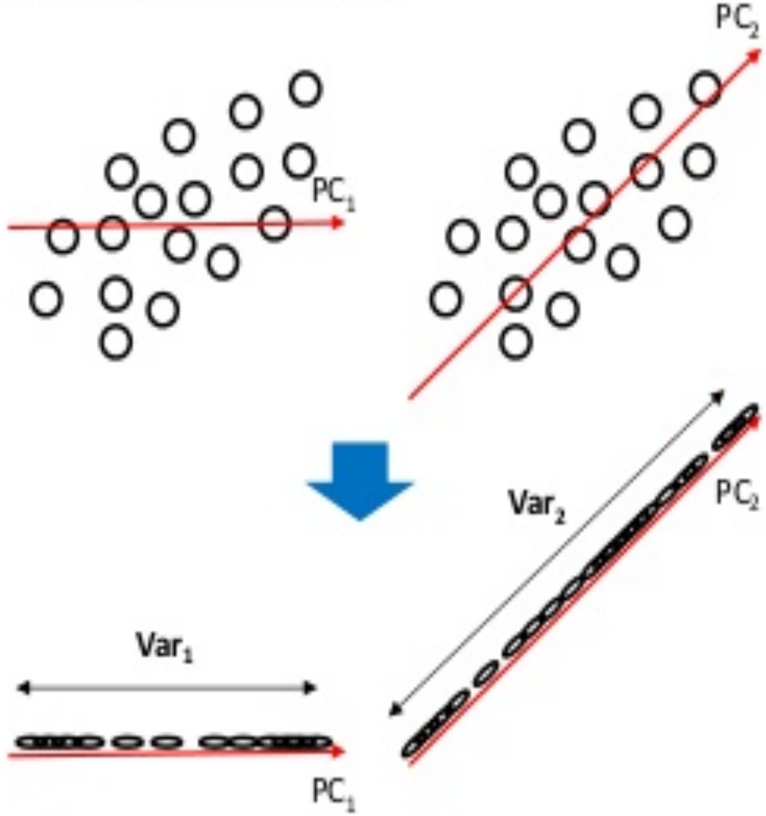
- Find a low-dimensional space such that when **x** is projected there, information loss is minimized.
- The projection of **x** on the direction of ***w*** is: $z = \mathbf{w}^T\mathbf{x}$
- Find ***w*** such that Var(z) is maximized

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

where $\mathrm{Var}(\mathbf{x}) = E[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T] = \sum$

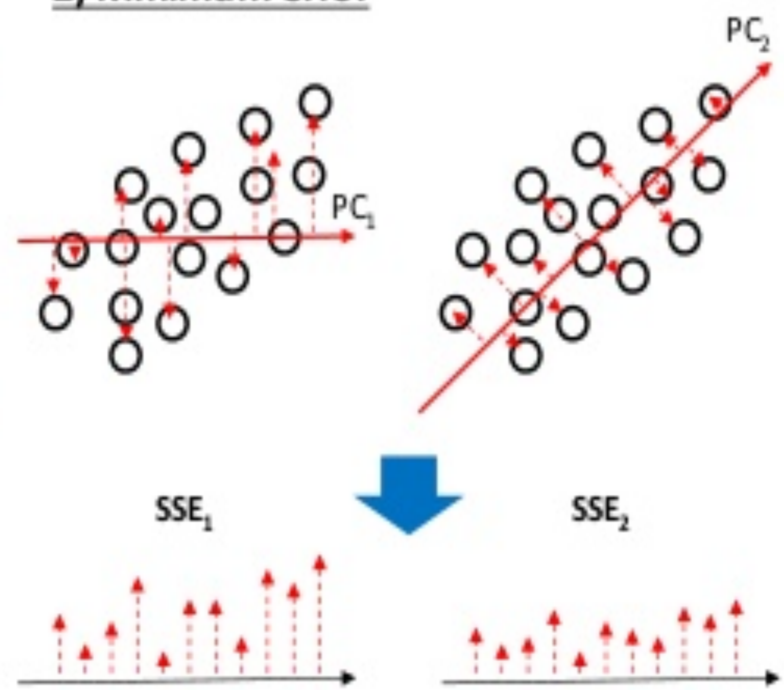
Principles of PCA

1) Maximum variance



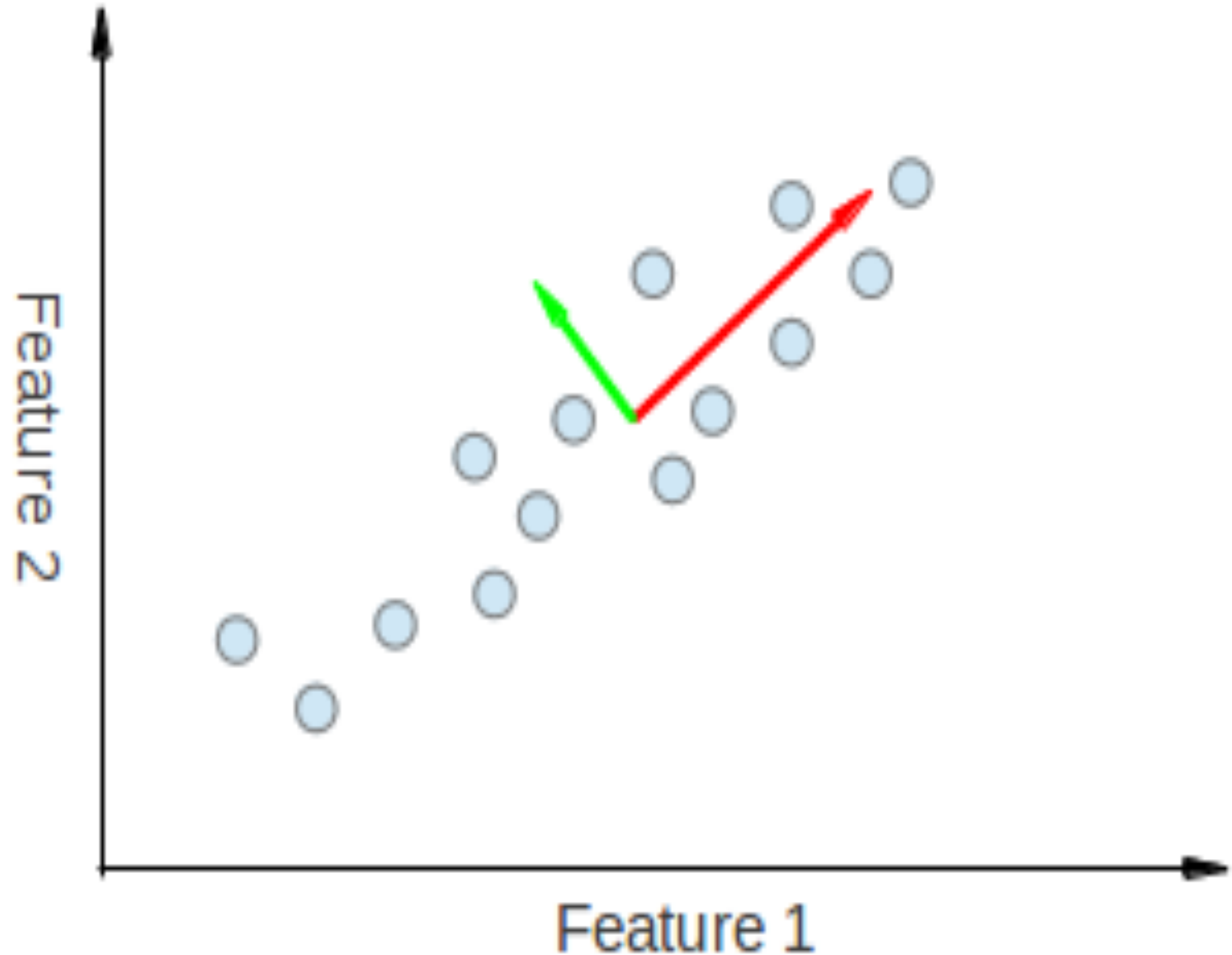
- To maximize the variance of the projected data on the certain dimension.

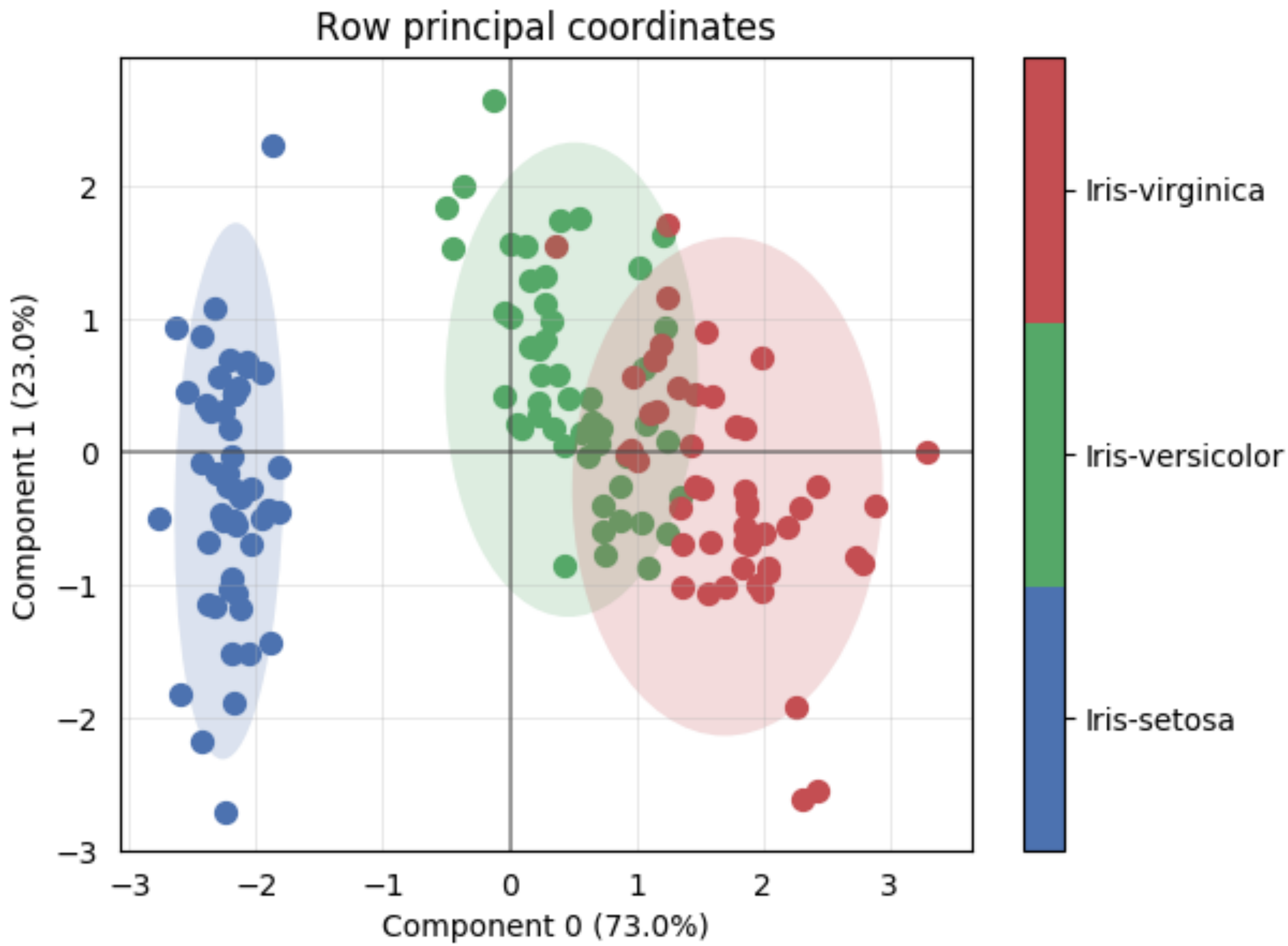
2) Minimum error

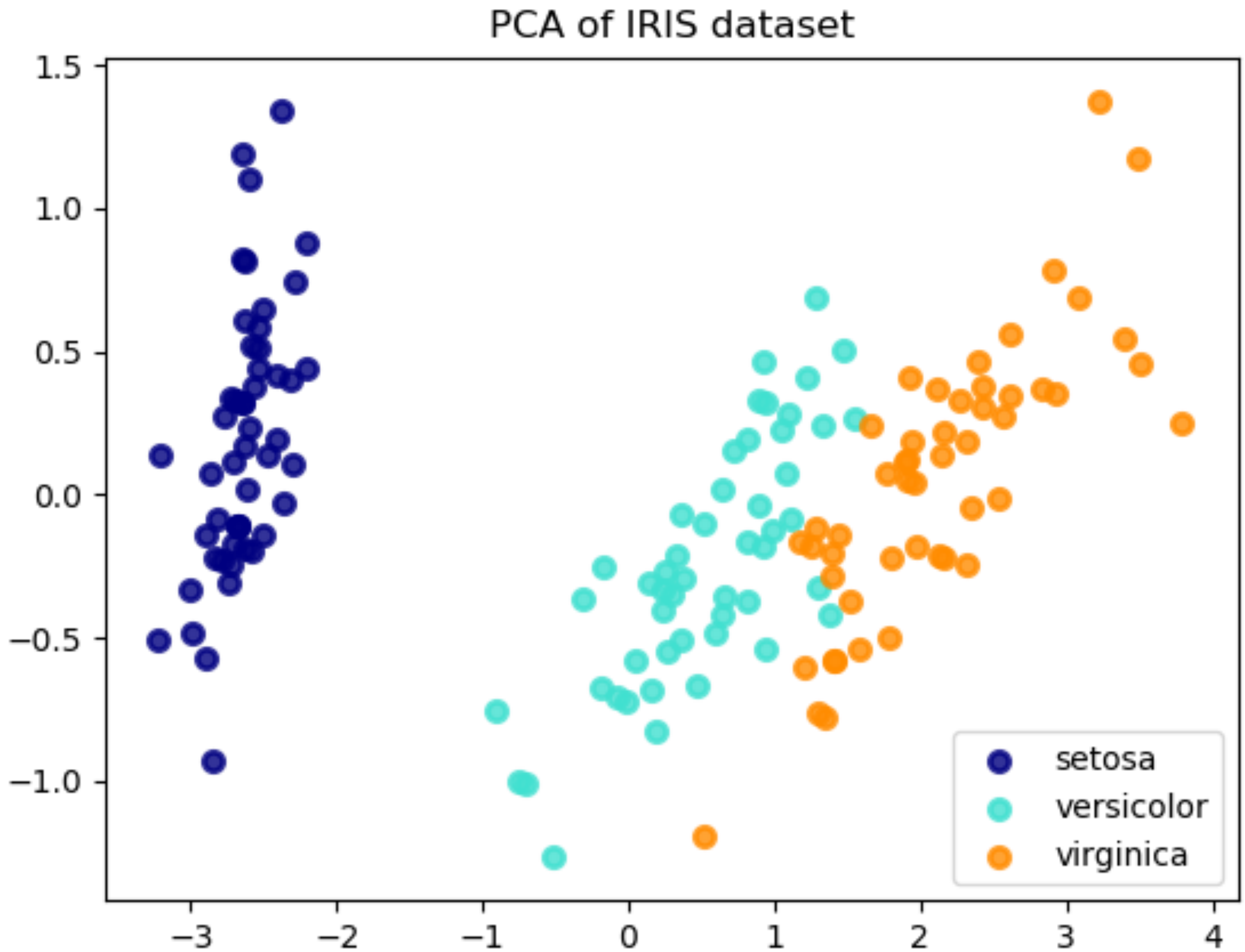


- To minimize the mean squared distance between the data and their projections.

