Machines will be capable, within twenty years, of doing any work that a man can do.

- Herbert Simon, 1965

CSE552
Machine Learning

Spring 2025

Dimensionality Reduction

1

Supervised Machine Learning

of experiences | observed outcome | observed input

$$\arg\max_{f \in F} \sum_{i=1}^{n} g(y_i, f(x_i))$$

hypothesis/program space | goodness/performance measure | selected hypothesis

2

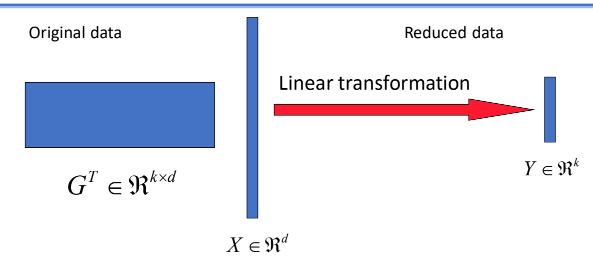
What is Feature Reduction?

- Feature reduction refers to the mapping of the original high-dimensional data onto a lower-dimensional space
  - Criterion for feature reduction can be different based on different problem settings
    - Unsupervised setting: minimize the information loss
    - Supervised setting: maximize the class discrimination
- Given a set of data points, compute the linear transformation (projection)

$$G \in \Re^{d \times k} : x \in \Re^{d} \to y = G^{T} x \in \Re^{k} \ (k << d)$$

3

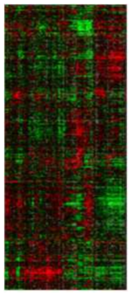
What is Feature Reduction?



$$G \in \Re^{d \times k} : X \to Y = G^{T} X \in \Re^{k}$$

4

High-dimensional Data


Gene expression


Face images


Handwritten digits

Why Reduce Dimensionality?

1. Reduces time complexity: Less computation
2. Reduces space complexity: Less parameters (compression)
3. Saves the cost of observing the feature
4. Simpler models are more robust on small datasets
5. More interpretable; simpler explanation
6. Noise removal
7. Data visualization (structure, groups, outliers, etc) if plotted in 2 or 3 dimensions

Accuracy, Dimensions & Overfitting

- Most machine learning and data mining techniques may not be effective for high-dimensional data
  - Curse of Dimensionality
  - Query accuracy and efficiency degrade rapidly as the dimension increases

- The intrinsic dimension may be small
  - For example, the number of genes responsible for a certain type of disease may be small

Dimensionality Reduction Algorithms

- Unsupervised
  - Principal Component Analysis (PCA)
  - Latent Semantic Indexing (LSI): truncated SVD
  - Independent Component Analysis (ICA)
  - Canonical Correlation Analysis (CCA)

- Supervised
  - Linear Discriminant Analysis (LDA)

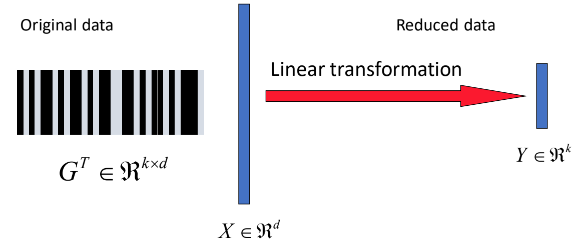
- Semi-supervised
  - Research topic

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) ...

9

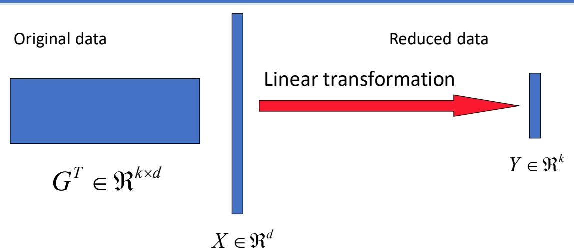
Feature Selection



$$G \in \Re^{d\times k} : X \to Y = G^T X \in \Re^k$$

10

Feature Extraction



$$G \in \Re^{d\times k} : X \to Y = G^T X \in \Re^k$$

11

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 $$j = arg \min_i E(F \cup x_i)$$
  - Add x_j to F if $E(F \cup x_j) < E(F)$
  - Greedy O(d^2) algorithm
- **Backward search**: Start with all features and remove one at a time, if possible
- **Floating search:** Add k, remove l