SSE : Sum or squared errors


Feature 2
Feature 1

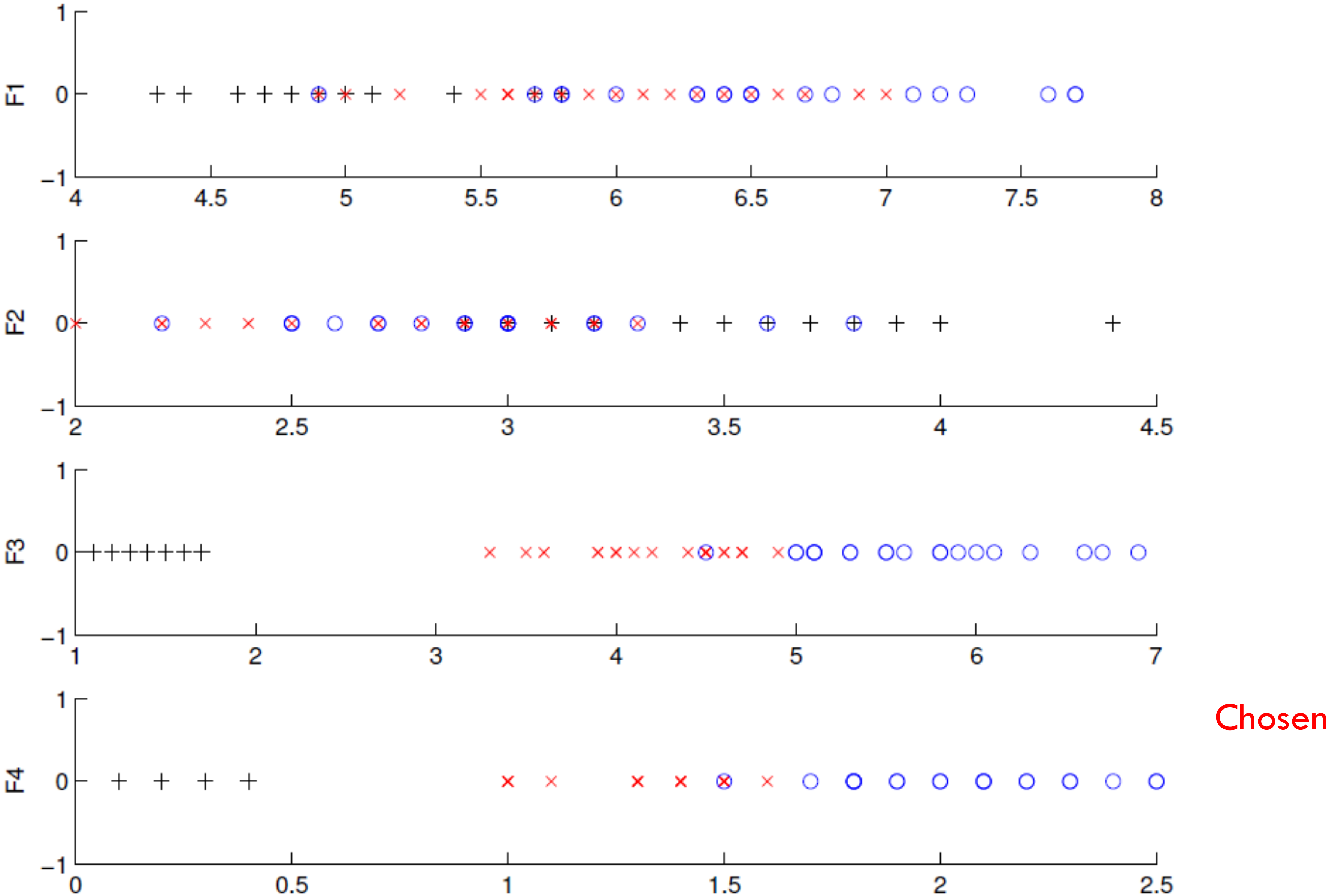

Row principal coordinates
Component 1 (23.0%)
Component 0 (73.0%)
Iris-virginica
Iris-versicolor
Iris-setosa
−3
−2
−1
0
1
2
3


PCA of IRIS dataset
1.5
1.0
0.5
0.0
−0.5
−1.0
−3
−2
−1
0
1
2
3
4
setosa
versicolor
virginica

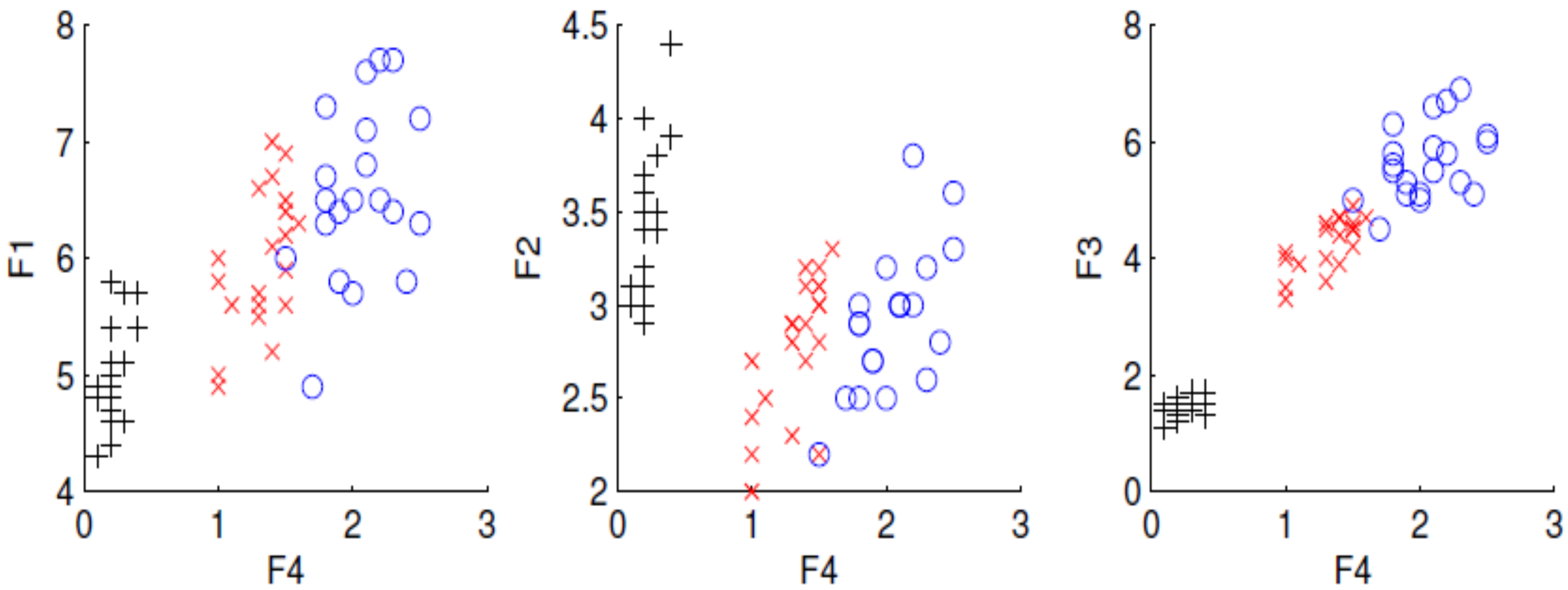
Subset Selection

- 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)

Iris data: Single feature



Iris data: Add one more feature to F4



Chosen

- Maximize Var(z) subject to $||\boldsymbol{w}||=1$

$$\max_{\mathbf{w}_1} \mathbf{w}_1^T \Sigma \mathbf{w}_1 - \alpha\left(\mathbf{w}_1^T \mathbf{w}_1 - 1\right)$$

  $\sum \boldsymbol{w}_1 = \alpha \boldsymbol{w}_1$ that is, $\boldsymbol{w}_1$ is an eigenvector of $\sum$

  Choose the one with the largest eigenvalue for Var(z) to be max

- Second principal component: Max Var(z_2), s.t., $||\boldsymbol{w}_2||=1$ and orthogonal to $\boldsymbol{w}_1$

$$\max_{\mathbf{w}_2} \mathbf{w}_2^T \Sigma \mathbf{w}_2 - \alpha\left(\mathbf{w}_2^T \mathbf{w}_2 - 1\right) - \beta\left(\mathbf{w}_2^T \mathbf{w}_1 - 0\right)$$

  $\sum \boldsymbol{w}_2 = \alpha\, \boldsymbol{w}_2$ that is, $\boldsymbol{w}_2$ is another eigenvector of $\sum$

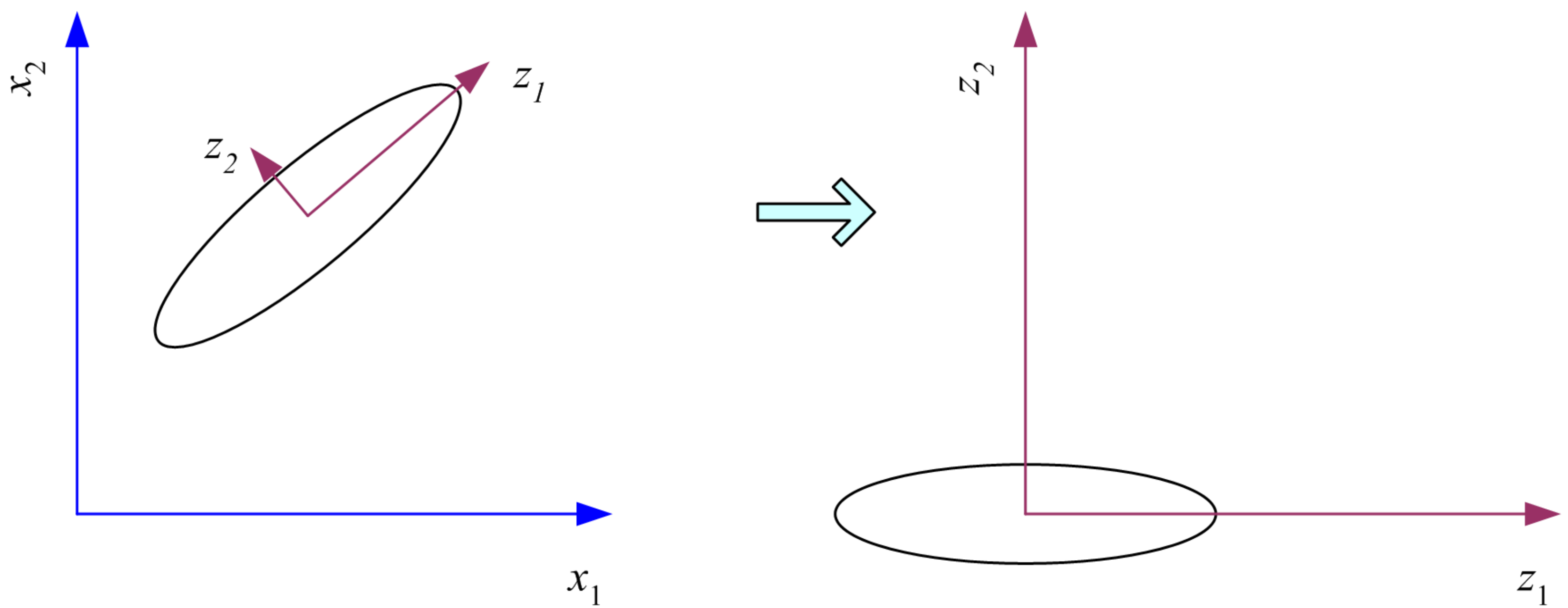
  and so on.

What PCA does

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

where the columns of **W** are the eigenvectors of ∑ and ***m*** is sample mean

Centers the data at the origin and rotates the axes

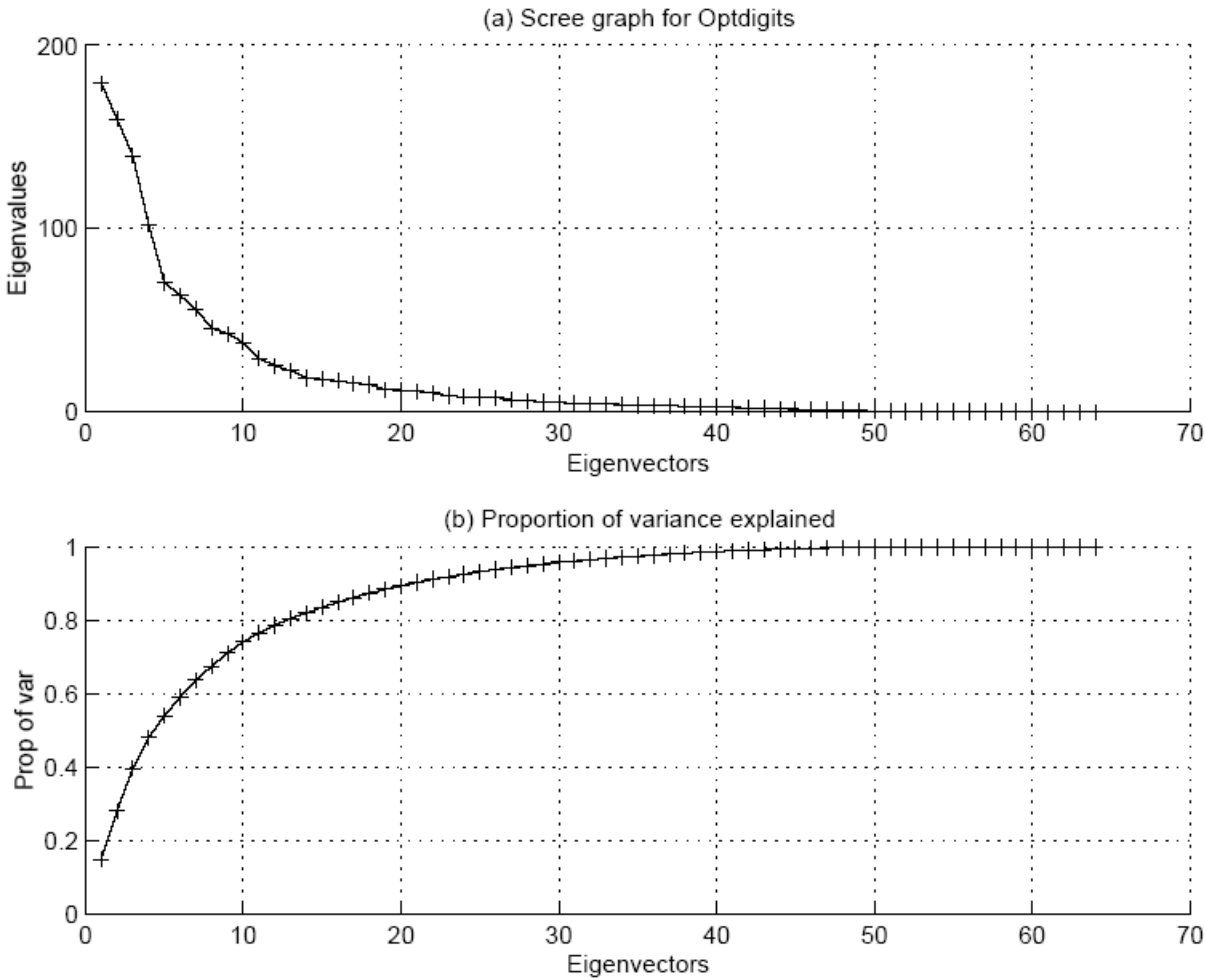


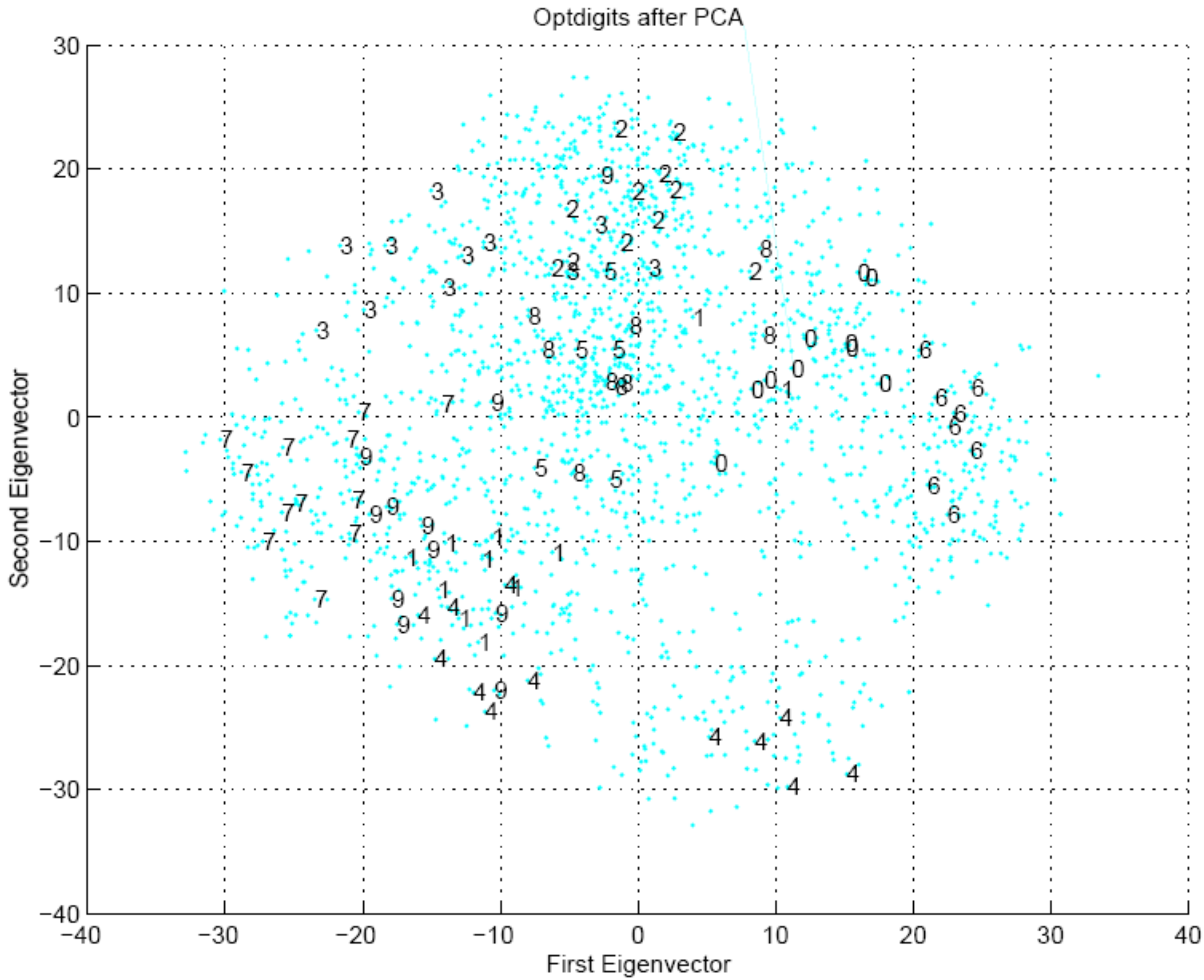
How to choose k ?