12

Key Feature Selection Methods

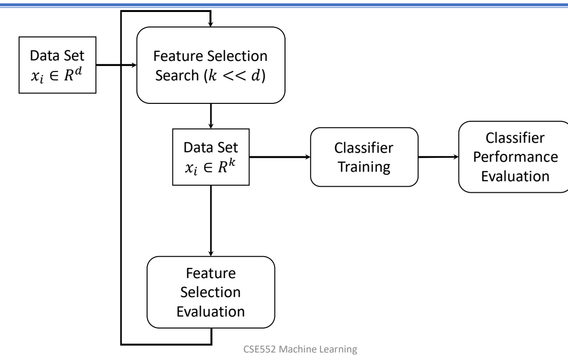
- Open-loop (filter / front-end / preset bias)
  - Select features for which the reduced data set maximizes between-class separability (by evaluating within-class and between-class covariance matrices)
  - no feedback mechanism from the processing algorithm
- Closed-loop (wrapper/ performance bias)
  - Select features based on the processing algorithm performance (feedback mechanism), which serves as a criterion for feature subset selection

Result: data set with reduced number of features according to a specified optimal criterion

What about decision trees?

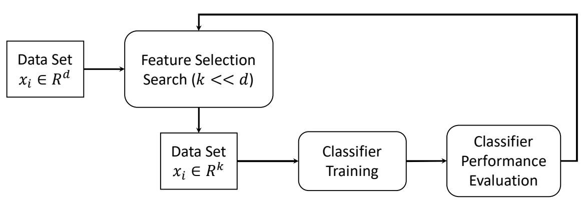
13

Open Loop



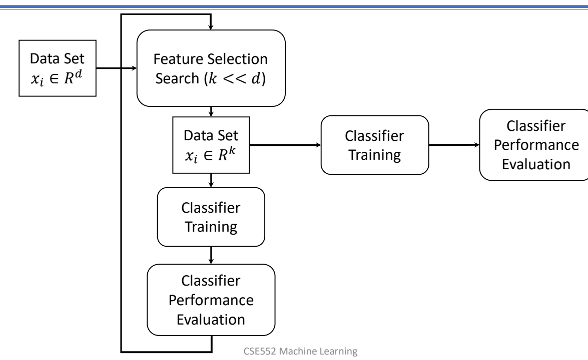
14

Closed Loop



15

Open Loop



16

Optimal Feature Selection

- Procedure for optimal FS:
  - Search procedure, to search through candidate subsets of features (given initial step of a search and stop criteria)
  - FS criterion, Ji, to judge if one subset of features is better than another
- Since feature selection methods are computationally intensive we use heuristic search methods; as a result only sub-optimal solutions can be obtained

Feature Selection

- FS criteria
  - We use criteria based on maximization, where a better subset of features always gives a bigger value of a criterion
  - and the optimal feature subset gives the maximum value of the criterion
- In practice:
  - For the limited data set and FS criterion based on a classifier performance, removing a feature may improve algorithm's performance (up to a point as it then starts to degrade) – peaking phenomenon

FS Paradigms

- Paradigms of optimal FS: minimal representations
- Occam's Razor:
  - The simplest explanation of the observed phenomena in a given domain is the most likely to be a correct one.
- Minimal Description Length (MDL) Principle:
  - Best feature selection can be done by choosing a minimal feature subset that fully describes all classes in a given data set.

MDL Principle

- Can be seen as a formalization of the Occam's razor heuristic
- In short, if a system can be defined in terms of input and the corresponding output data, then in the worst case (longest) it can be described by supplying the entire data set
- On the other hand, if regularities can be discovered, then a much shorter description is possible and can be measured by the MDL principle

Feature Selection

- Criteria
  - A feature selection algorithm uses predefined feature selection criterion (which measures goodness of the subset of features)
- Hope (via MDL principle) is that:
  - by reducing dimensionality we improve generalization ability, up to some max value, but we know that it will start to degrade at some point of reduction

Search

- Goal of SEARCH METHODS: search only through a subset of all possible feature subsets.
- Only sub-optimal subset of features is obtained but at a (much) lower cost.
- Reason
  - The number of possible feature subsets is 2^n where n – original number of features;
  - search for that number of subsets is computationally very expensive.
- Optimal feature selection is NP-hard thus we need to use sub-optimal feature selection methods

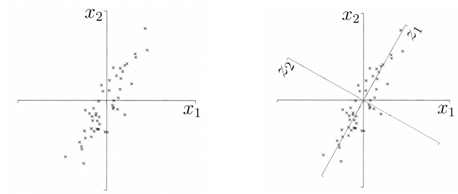
Search Methods

- Exhaustive search
- Branch and Bound
- Individual Feature Ranking
- Sequential Forward and Backward FS
- Stepwise Forward Search
- Stepwise Backward Search
- Probabilistic FS

What is Principal Component Analysis?

- Principal component analysis (PCA)
  - Reduce the dimensionality of a data set by finding a new set of variables, smaller than the original set of variables
  - Retains most of the sample's information
  - Useful for the compression and classification of data
- By information we mean the variation present in the sample, given by the correlations between the original variables
  - The new variables, called principal components (PCs), are uncorrelated, and are ordered by the fraction of the total information each retains

Geometry



- 1st principle component (PC) z_1 is a minimum distance fit to a line in X space
- 2nd PC z_2 is a minimum distance fit to a line in the plane perpendicular to the 1st PC
- PCs are a series of linear least squares fits to a sample, each orthogonal to all the previous

Algebraic Definition of PCs

Given a sample of n observations on a vector of d variables

$$\{x_1, x_2, \cdots, x_n\} \in \Re^d$$

define the first principal component of the sample by the linear transformation

$$z_1 = a_1^T x_j = \sum_{i=1}^{d} a_{i1} x_{ij}, \quad j = 1, 2, \cdots, n.$$

where the vector $a_1 = (a_{11}, a_{21}, \cdots, a_{d1})$

$$x_j = (x_{1j}, x_{2j}, \cdots, x_{dj})$$

is chosen such that $\mathrm{var}[z_1]$ is maximum.

Algebraic Definition of PCs

To find a_1 first note that $\mathrm{var}[z_1] = E((z_1 - \bar{z}_1)^2) = \frac{1}{n}\sum_{i=1}^{n}\left(a_1^T x_i - a_1^T \bar{x}\right)^2$

$$= \frac{1}{n}\sum_{i=1}^{n} a_1^T \left(x_i - \bar{x}\right)\left(x_i - \bar{x}\right)^T a_1 = a_1^T S a_1$$

where $S = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)\left(x_i - \bar{x}\right)^T$ is the covariance matrix.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \text{ is the mean.}$$

In the following, we assume the Data is centered. $\Rightarrow$ $\bar{x} = 0$

Algebraic Definition of PCs

Assume $\bar{x} = 0$

Form the matrix: $X = [x_1, x_2, \cdots, x_n] \in \Re^{d \times n}$

then $S = \frac{1}{n} X X^T$

Obtain eigenvectors of S by computing the SVD of X:

$$X = U \Sigma V^T$$

Algebraic Definition of PCs

To find a_1 that maximizes $\text{var}[z_1]$ subject to $a_1^T a_1 = 1$

Let λ be a Lagrange multiplier

$$L = a_1^T S a_1 - \lambda(a_1^T a_1 - 1)$$

$$\frac{\partial}{\partial a_1} L = S a_1 - \lambda a_1 = 0$$

$$\Rightarrow (S - \lambda I_p) a_1 = 0$$

therefore a_1 is an eigenvector of S

corresponding to the largest eigenvalue $\lambda = \lambda_1$.

Algebraic Definition of PCs

To find the next coefficient vector a_2 maximizing $\text{var}[z_2]$

subject to $\text{cov}[z_2, z_1] = 0$ **uncorrelated**

and to $a_2^T a_2 = 1$

First note that $\text{cov}[z_2, z_1] = a_1^T S a_2 = \lambda_1 a_1^T a_2$

then let λ and φ be Lagrange multipliers, and maximize

$$L = a_2^T S a_2 - \lambda(a_2^T a_2 - 1) - \phi a_2^T a_1$$

Algebraic Definition of PCs

$$L = a_2^T S a_2 - \lambda(a_2^T a_2 - 1) - \phi a_2^T a_1$$

$$\Downarrow$$

$$\frac{\partial}{\partial a_2} L = S a_2 - \lambda a_2 - \phi a_1 = 0 \Rightarrow \phi = 0$$

$$\Downarrow$$

$$S a_2 = \lambda a_2 \quad \text{and} \quad \lambda = a_2^T S a_2$$

Algebraic Definition of PCs

We find that a_2 is also an eigenvector of S

whose eigenvalue $\lambda = \lambda_2$ is the second largest.