- 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
200
100
0
0
10
20
30
40
50
60
70
Eigenvectors
(b) Proportion of variance explained
Prop of var
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60
70
Eigenvectors


Optdigits after PCA
Second Eigenvector
First Eigenvector
30
20
10
0
−10
−20
−30
−40
−40
−30
−20
−10
0
10
20
30
40

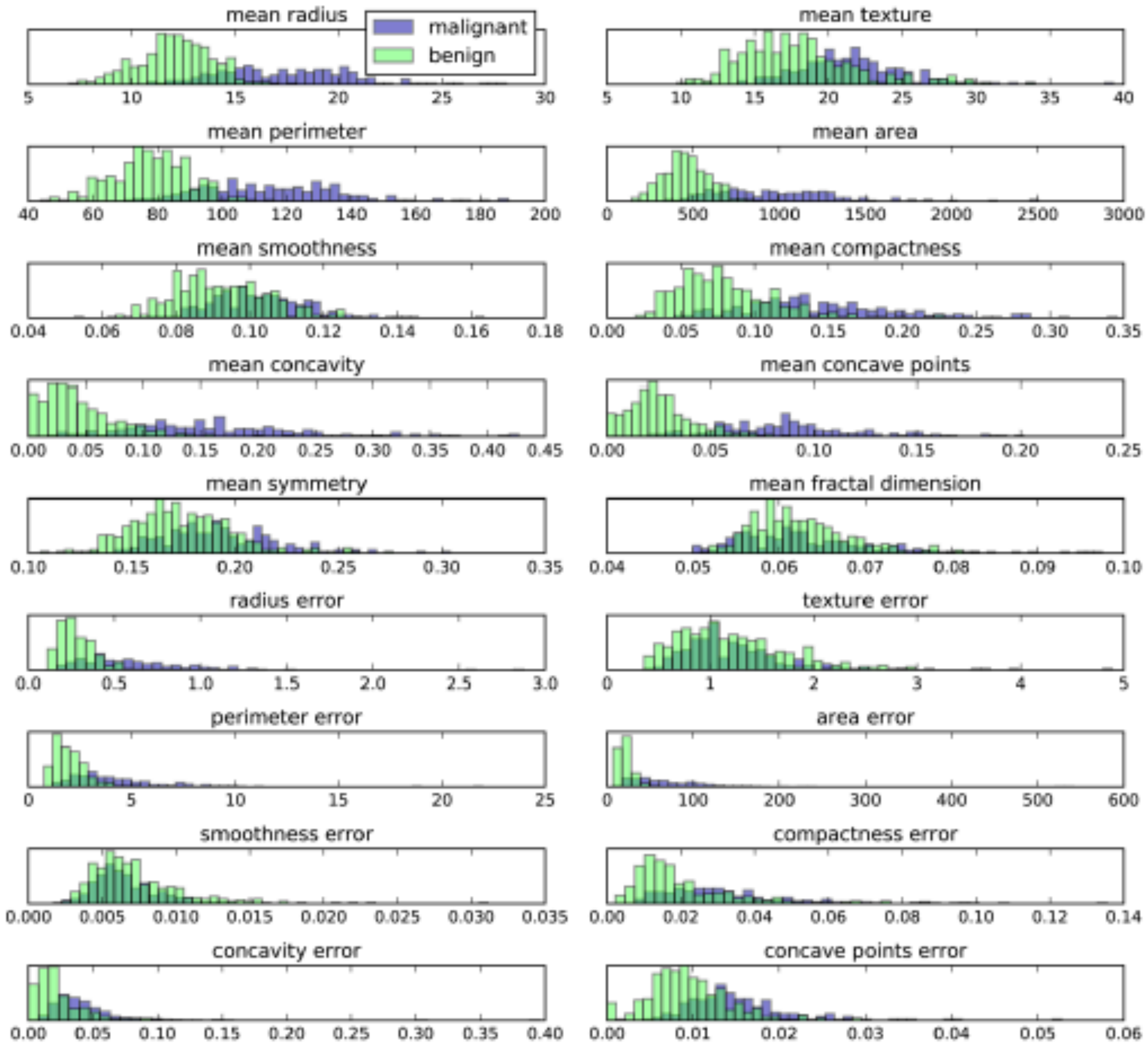
Feature Embedding

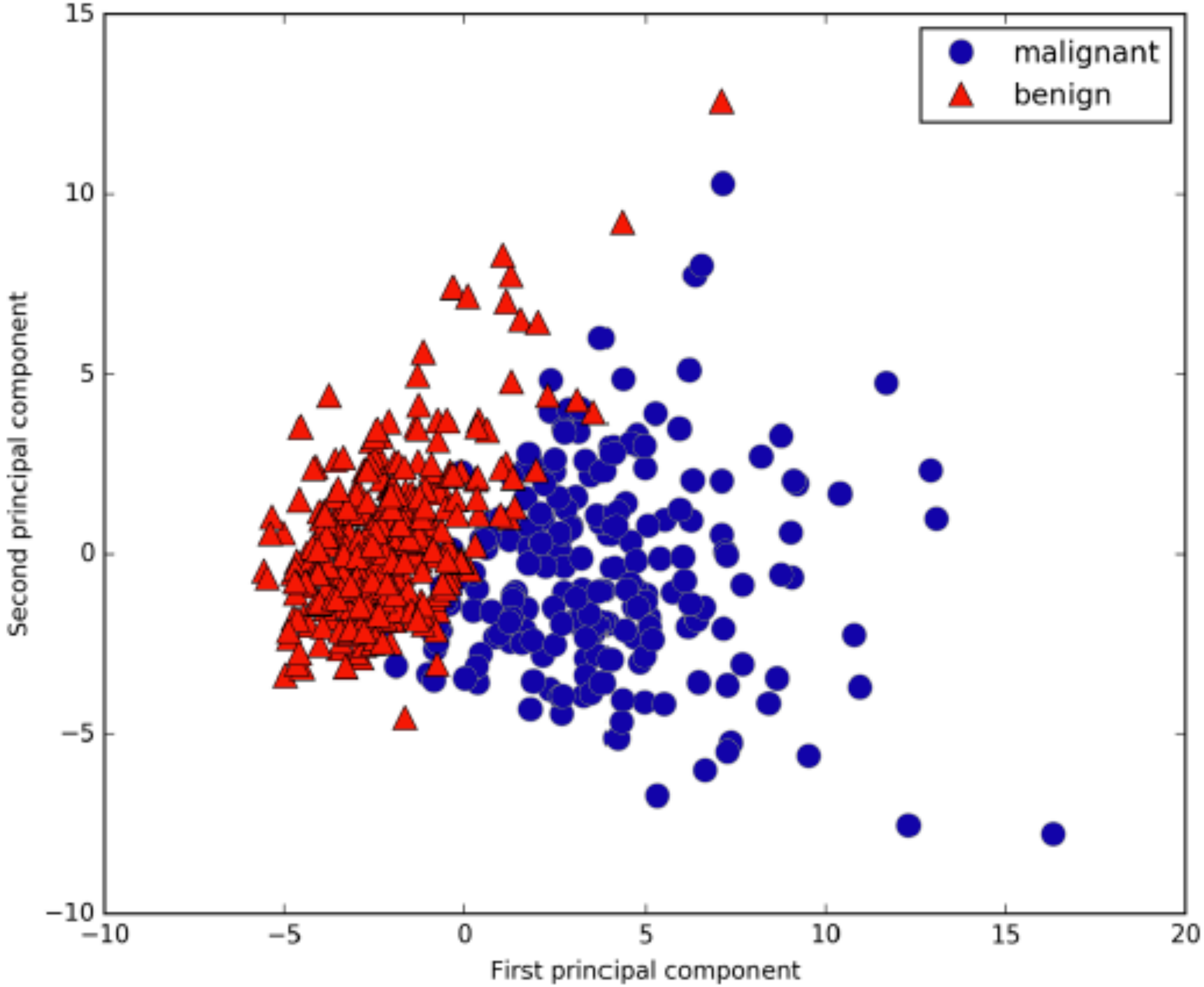
- When $\boldsymbol{X}$ is the Nxd data matrix,

$\boldsymbol{X}^T\boldsymbol{X}$ is the dxd matrix (covariance of features, if mean-centered)

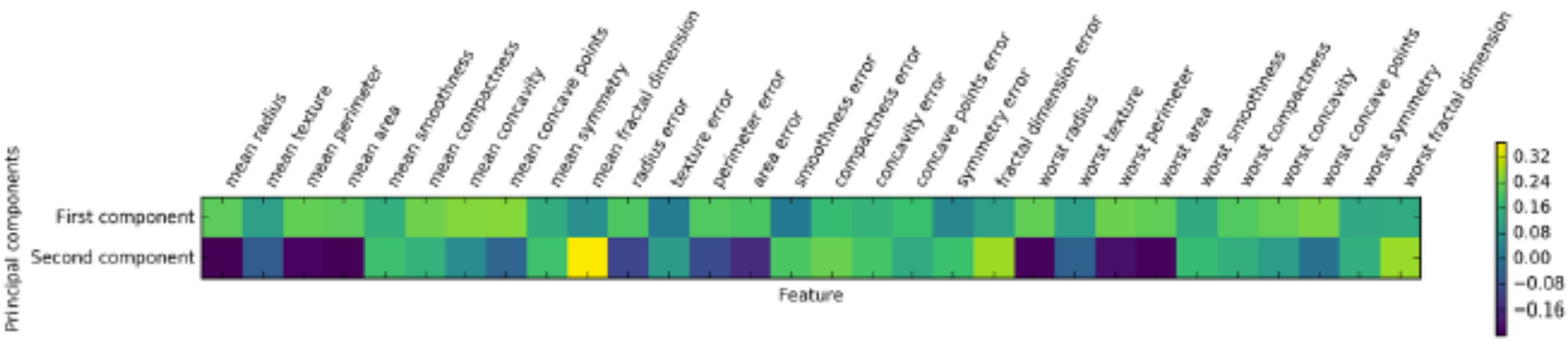
$\boldsymbol{X}\boldsymbol{X}^T$ is the NxN matrix (pairwise similarities of instances)

- PCA uses the eigenvectors of $\boldsymbol{X}^T\boldsymbol{X}$ which are d-dim and can be used for projection
- Feature embedding uses the eigenvectors of $\boldsymbol{X}\boldsymbol{X}^T$ which are N-dim and which give directly the coordinates after projection
- Sometimes, we can define pairwise similarities (or distances) between instances, then we can use feature embedding without needing to represent instances as vectors.


mean radius
malignant
benign
mean texture
mean perimeter
mean area
mean smoothness
mean compactness
mean concavity
mean concave points
mean symmetry
mean fractal dimension
radius error
texture error
perimeter error
area error
smoothness error
compactness error
concavity error
concave points error



malignant
benign
Second principal component
First principal component
15
10
5
0
−5
−10
−10
−5
0
5
10
15
20



mean radius
mean texture
mean perimeter
mean area
mean smoothness
mean compactness
mean concavity
mean concave points
mean symmetry
mean fractal dimension
radius error
texture error
perimeter error
area error
smoothness error
compactness error
concavity error
concave points error
symmetry error
fractal dimension error
worst radius
worst texture
worst perimeter
worst area
worst smoothness
worst compactness
worst concavity
worst concave points
worst symmetry
worst fractal dimension
Principal components
First component
Second component
Feature
0.32
0.24
0.16
0.08
0.00
−0.08
−0.16

Factor Analysis

- Find a small number of factors **z**, which when combined generate **x** :

$$x_i - \mu_i = v_{i1}z_1 + v_{i2}z_2 + \ldots + v_{ik}z_k + \varepsilon_i$$

where z_j, $j = 1,\ldots,k$ are the latent factors with

$E[\, z_j \,]=0$, $Var(z_j)=1$, $Cov(z_i, z_j)=0$, $i \neq j$,

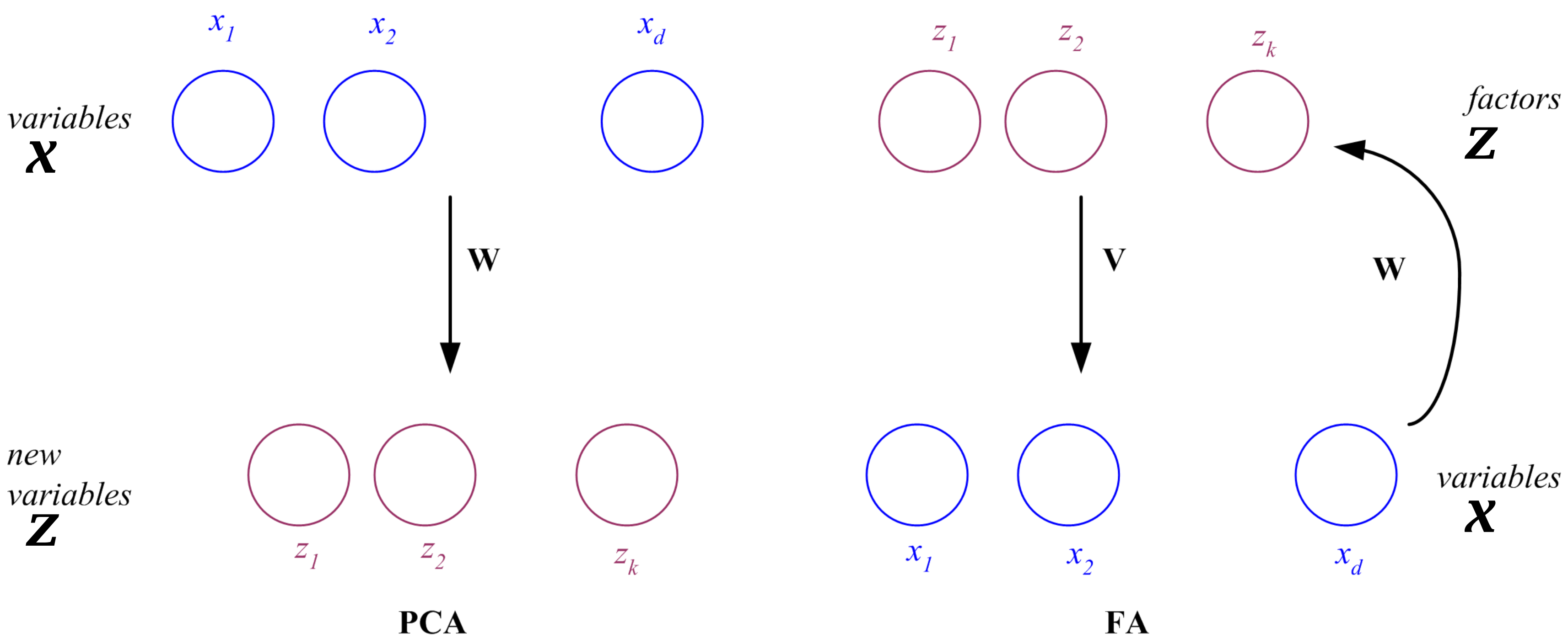
ε_i are the noise sources

$E[\, \varepsilon_i \,]= \psi_i$, $Cov(\varepsilon_i , \varepsilon_j) =0$, $i \neq j$, $Cov(\varepsilon_i , z_j) =0$,

and v_{ij} are the factor loadings

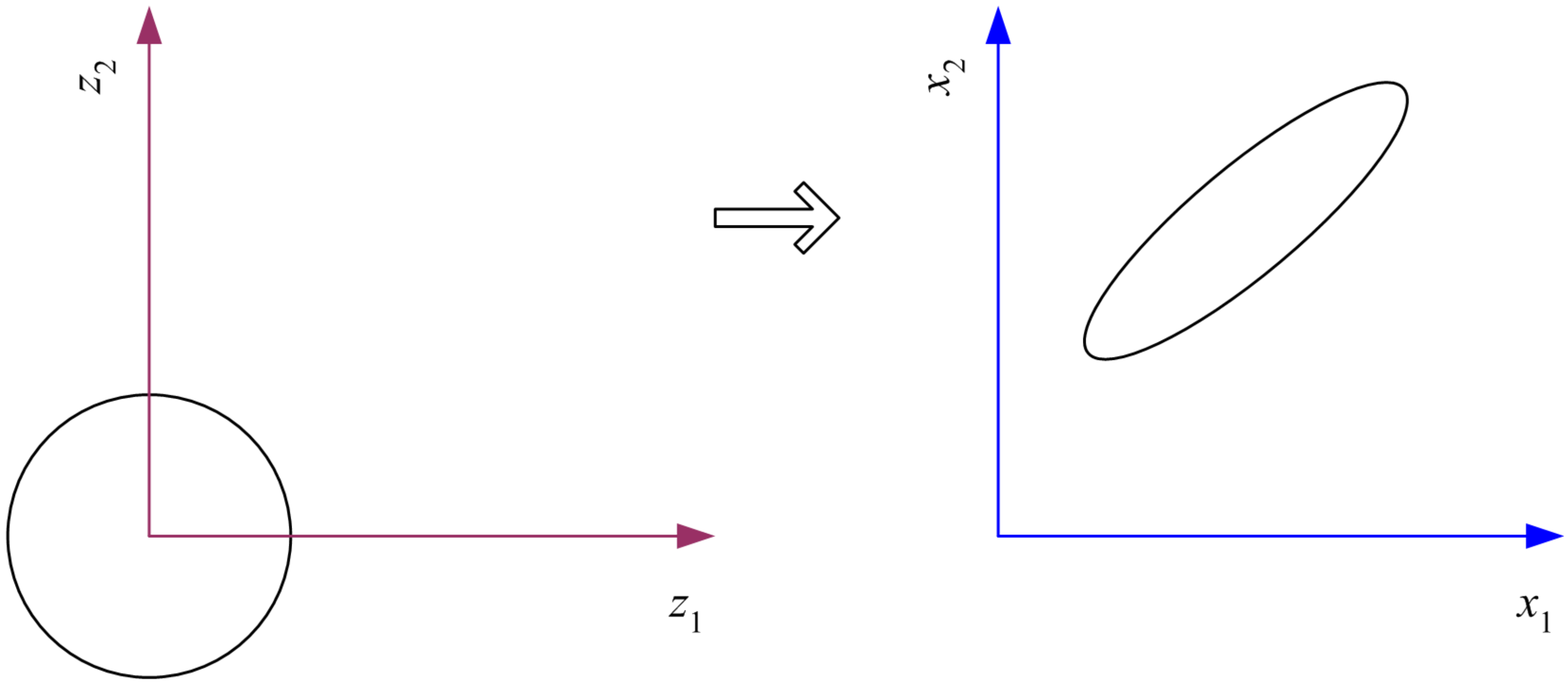
PCA vs FA

- PCA From **x** to ***z*** $\mathbf{z} = \mathbf{W}^T(\mathbf{x} - \boldsymbol{\mu})$
- FA From **z** to **x** $\mathbf{x} - \boldsymbol{\mu} = \mathbf{V}\mathbf{z} + \boldsymbol{\varepsilon}$