In general

$$\text{var}[z_k] = a_k^T S a_k = \lambda_k$$

- The k^{th} largest eigenvalue of S is the variance of the k^{th} PC.
- The k^{th} PC z_k retains the k^{th} greatest fraction of the variation in the sample.

Algebraic Definition of PCs

- Main steps for computing PCs
  - Form the covariance matrix S.
  - Compute its eigenvectors: $\{a_i\}_{i=1}^{d}$
  - Use the first d eigenvectors $\{a_i\}_{i=1}^{d}$ to form the d PCs.
  - The transformation G is given by $G \leftarrow [a_1, a_2, \cdots, a_k]$

A test point $\boxed{x \in \Re^d \rightarrow G^T x \in \Re^k.}$

Principal Components Analysis (PCA)

- 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: $z = \boldsymbol{w}^T\boldsymbol{x}$
- Find ***w*** such that Var(z) is maximized

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

where

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

Principal Components Analysis (PCA)

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

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

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

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 $\boldsymbol{w}_1$

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

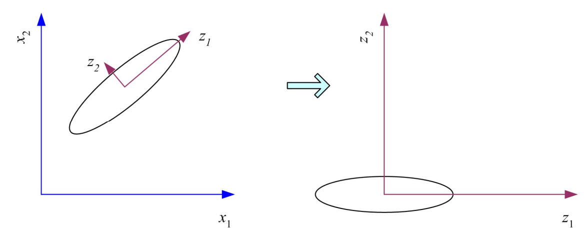
$\Sigma\, \boldsymbol{w}_2 = \alpha\, \boldsymbol{w}_2$ that is, $\boldsymbol{w}_2$ is another eigenvector of Σ and so on.

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

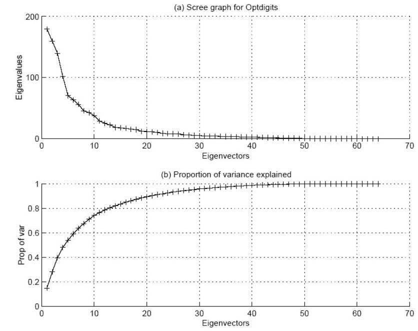
$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

when λ_i are sorted in descending order

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

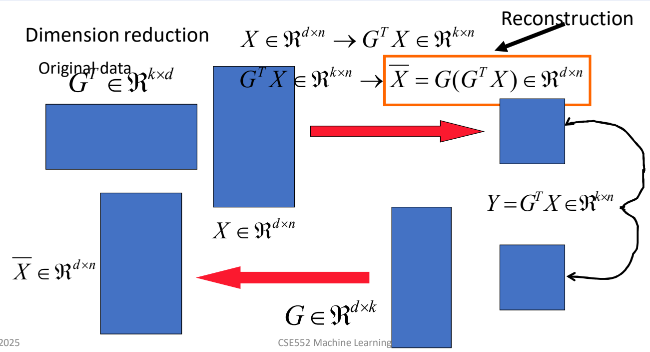
37

How to choose k ?



38

Optimality Property of PCA



39

Optimality Property of PCA

Main theoretical result:

The matrix G consisting of the first d eigenvectors of the covariance matrix S solves the following min problem:

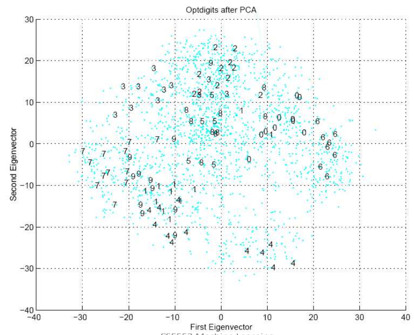
$$\boxed{\min_{G\in\Re^{d\times k}} \left\| X - G(G^T X) \right\|_F^2 \text{ subject to } G^T G = I_k}$$

$$\left\| X - \overline{X} \right\|_F^2$$

reconstruction error

PCA projection minimizes the reconstruction error among all linear projections of size k.

40



Factor Analysis

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

$$x_i - \mu_i = v_1 \; z_1 + v_{i_2} z_2 + \cdots + v_{i_k} z_k + \varepsilon_i$$

where z_j, j =1,...,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**

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,...,k are the **latent factors** with

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

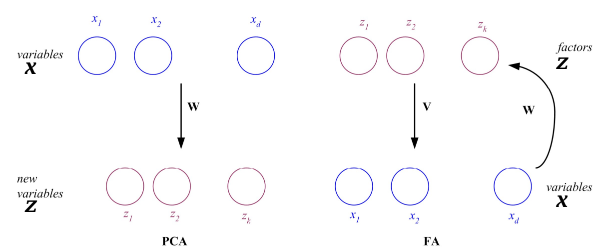
ε_i are the **noise sources**

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

and v_{ij} are the **factor loadings**

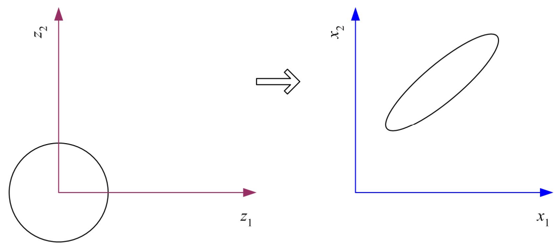
PCA vs FA

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



Factor Analysis

- In FA, factors z_j are stretched, rotated and translated to generate $\boldsymbol{x}$



45

Multidimensional Scaling

- Given pairwise distances between N points,

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

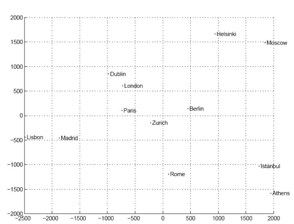
  place on a low-dim map s.t. distances are preserved.
- $\boldsymbol{z} = \boldsymbol{g}\,(\boldsymbol{x} \mid \vartheta)$ Find ϑ that min **Sammon stress**

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

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

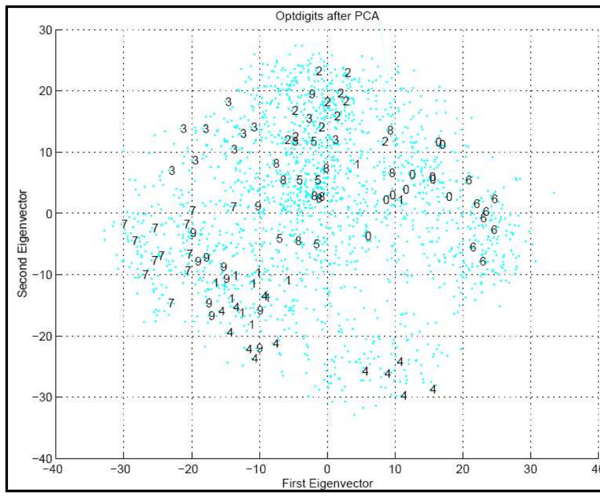
46

Map of Europe by MDS



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

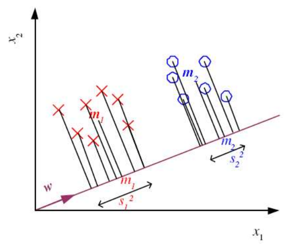
47



48

Linear Discriminant Analysis

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



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

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

49

Linear Discriminant Analysis

- Between-class scatter:

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

- Within-class scatter:

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

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

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

50

Fisher's Linear Discriminant

- Find ***w*** that max

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

- LDA solution:

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

- Parametric solution:

$$\boldsymbol{w} = \Sigma^{-1}(\mu_1 - \mu_2)$$
$$\text{when } p(\boldsymbol{x} \mid C_i) \sim \mathcal{N}(\mu_i, \Sigma)$$

51

K>2 Classes

- Within-class scatter:

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

- Between-class scatter:

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

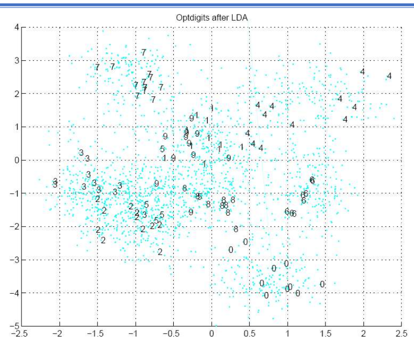
- Find **W** that max

$$J(\mathbf{W}) = \frac{\left|\mathbf{W}^T \mathbf{S}_B \mathbf{W}\right|}{\left|\mathbf{W}^T \mathbf{S}_W \mathbf{W}\right|}$$

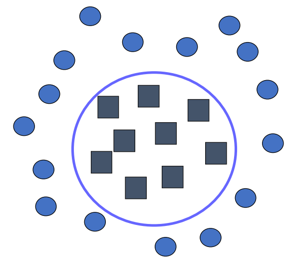
The largest eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$
Maximum rank of K-1

52

LDA



Motivation



Linear projections will not detect the pattern

Nonlinear PCA using Kernels

- Traditional PCA applies linear transformation
  - May not be effective for nonlinear data
- Solution: apply nonlinear transformation to potentially very high-dimensional space.

$$\phi : x \to \phi(x)$$

- Computational efficiency: apply the kernel trick.
  - Require PCA can be rewritten in terms of dot product.

$$K(x_i, x_j) = \phi(x_i) \bullet \phi(x_j)$$

Nonlinear PCA using Kernels

Rewrite PCA in terms of dot product

Assume the data has been centered, i.e., $\sum_i x_i = 0$.