Factor Analysis

- In FA, factors z_i are stretched, rotated and translated to generate **x**



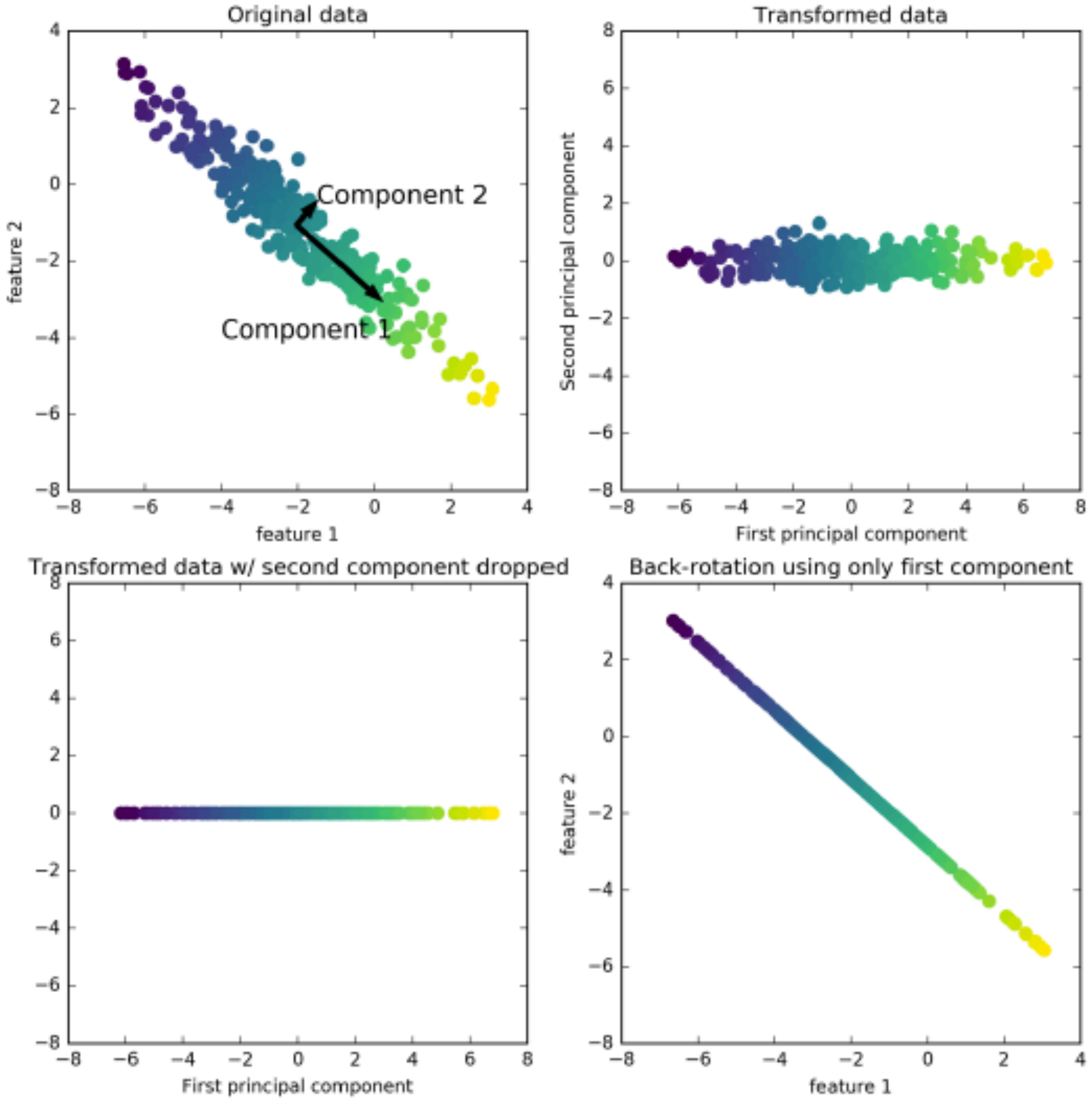
Singular Value Decomposition and Matrix Factorization

- Singular value decomposition: $\boldsymbol{X}=\boldsymbol{VAW}^T$

  $\boldsymbol{V}$ is NxN and contains the eigenvectors of $\boldsymbol{XX}^T$

  $\boldsymbol{W}$ is dxd and contains the eigenvectors of $\boldsymbol{X}^T\boldsymbol{X}$

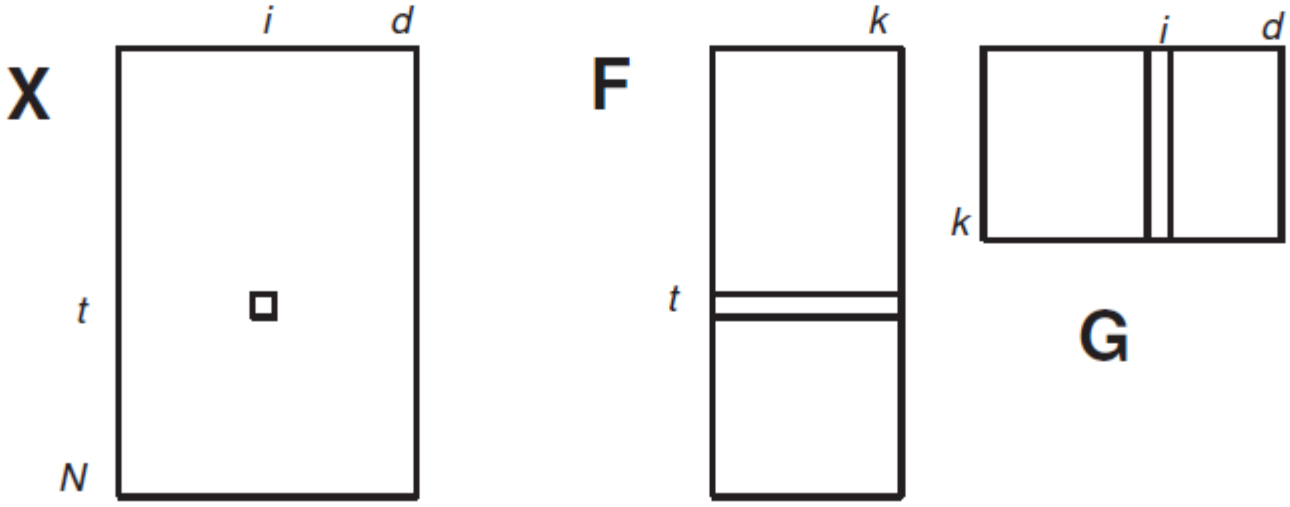
  and $\mathbf{A}$ is Nxd and contains singular values on its first k diagonal
- $\boldsymbol{X}=\boldsymbol{u}_1 a_1 \boldsymbol{v}_1^T+\ldots+\boldsymbol{u}_k a_k \boldsymbol{v}_k^T$ where k is the rank of $\boldsymbol{X}$


Original data
Component 2
Component 1
feature 1
feature 2
Transformed data
First principal component
Second principal component
Transformed data w/ second component dropped
First principal component
Back-rotation using only first component
feature 1
feature 2

Matrix Factorization

- Matrix factorization: ***X=FG***

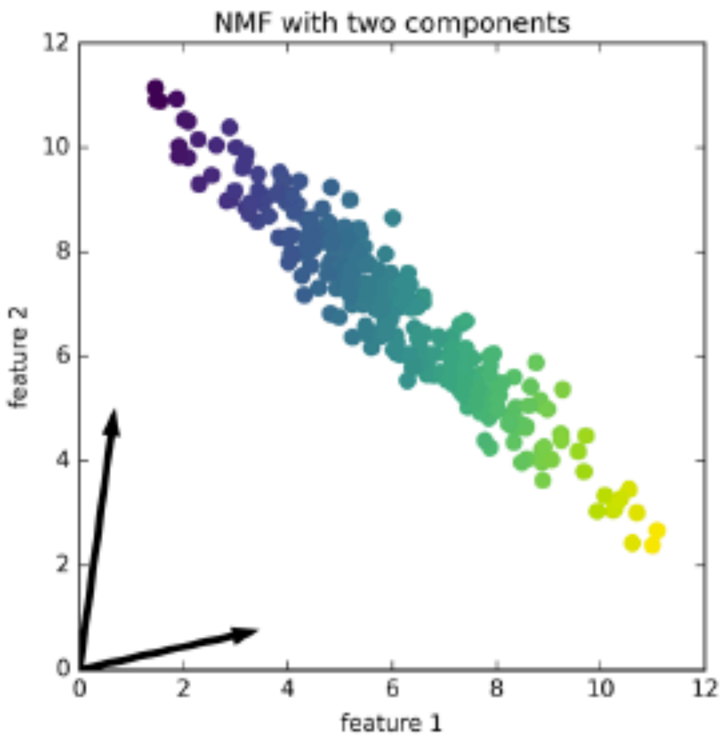
  F is *Nxk* and ***G*** is *kxd*

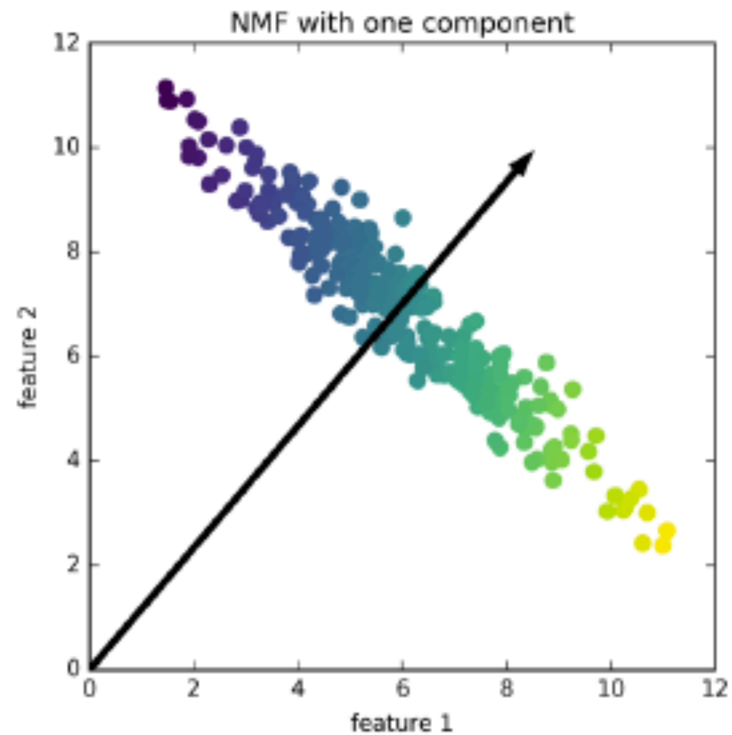


$$X_{ti} = F_t^T G_i = \sum_{j=1}^{k} F_{tj} G_{ji}$$

Latent semantic indexing

Non-negative Matrix Factorization





Multidimensional Scaling (MDS)

- Given pairwise distances between N points,

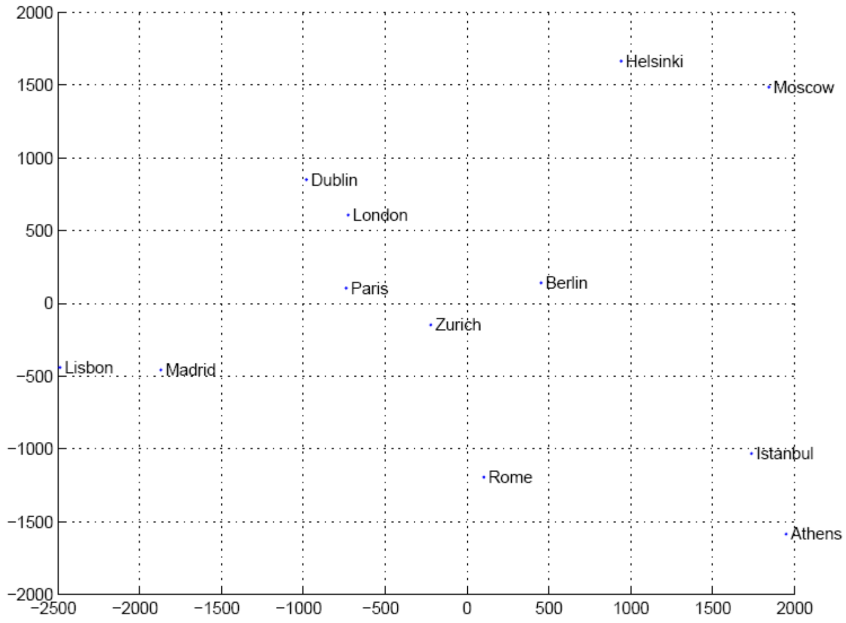
  d_{ij}, $i,j = 1,\ldots,N$

  place on a low-dim map s.t. distances are preserved (by feature embedding)
- $\mathbf{z} = \mathbf{g}(\mathbf{x} \mid \vartheta)$ Find ϑ that min Sammon stress

$$E(\theta \mid \mathcal{X}) = \sum_{r,s} \frac{\left(\|\mathbf{z}^r - \mathbf{z}^s\| - \|\mathbf{x}^r - \mathbf{x}^s\|\right)^2}{\|\mathbf{x}^r - \mathbf{x}^s\|^2}$$

$$= \sum_{r,s} \frac{\left(\|\mathbf{g}(\mathbf{x}^r \mid \theta) - \mathbf{g}(\mathbf{x}^s \mid \theta)\| - \|\mathbf{x}^r - \mathbf{x}^s\|\right)^2}{\|\mathbf{x}^r - \mathbf{x}^s\|^2}$$

Map of Europe by MDS

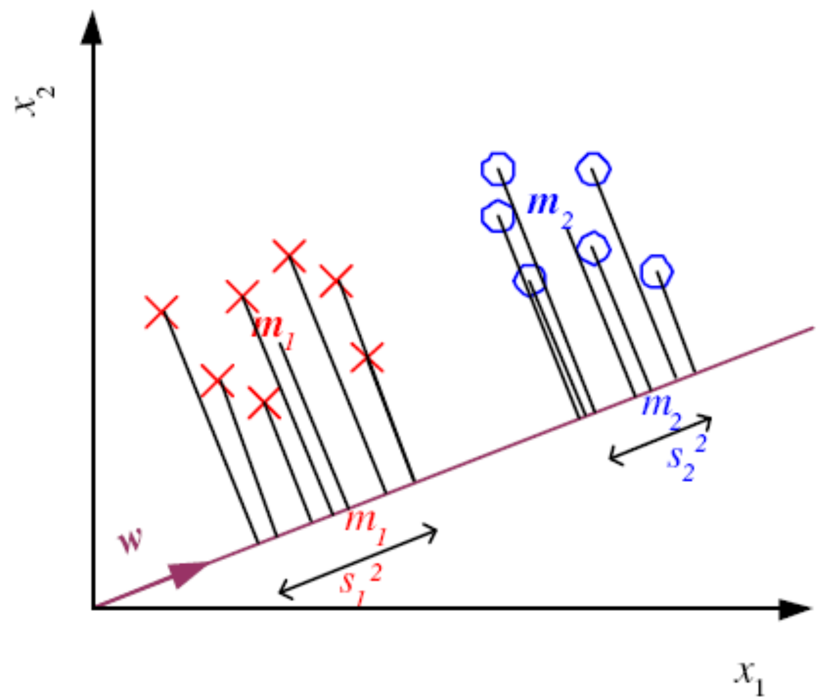


Map from CIA – The World Factbook: http://www.cia.gov/

Linear Discriminant Analysis (LDA)

- Find a low-dimensional space such that when **x** is projected, classes are well-separated.
- Find ***w*** that maximizes

$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$$



$$m_1 = \frac{\sum_t \mathbf{w}^T \mathbf{x}^t r^t}{\sum_t r^t} \qquad s_1^2 = \sum_t \left(\mathbf{w}^T \mathbf{x}^t - m_1\right)^2 r^t$$

- Between-class scatter:

$$\begin{aligned}(m_1 - m_2)^2 &= \left(\mathbf{w}^T\mathbf{m}_1 - \mathbf{w}^T\mathbf{m}_2\right)^2 \\ &= \mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T\mathbf{w} \\ &= \mathbf{w}^T\mathbf{S}_B\mathbf{w} \text{ where } \mathbf{S}_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T\end{aligned}$$

- Within-class scatter:

$$\begin{aligned}s_1^2 &= \sum_t \left(\mathbf{w}^T\mathbf{x}^t - m_1\right)^2 r^t \\ &= \sum_t \mathbf{w}^T\left(\mathbf{x}^t - \mathbf{m}_1\right)\left(\mathbf{x}^t - \mathbf{m}_1\right)^T\mathbf{w}r^t = \mathbf{w}^T\mathbf{S}_1\mathbf{w}\end{aligned}$$

$$\text{where } \mathbf{S}_1 = \sum_t \left(\mathbf{x}^t - \mathbf{m}_1\right)\left(\mathbf{x}^t - \mathbf{m}_1\right)^T r^t$$

$$s_1^2 + s_1^2 = \mathbf{w}^T\mathbf{S}_W\mathbf{w} \text{ where } \mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2$$

Fisher's Linear Discriminant (LDA)

- Find *w* that max

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} = \frac{\left|\mathbf{w}^T (\mathbf{m}_1 - \mathbf{m}_2)\right|^2}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

- LDA soln:

$$\mathbf{w} = c \cdot \mathbf{S}_W^{-1} (\mathbf{m}_1 - \mathbf{m}_2)$$

- Parametric soln:

$$\mathbf{w} = \Sigma^{-1} (\mu_1 - \mu_2)$$

when $p(\mathbf{x} \mid C_i) \sim \mathcal{N}(\mu_i, \Sigma)$

K>2 Classes

- Within-class scatter:

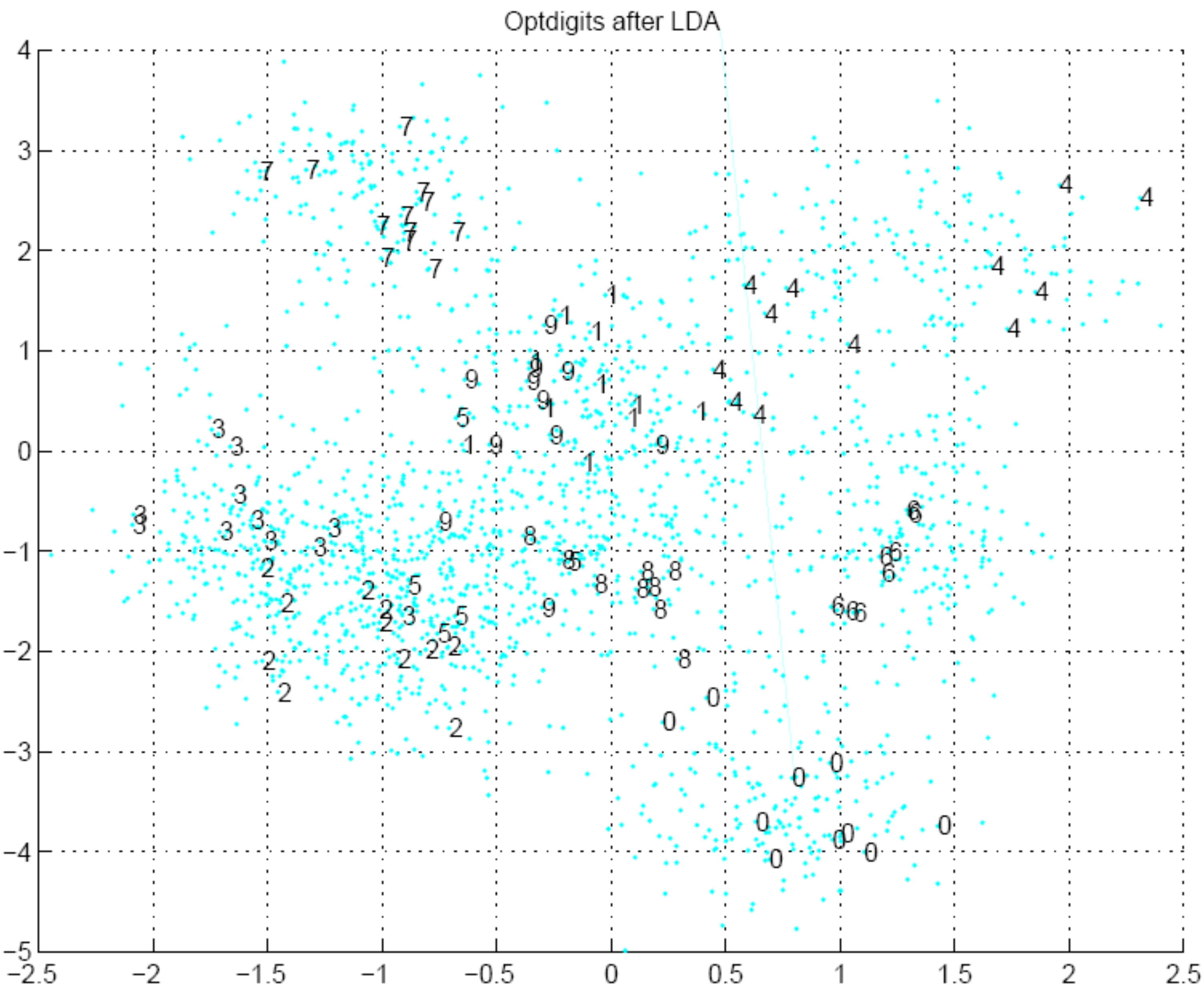
$$\mathbf{S}_W = \sum_{i=1}^{K} \mathbf{S}_i \qquad \mathbf{S}_i = \sum_t r_i^t \left(\mathbf{x}^t - \mathbf{m}_i\right)\left(\mathbf{x}^t - \mathbf{m}_i\right)^T$$

- Between-class scatter:

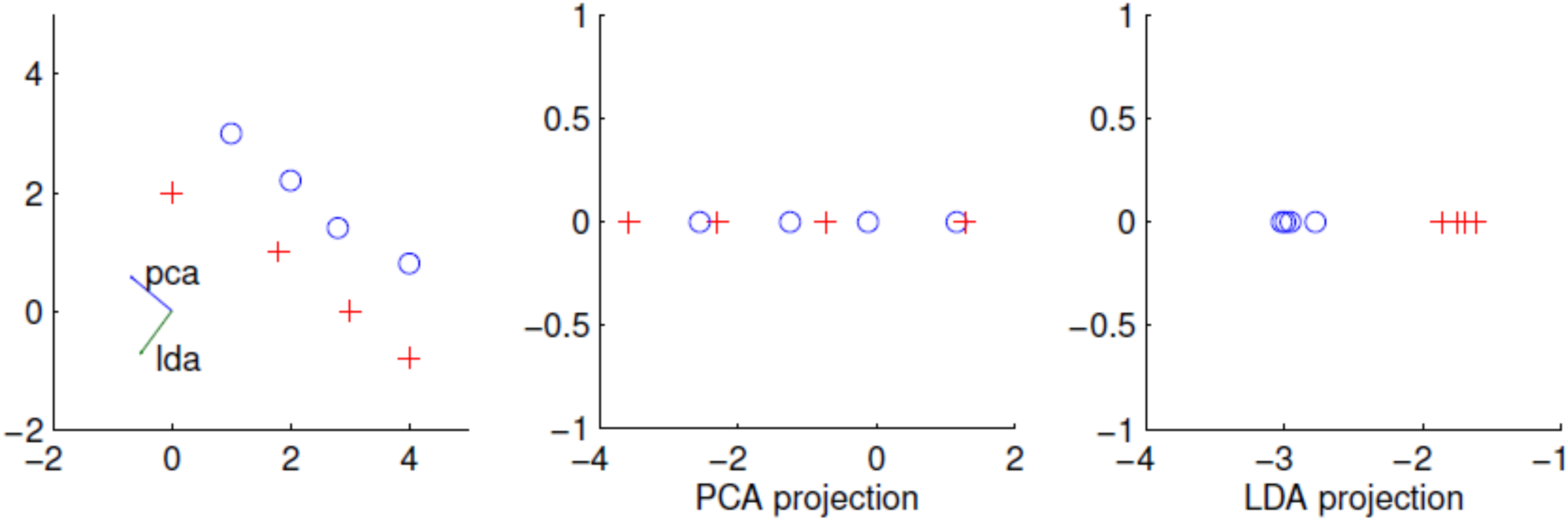
$$\mathbf{S}_B = \sum_{i=1}^{K} N_i \left(\mathbf{m}_i - \mathbf{m}\right)\left(\mathbf{m}_i - \mathbf{m}\right)^T \qquad \mathbf{m} = \frac{1}{K}\sum_{i=1}^{K} \mathbf{m}_i$$

- Find **W** that max $J(\mathbf{W}) = \dfrac{\left|\mathbf{W}^T \mathbf{S}_B \mathbf{W}\right|}{\left|\mathbf{W}^T \mathbf{S}_W \mathbf{W}\right|}$

The largest eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$; maximum rank of K-1


Optdigits after LDA
4
3
2
1
0
−1
−2
−3
−4
−5
−2.5
−2
−1.5
−1
−0.5
0
0.5
1
1.5
2
2.5

PCA vs LDA



4
2
0
−2
−2
0
2
4
pca
lda
1
0.5
0
−0.5
−1
−4
−2
0
2
PCA projection
1
0.5
0
−0.5
−1
−4
−3
−2
−1
LDA projection

Canonical Correlation Analysis

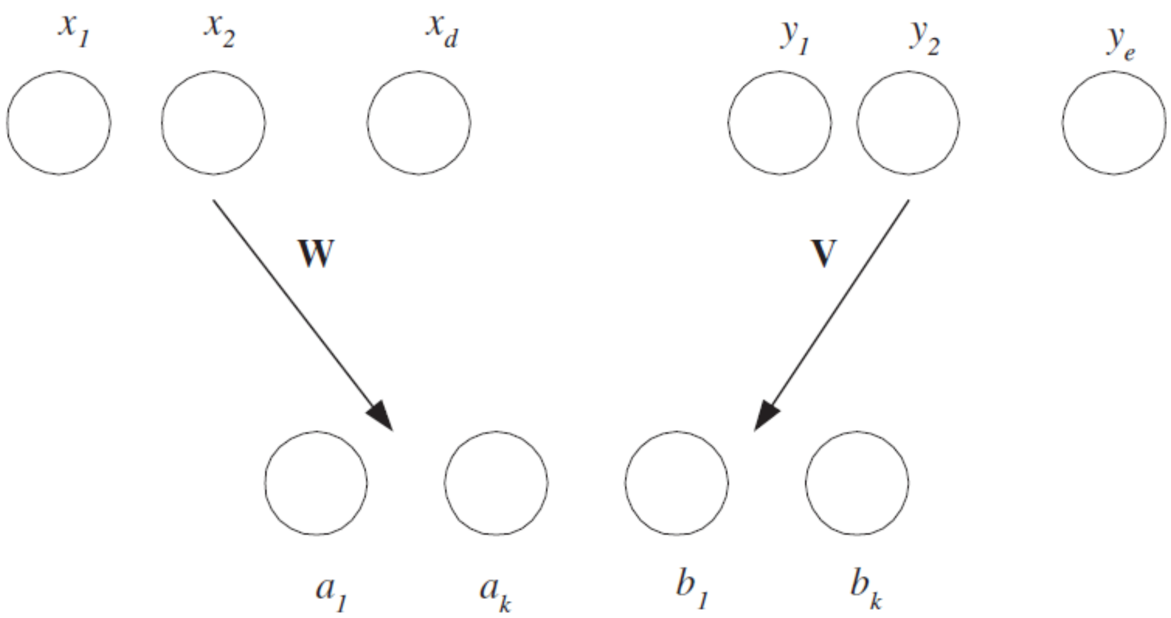
- $\boldsymbol{X}=\{\boldsymbol{x}^t,\boldsymbol{y}^t\}_t$; two sets of variables **x** and ***y*** x
- We want to find two projections ***w*** and **v** st when ***x*** is projected along ***w*** and ***y*** is projected along **v**, the correlation is maximized:

$$\rho = \text{Corr}(\boldsymbol{w}^T\boldsymbol{x},\boldsymbol{v}^T\boldsymbol{y}) = \frac{\text{Cov}(\boldsymbol{w}^T\boldsymbol{x},\boldsymbol{v}^T\boldsymbol{y})}{\sqrt{\text{Var}(\boldsymbol{w}^T\boldsymbol{x})}\sqrt{\text{Var}(\boldsymbol{v}^T\boldsymbol{y})}}$$

$$= \frac{\boldsymbol{w}^T\text{Cov}(\boldsymbol{x},\boldsymbol{y})\boldsymbol{v}}{\sqrt{\boldsymbol{w}^T\text{Var}(\boldsymbol{x})\boldsymbol{w}}\sqrt{\boldsymbol{v}^T\text{Var}(\boldsymbol{y})\boldsymbol{v}}} = \frac{\boldsymbol{w}^T\mathbf{S}_{xy}\boldsymbol{v}}{\sqrt{\boldsymbol{w}^T\mathbf{S}_{xx}\boldsymbol{w}}\sqrt{\boldsymbol{v}^T\mathbf{S}_{yy}\boldsymbol{v}}}$$

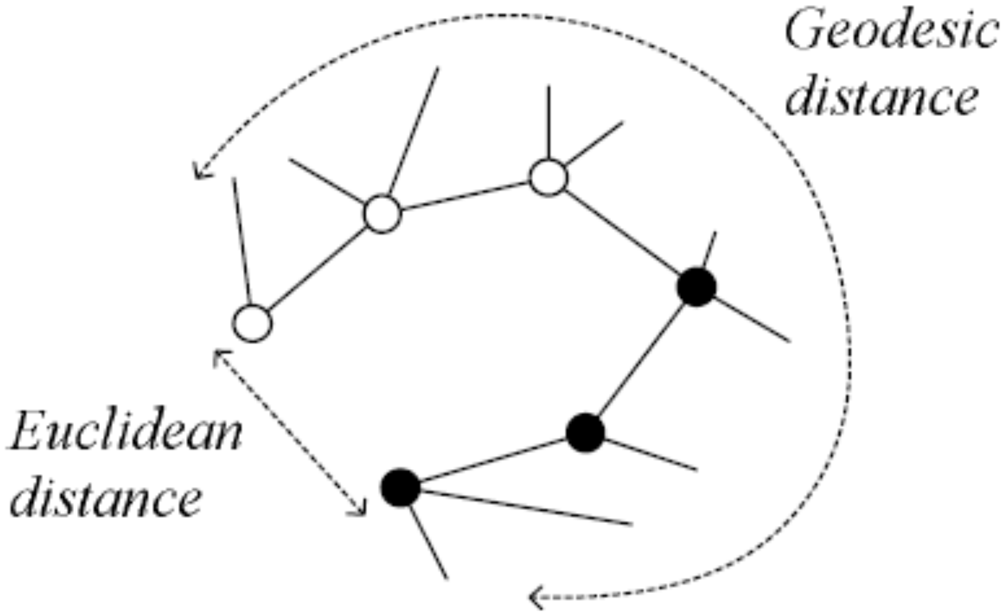
CCA

- ***x*** and ***y*** may be two different views or modalities; e.g., image and word tags, and CCA does a joint mapping



Isomap

- Geodesic distance is the distance along the manifold that the data lies in, as opposed to the Euclidean distance in the input space

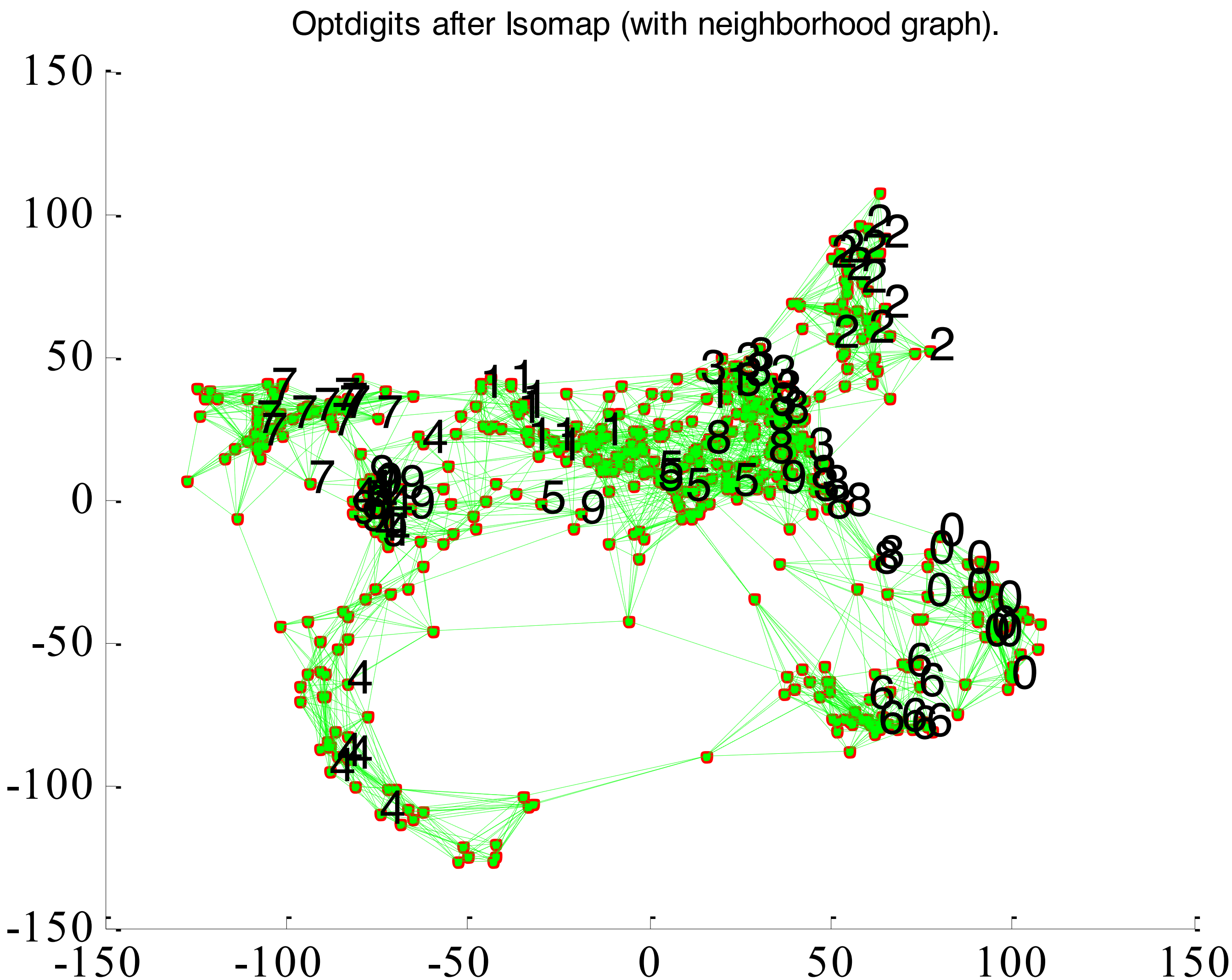


Isomap

- Instances r and s are connected in the graph if

  $||\boldsymbol{x}^r-\boldsymbol{x}^s||<\varepsilon$ or if $\boldsymbol{x}^s$ is one of the k neighbors of $\boldsymbol{x}^r$

  The edge length is $||\boldsymbol{x}^r-\boldsymbol{x}^s||$
- For two nodes r and s not connected, the distance is equal to the shortest path between them
- Once the *NxN* distance matrix is thus formed, use MDS to find a lower-dimensional mapping



Matlab source from http://web.mit.edu/cocosci/isomap/isomap.html

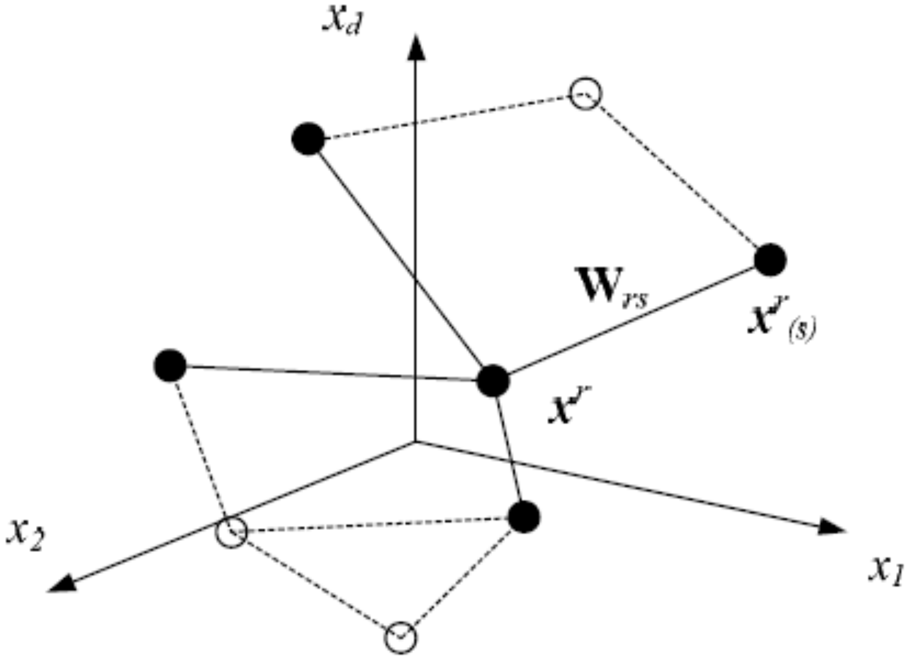
Locally Linear Embedding

1. Given $\boldsymbol{x}^r$ find its neighbors $\boldsymbol{x}^s_{(r)}$
2. Find $\mathbf{W}_{rs}$ that minimize

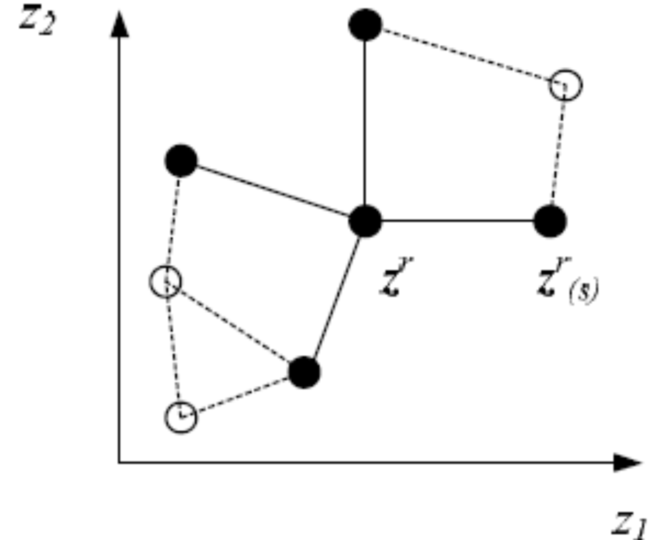
$$E(\mathbf{W} \mid X) = \sum_r \left\| \mathbf{x}^r - \sum_s \mathbf{W}_{rs} \mathbf{x}^s_{(r)} \right\|^2$$

3. Find the new coordinates $\mathbf{z}^r$ that minimize

$$E(\mathbf{z} \mid \mathbf{W}) = \sum_r \left\| z^r - \sum_s \mathbf{W}_{rs} z^s_{(r)} \right\|^2$$

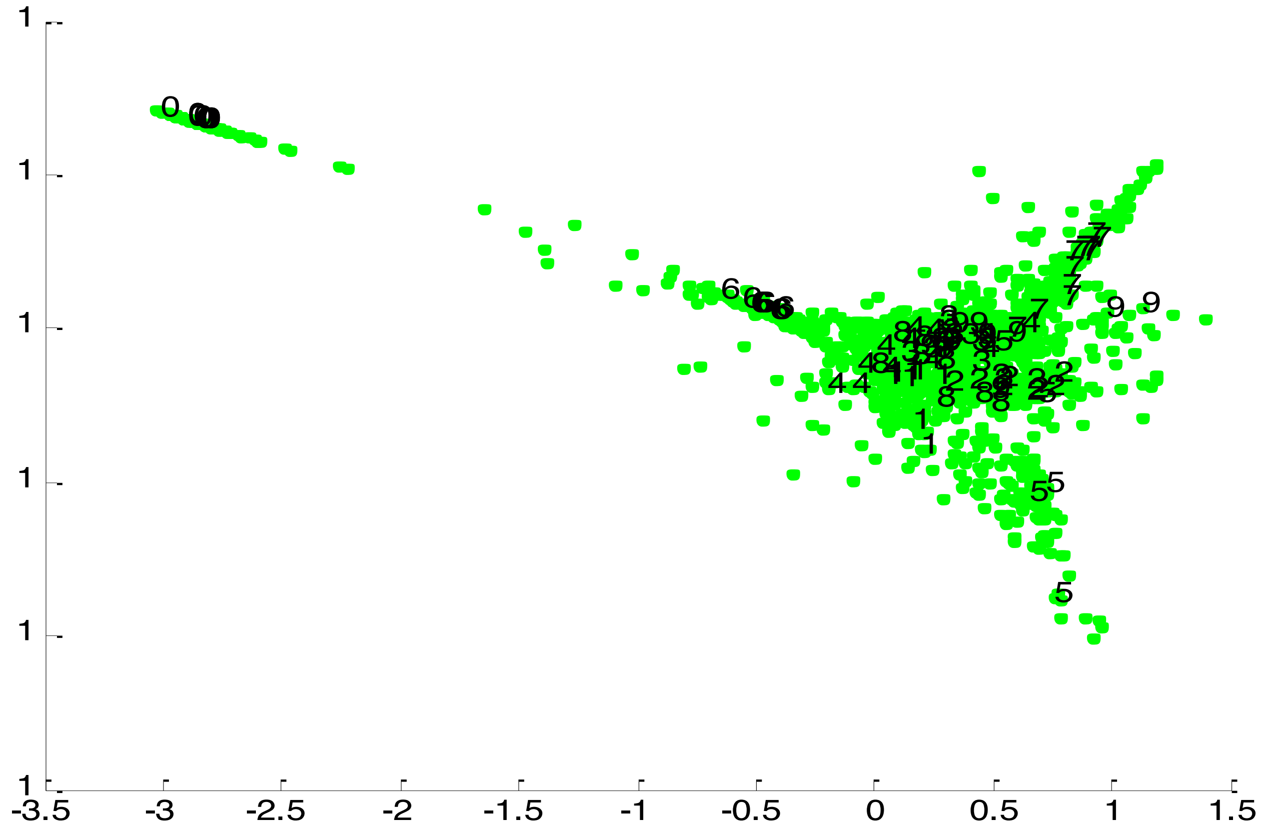


x space



z space

LLE on Optdigits



Matlab source from http://www.cs.toronto.edu/~roweis/lle/code.html

Laplacian Eigenmaps

- Let r and s be two instances and B_{rs} is their similarity, we want to find $\mathbf{z}^r$ and $\mathbf{z}^s$ that

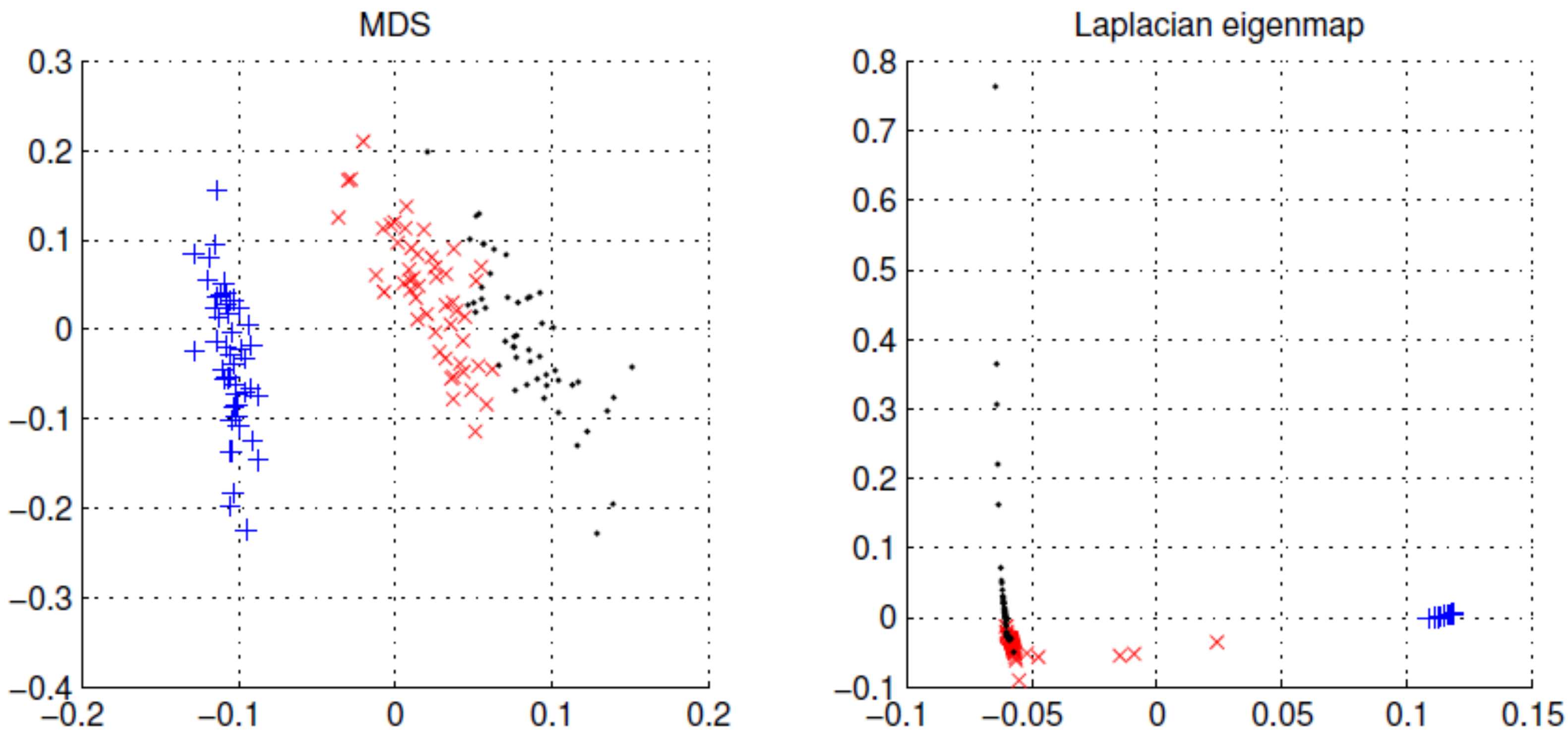
$$\min \sum_{r,s} \|\mathbf{z}^r - \mathbf{z}^s\|^2 B_{rs}$$

- B_{rs} can be defined in terms of similarity in an original space: 0 if $\mathbf{x}^r$ and $\mathbf{x}^s$ are too far, otherwise

$$B_{rs} = \exp\left[-\frac{\|\mathbf{x}^r - \mathbf{x}^s\|^2}{2\sigma^2}\right]$$

- Defines a graph Laplacian, and feature embedding returns $\mathbf{z}^r$

Laplacian Eigenmaps on Iris



Spectral clustering (chapter 7)

CHAPTER 10:

LINEAR DISCRIMINATION

Likelihood- vs. Discriminant-based Classification

- Likelihood-based: Assume a model for $p(\mathbf{x}|C_i)$, use Bayes' rule to calculate $P(C_i|\mathbf{x})$

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

- Discriminant-based: Assume a model for $g_i(\mathbf{x}|\Phi_i)$; no density estimation
- Estimating the boundaries is enough; no need to accurately estimate the densities inside the boundaries

Linear Discriminant

- Linear discriminant:

$$g_i\left(\mathbf{x} \mid \mathbf{w}_i, w_{i0}\right) = \mathbf{w}_i^T \mathbf{x} + w_{i0} = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$$

- Advantages:
  - Simple: O(d) space/computation
  - Knowledge extraction: Weighted sum of attributes; positive/negative weights, magnitudes (credit scoring)
  - Optimal when $p(\mathbf{x}|C_i)$ are Gaussian with shared cov matrix; useful when classes are (almost) linearly separable

Generalized Linear Model

- Quadratic discriminant:

$$g_i\left(\mathbf{x} \mid \mathbf{W}_i, \mathbf{w}_i, w_{i0}\right) = \mathbf{x}^T \mathbf{W}_i \mathbf{x} + \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

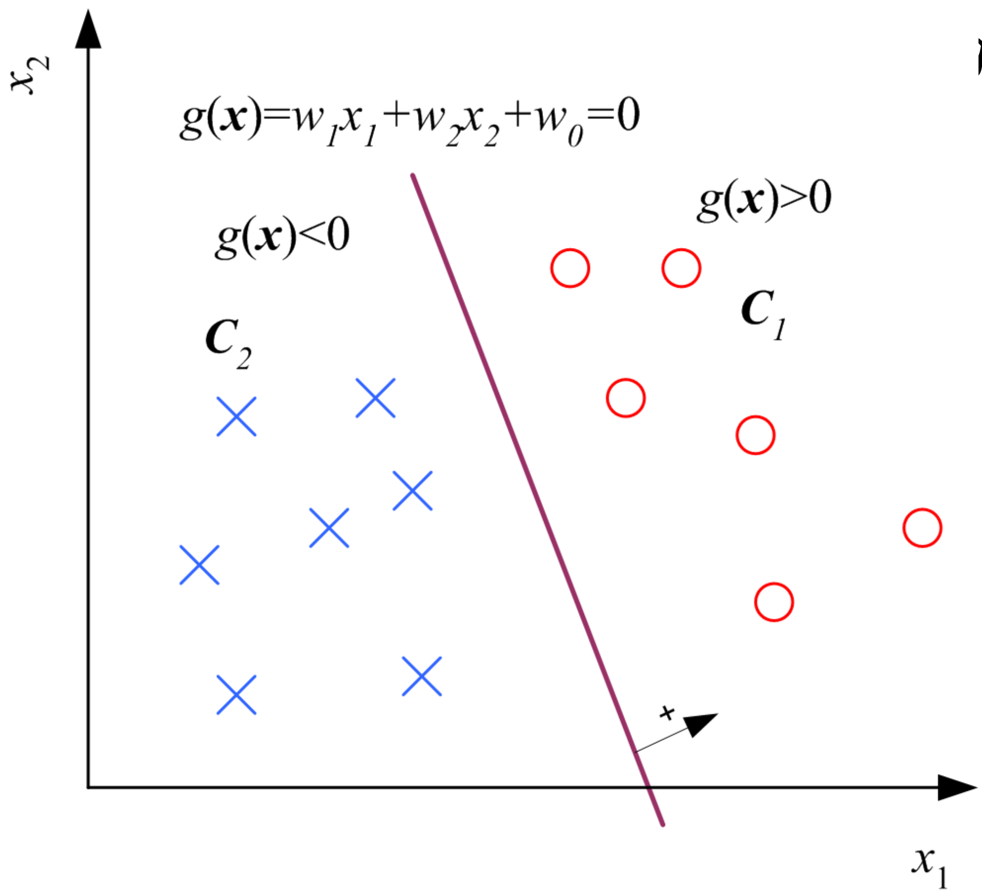
- Higher-order (product) terms:

$$z_1 = x_1,\ z_2 = x_2,\ z_3 = x_1^2,\ z_4 = x_2^2,\ z_5 = x_1 x_2$$

Map from **x** to **z** using nonlinear basis functions and use a linear discriminant in **z**-space

$$g_i(\mathbf{x}) = \sum_{j=1}^{k} w_{ij} \phi_j(\mathbf{x})$$

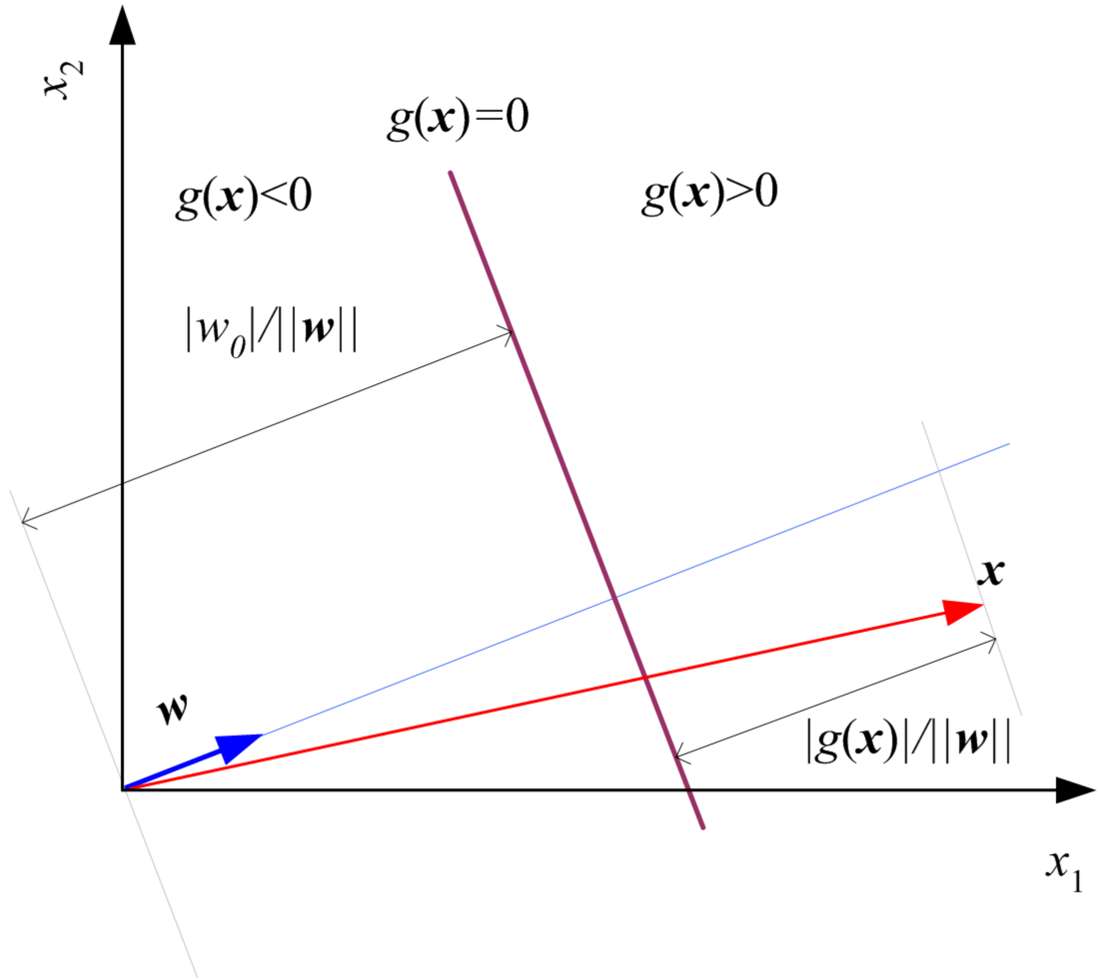
Two Classes



$$
\begin{aligned}
g(\mathbf{x}) &= g_1(\mathbf{x}) - g_2(\mathbf{x}) \\
&= \left(\mathbf{w}_1^T\mathbf{x} + w_{10}\right) - \left(\mathbf{w}_2^T\mathbf{x} + w_{20}\right) \\
&= \left(\mathbf{w}_1 - \mathbf{w}_2\right)^T\mathbf{x} + \left(w_{10} - w_{20}\right) \\
&= \mathbf{w}^T\mathbf{x} + w_0
\end{aligned}
$$

$$
\text{choose} \begin{cases} C_1 & \text{if } g(\mathbf{x}) > 0 \\ C_2 & \text{otherwise} \end{cases}
$$

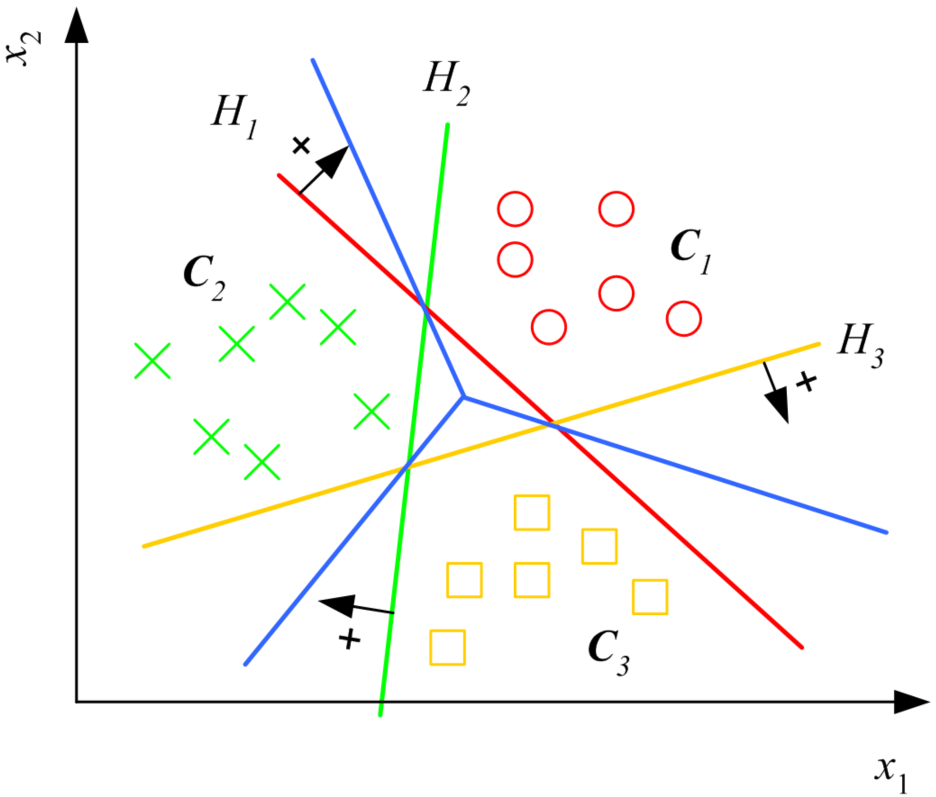
Geometry



x_2
$g(\boldsymbol{x})=0$
$g(\boldsymbol{x})<0$
$g(\boldsymbol{x})>0$
$|w_0|/||\boldsymbol{w}||$
$\boldsymbol{x}$
$\boldsymbol{w}$
$|g(\boldsymbol{x})|/||\boldsymbol{w}||$
x_1

Multiple Classes

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

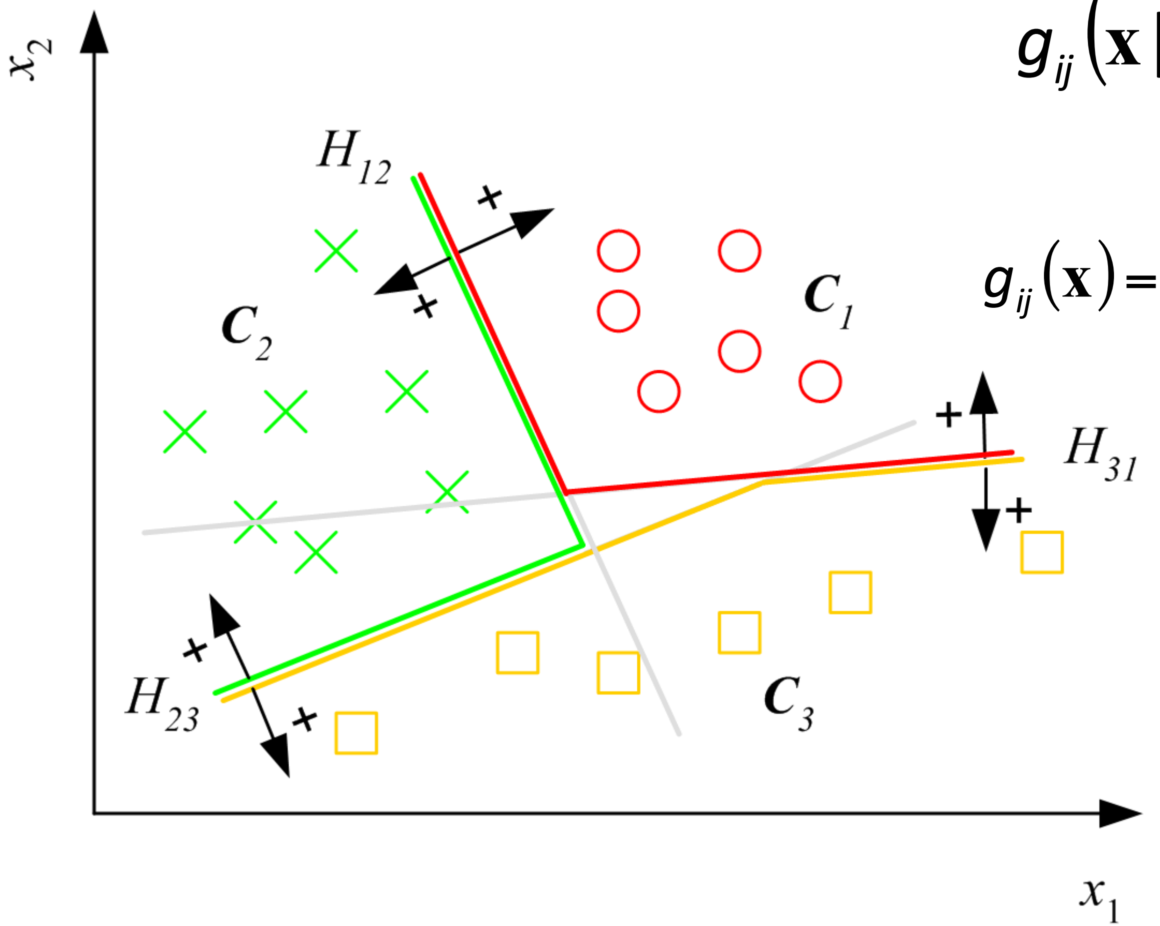


Choose C_i if

$$g_i(\mathbf{x}) = \max_{j=1}^{K} g_j(\mathbf{x})$$

Classes are
linearly separable

Pairwise Separation



$$g_{ij}\left(\mathbf{x}\,|\,\mathbf{w}_{ij}, w_{ij0}\right) = \mathbf{w}_{ij}^T\mathbf{x} + w_{ij0}$$

$$g_{ij}(\mathbf{x}) = \begin{cases} > 0 & \text{if } \mathbf{x} \in C_i \\ \leq 0 & \text{if } \mathbf{x} \in C_j \\ \text{don't care} & \text{otherwise} \end{cases}$$

choose C_i if

$$\forall j \neq i, g_{ij}(\mathbf{x}) > 0$$

From Discriminants to Posteriors

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

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

$$\mathbf{w}_i = \Sigma^{-1}\mu_i \quad w_{i0} = -\frac{1}{2}\mu_i^T \Sigma^{-1}\mu_i + \log P(C_i)$$

$$y \equiv P(C_1 \mid \mathbf{x}) \text{ and } P(C_2 \mid \mathbf{x}) = 1 - y$$

$$\text{choose } C_1 \text{ if} \begin{cases} y > 0.5 \\ y/(1-y) > 1 \\ \log[y/(1-y)] > 0 \end{cases} \text{ and } C_2 \text{ otherwise}$$

$$\operatorname{logit}(P(C_1 \mid \mathbf{x})) = \log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \log \frac{P(C_1 \mid \mathbf{x})}{P(C_2 \mid \mathbf{x})}$$

$$= \log \frac{p(\mathbf{x} \mid C_1)}{p(\mathbf{x} \mid C_2)} + \log \frac{P(C_1)}{P(C_2)}$$

$$= \log \frac{(2\pi)^{-d/2} |\Sigma|^{-1/2} \exp\left[-(1/2)(\mathbf{x} - \mu_1)^T \Sigma^{-1} (\mathbf{x} - \mu_1)\right]}{(2\pi)^{-d/2} |\Sigma|^{-1/2} \exp\left[-(1/2)(\mathbf{x} - \mu_2)^T \Sigma^{-1} (\mathbf{x} - \mu_2)\right]} + \log \frac{P(C_1)}{P(C_2)}$$

$$= \mathbf{w}^T \mathbf{x} + w_0$$

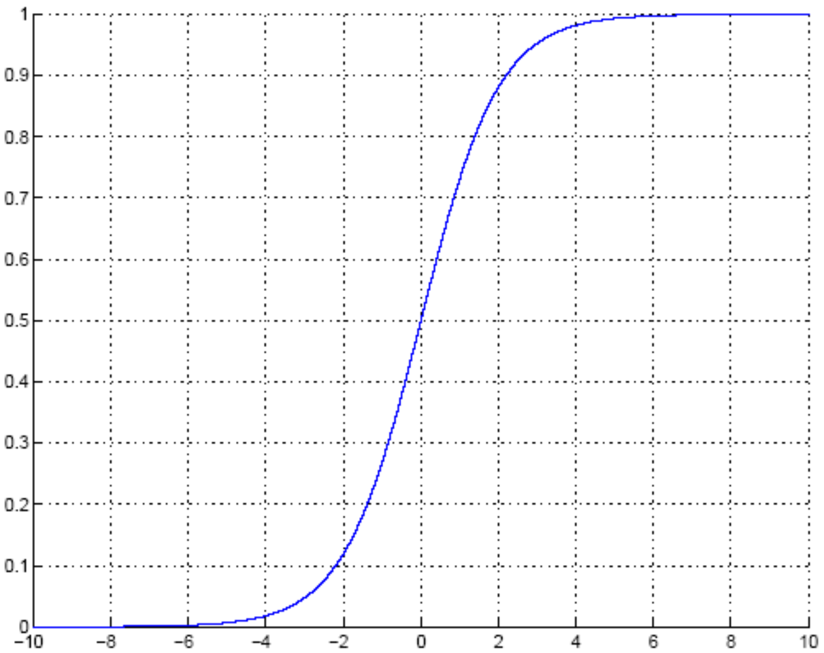
where $\mathbf{w} = \Sigma^{-1}(\mu_1 - \mu_2) \quad w_0 = -\frac{1}{2}(\mu_1 + \mu_2)^T \Sigma^{-1} (\mu_1 - \mu_2)$

The inverse of logit

$$\log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \mathbf{w}^T \mathbf{x} + w_0$$

$$P(C_1 \mid \mathbf{x}) = \operatorname{sigmoid}(\mathbf{w}^T \mathbf{x} + w_0) = \frac{1}{1 + \exp\left[-(\mathbf{w}^T \mathbf{x} + w_0)\right]}$$

Sigmoid (Logistic) Function



Calculate $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ and choose C_1 if $g(\mathbf{x}) > 0$, or

Calculate $y = \text{sigmoid}\left(\mathbf{w}^T\mathbf{x} + w_0\right)$ and choose C_1 if $y > 0.5$

Gradient-Descent

- E(**w**|X) is error with parameters **w** on sample X

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} E(\mathbf{w} \mid X)$$

- Gradient

$$\nabla_w E = \left[\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots, \frac{\partial E}{\partial w_d} \right]^T$$

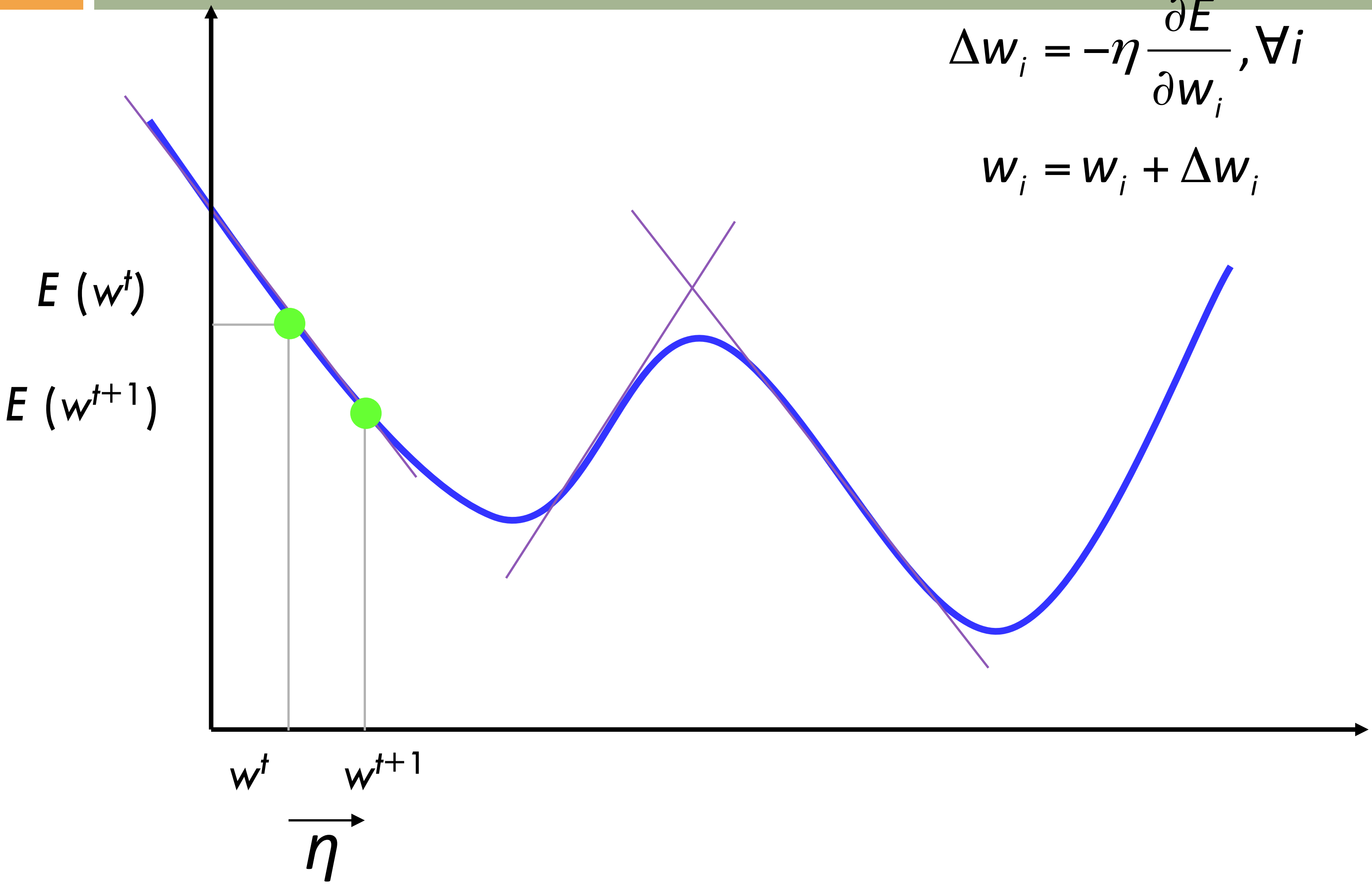
- Gradient-descent:

  Starts from random **w** and updates **w** iteratively in the negative direction of gradient

Gradient-Descent

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \forall i$$

$$w_i = w_i + \Delta w_i$$



Logistic Discrimination

Two classes: Assume log likelihood ratio is linear

$$\log \frac{p(\mathbf{x} | C_1)}{p(\mathbf{x} | C_2)} = \mathbf{w}^T \mathbf{x} + w_0^o$$

$$\operatorname{logit}(P(C_1 | \mathbf{x})) = \log \frac{P(C_1 | \mathbf{x})}{1 - P(C_1 | \mathbf{x})} = \log \frac{p(\mathbf{x} | C_1)}{p(\mathbf{x} | C_2)} + \log \frac{P(C_1)}{P(C_2)}$$

$$= \mathbf{w}^T \mathbf{x} + w_0$$

$$\text{where } w_0 = w_0^o + \log \frac{P(C_1)}{P(C_2)}$$

$$y = \hat{P}(C_1 | \mathbf{x}) = \frac{1}{1 + \exp\left[-\left(\mathbf{w}^T \mathbf{x} + w_0\right)\right]}$$

Training: Two Classes

$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_t \quad r^t \mid \mathbf{x}^t \sim \text{Bernoulli}(y^t)$$

$$y = P(C_1 \mid \mathbf{x}) = \frac{1}{1 + \exp[-(\mathbf{w}^T\mathbf{x} + w_0)]}$$

$$l(\mathbf{w}, w_0 \mid \mathcal{X}) = \prod_t (y^t)^{(r^t)} (1 - y^t)^{(1-r^t)}$$

$$E = -\log l$$

$$E(\mathbf{w}, w_0 \mid \mathcal{X}) = -\sum_t r^t \log y^t + (1 - r^t) \log (1 - y^t)$$

Training: Gradient-Descent

$$E(\mathbf{w}, w_0 \mid \mathcal{X}) = -\sum_t r^t \log y^t + \left(1 - r^t\right) \log\left(1 - y^t\right)$$

$$\text{If } y = \text{sigmoid}(a) \quad \frac{dy}{da} = y(1-y)$$

$$\Delta w_j = -\eta \frac{\partial E}{\partial w_j} = \eta \sum_t \left(\frac{r^t}{y^t} - \frac{1-r^t}{1-y^t}\right) y^t \left(1 - y^t\right) x_j^t$$

$$= \eta \sum_t \left(r^t - y^t\right) x_j^t, \; j = 1, \ldots, d$$

$$\Delta w_0 = -\eta \frac{\partial E}{\partial w_0} = \eta \sum_t \left(r^t - y^t\right)$$

For $j = 0, \ldots, d$

  $w_j \leftarrow$rand(-0.01,0.01)

Repeat

  For $j = 0, \ldots, d$

    $\Delta w_j \leftarrow 0$

  For $t = 1, \ldots, N$

    $o \leftarrow 0$

    For $j = 0, \ldots, d$

      $o \leftarrow o + w_j x_j^t$

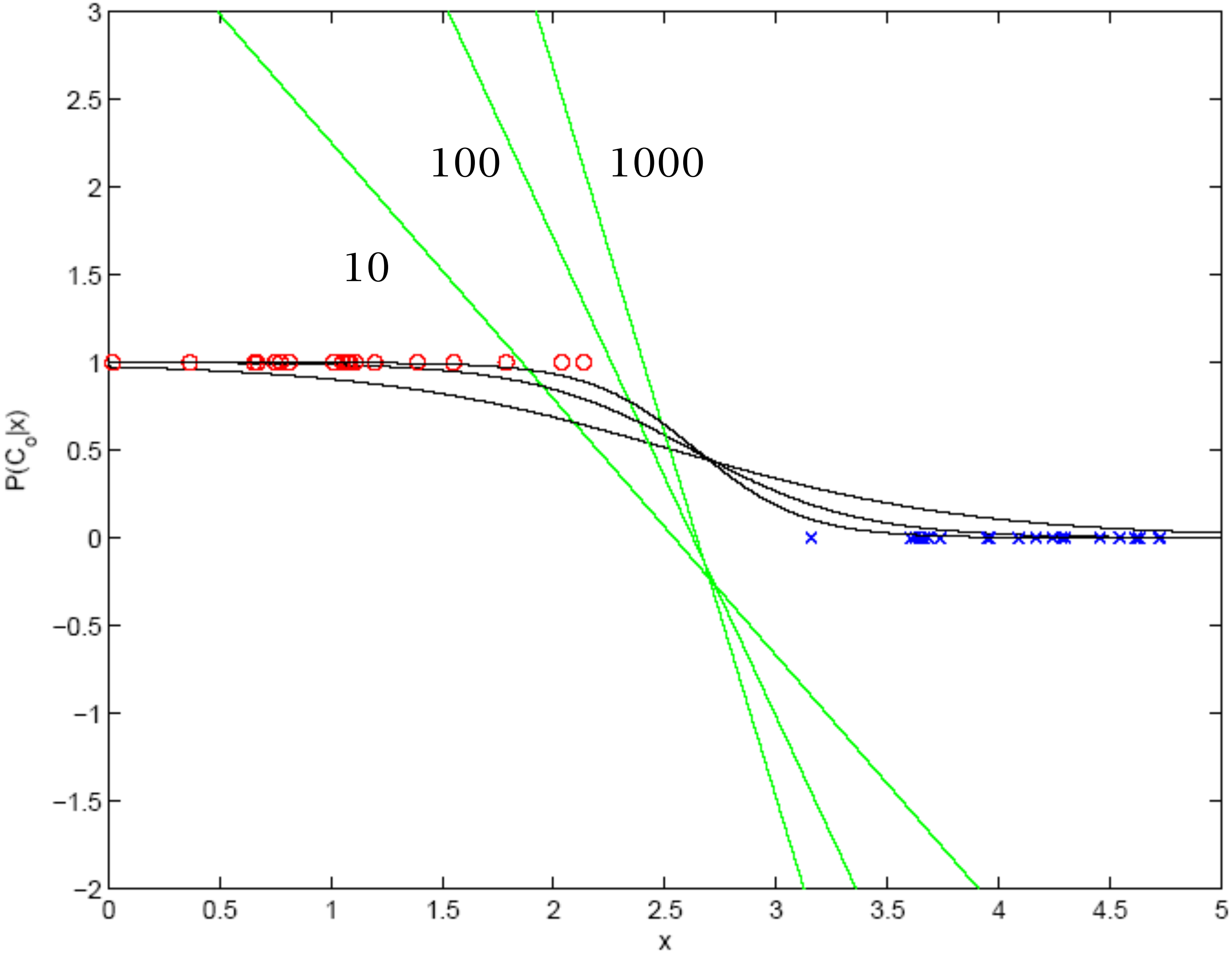
    $y \leftarrow \text{sigmoid}(o)$

    $\Delta w_j \leftarrow \Delta w_j + (r^t - y)x_j^t$

  For $j = 0, \ldots, d$

    $w_j \leftarrow w_j + \eta \Delta w_j$

Until convergence



100
1000
10
P(C_o|x)
x

$K>2$ Classes

$$\mathcal{X} = \{\mathbf{x}^t, \mathbf{r}^t\}_t \quad r^t \mid \mathbf{x}^t \sim \text{Mult}_K(1, \mathbf{y}^t)$$

$$\log \frac{p(\mathbf{x} \mid C_i)}{p(\mathbf{x} \mid C_K)} = \mathbf{w}_i^T \mathbf{x} + w_{i0}^o$$

$$y = \hat{P}(C_i \mid \mathbf{x}) = \boxed{\frac{\exp\left[\mathbf{w}_i^T \mathbf{x} + w_{i0}\right]}{\sum_{j=1}^{K} \exp\left[\mathbf{w}_j^T \mathbf{x} + w_{j0}\right]}}, i = 1, \ldots, K$$

softmax

$$l(\{\mathbf{w}_i, w_{i0}\}_i \mid \mathcal{X}) = \prod_t \prod_i (y_i^t)^{(r_i^t)}$$

$$E(\{\mathbf{w}_i, w_{i0}\}_i \mid \mathcal{X}) = -\sum_t r_i^t \log y_i^t$$

$$\Delta \mathbf{w}_j = \eta \sum_t (r_j^t - y_j^t) \mathbf{x}^t \quad \Delta w_{j0} = \eta \sum_t (r_j^t - y_j^t)$$

For $i = 1, \ldots, K$, For $j = 0, \ldots, d$, $w_{ij} \leftarrow$ rand$(-0.01, 0.01)$

Repeat

  For $i = 1, \ldots, K$, For $j = 0, \ldots, d$, $\Delta w_{ij} \leftarrow 0$

  For $t = 1, \ldots, N$

    For $i = 1, \ldots, K$

      $o_i \leftarrow 0$

      For $j = 0, \ldots, d$

        $o_i \leftarrow o_i + w_{ij} x_j^t$

    For $i = 1, \ldots, K$

      $y_i \leftarrow \exp(o_i) / \sum_k \exp(o_k)$

    For $i = 1, \ldots, K$

      For $j = 0, \ldots, d$

        $\Delta w_{ij} \leftarrow \Delta w_{ij} + (r_i^t - y_i) x_j^t$

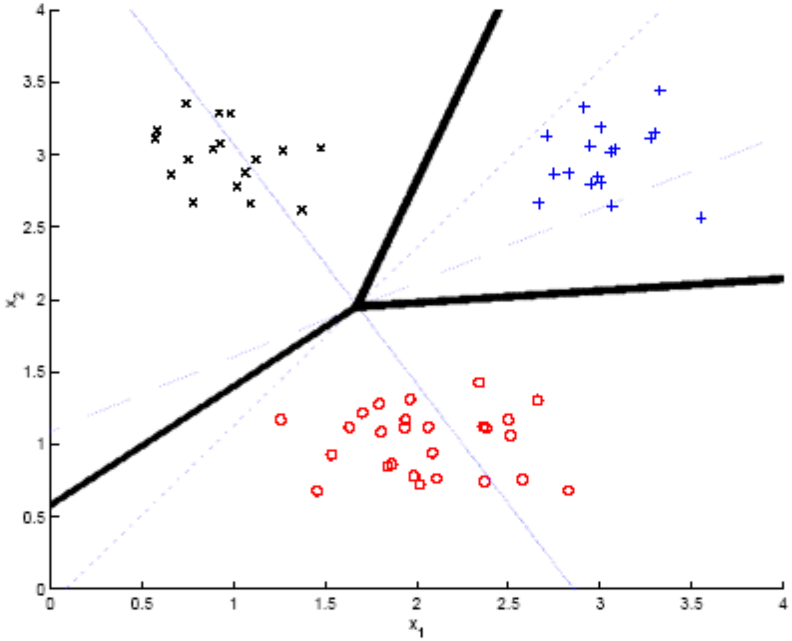
  For $i = 1, \ldots, K$

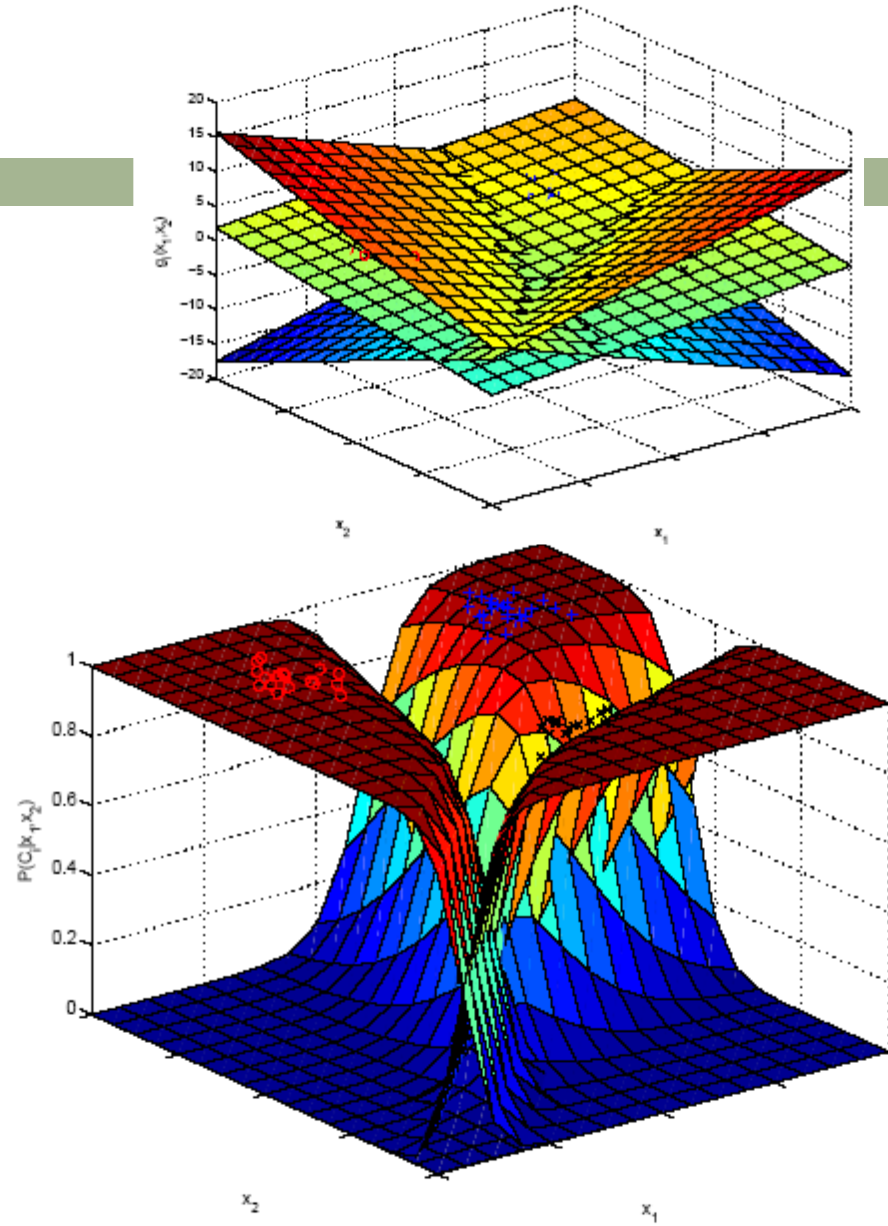
    For $j = 0, \ldots, d$

      $w_{ij} \leftarrow w_{ij} + \eta \Delta w_{ij}$

Until convergence

Example





Generalizing the Linear Model

- Quadratic:

$$\log \frac{p(\mathbf{x} \mid C_i)}{p(\mathbf{x} \mid C_K)} = \mathbf{x}^T \mathbf{W}_i \mathbf{x} + \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

- Sum of basis functions:

$$\log \frac{p(\mathbf{x} \mid C_i)}{p(\mathbf{x} \mid C_K)} = \mathbf{w}_i^T \phi(\mathbf{x}) + w_{i0}$$

where $\varphi(\boldsymbol{x})$ are basis functions. Examples:

  - Hidden units in neural networks (Chapters 11 and 12)
  - Kernels in SVM (Chapter 13)

Discrimination by Regression

- Classes are NOT mutually exclusive and exhaustive

$$r^t = y^t + \varepsilon \text{ where } \varepsilon \sim \mathcal{N}\left(0, \sigma^2\right)$$

$$y^t = \text{sigmoid}\left(\mathbf{w}^T \mathbf{x}^t + w_0\right) = \frac{1}{1 + \exp\left[-\left(\mathbf{w}^T \mathbf{x}^t + w_0\right)\right]}$$

$$l\left(\mathbf{w}, w_0 \mid \mathcal{X}\right) = \prod_t \frac{1}{\sqrt{2\pi\sigma}} \exp\left[-\frac{\left(r^t - y^t\right)^2}{2\sigma^2}\right]$$

$$E\left(\mathbf{w}, w_0 \mid \mathcal{X}\right) = \frac{1}{2} \sum_t \left(r^t - y^t\right)^2$$

$$\Delta \mathbf{w} = \eta \sum_t \left(r^t - y^t\right) y^t \left(1 - y^t\right) \mathbf{x}^t$$

Learning to Rank

- Ranking: A different problem than classification or regression
- Let us say $\mathbf{x}^u$ and $\mathbf{x}^v$ are two instances, e.g., two movies

  We prefer u to v implies that $g(\mathbf{x}^u) > g(\mathbf{x}^v)$

  where $g(\mathbf{x})$ is a score function, here linear:

  $$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$$

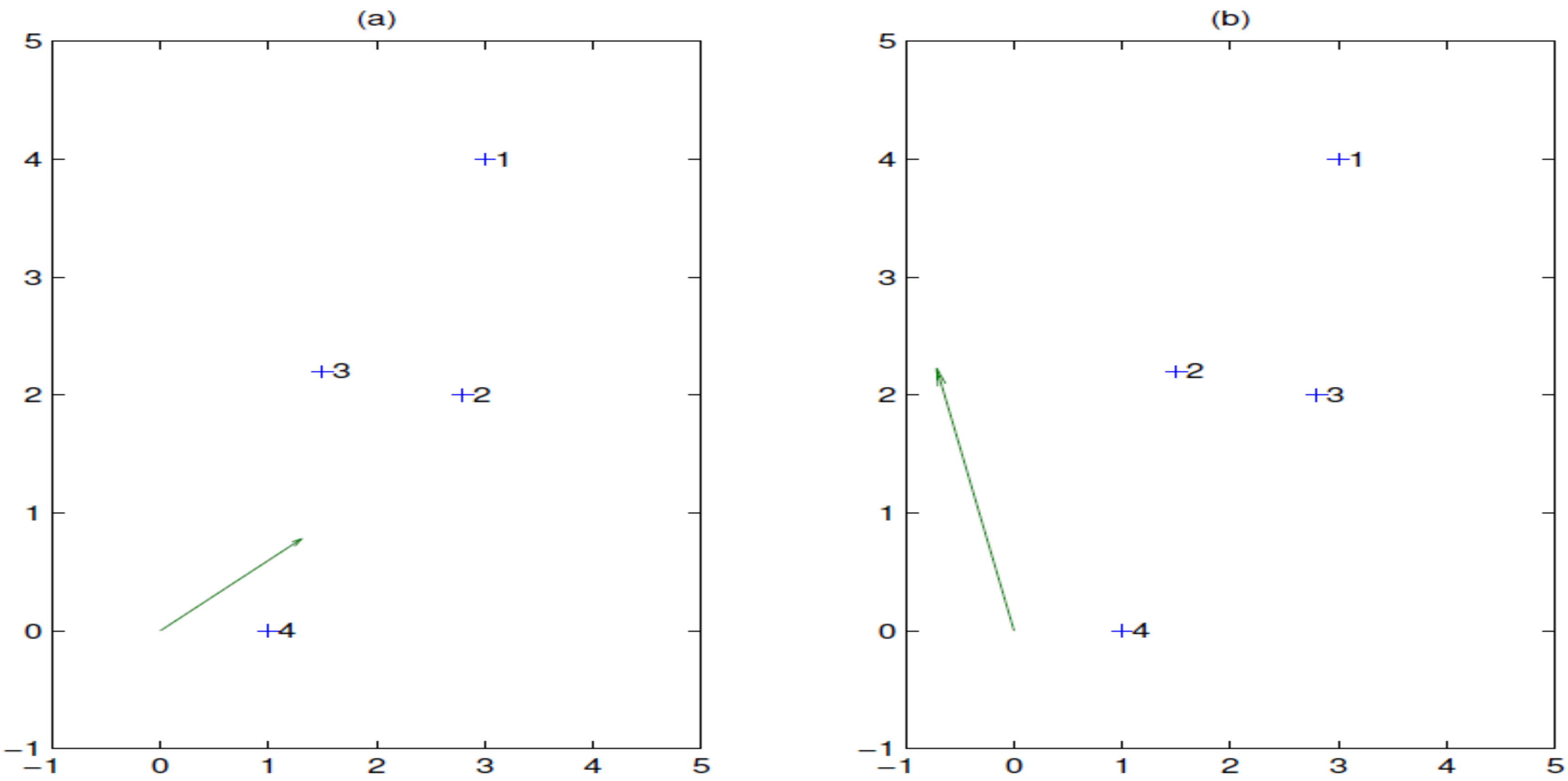
- Find a direction $\boldsymbol{w}$ such that we get the desired ranks when instances are projected along $\boldsymbol{w}$

Ranking Error

- We prefer u to v implies that $g(\mathbf{x}^u)>g(\mathbf{x}^v)$, so error is $g(\mathbf{x}^v)-g(\mathbf{x}^u)$, if $g(\mathbf{x}^u)<g(\mathbf{x}^v)$

$$E(\mathbf{w}|\{r^u, r^v\}) = \sum_{r^u \prec r^v} [g(\mathbf{x}^v|\theta) - g(\mathbf{x}^u|\theta)]_+$$

where a_+ is equal to a if $a \geq 0$ and 0 otherwise.



(a)
(b)
+1
+2
+3
+4