The covariance matrix S can be written as $S = \frac{1}{n}\sum_i x_i x_i^T$

Let v be The eigenvector of S corresponding to nonzero eigenvalue

$$\boxed{Sv = \frac{1}{n}\sum_i x_i x_i^T v = \lambda v \Rightarrow v = \frac{1}{n\lambda}\sum_i (x_i^T v) x_i}$$

Eigenvectors of S lie in the space spanned by all data points.

Nonlinear PCA using Kernels

$$Sv = \frac{1}{n}\sum_i x_i x_i^T v = \lambda v \Rightarrow v = \frac{1}{n\lambda}\sum_i (x_i^T v) x_i$$

The covariance matrix can be written in matrix form:

$$S = \frac{1}{n} XX^T, \text{ where } X = [x_1, x_2, \cdots, x_n].$$

$$v = \sum_i \alpha_i x_i = X\alpha \qquad Sv = \frac{1}{n} XX^T X\alpha = \lambda X\alpha$$

$$\frac{1}{n}(X^T X)(X^T X)\alpha = \lambda (X^T X)\alpha$$

$$\boxed{\frac{1}{n}(X^T X)\alpha = \lambda \alpha}$$

Any benefits?

Nonlinear PCA using Kernels

Next consider the feature space: $\phi : x \to \phi(x)$

$$S^\phi = \frac{1}{n} X^\phi (X^\phi)^T, \text{ where } X^\phi = [x_1^\phi, x_2^\phi, \cdots, x_n^\phi].$$

$$v = \sum_i \alpha_i \phi(x_i) = X^\phi \alpha \qquad \frac{1}{n}(X^\phi)^T X^\phi \alpha = \lambda \alpha$$

The (i,j)-th entry of $(X^\phi)^T X^\phi$ is $\phi(x_i) \bullet \phi(x_j)$

Apply the kernel trick: $K(x_i, x_j) = \phi(x_i) \bullet \phi(x_j)$

K is called the kernel matrix. $\boxed{\frac{1}{n} K\alpha = \lambda \alpha}$

Nonlinear PCA using Kernels

- Projection of a test point x onto v:

$$\phi(x) \bullet v = \phi(x) \bullet \sum_i \alpha_i \phi(x_i)$$

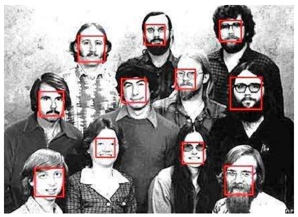
$$= \sum_i \alpha_i \phi(x) \bullet \phi(x_i) = \sum_i \alpha_i K(x, x_i)$$

Explicit mapping is not required here.

Eigenfaces

(Slides adapted from Lazebnik, Grauman & Lowe)

Face Detection and Recognition

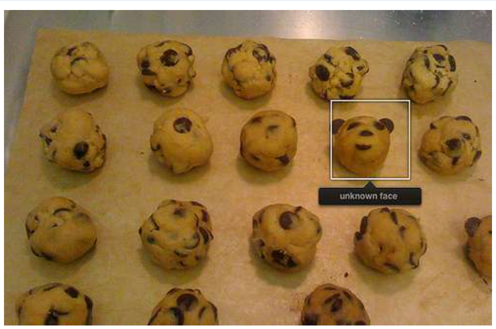


61

Face Detection and Recognition

62

Consumer Application: iPhoto 2009



63

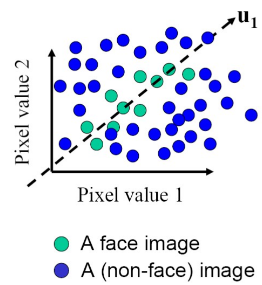
The Space of All Face Images

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
- However, relatively few 10,000-dimensional vectors correspond to valid face images
- We want to effectively model the subspace of face images

64

The Space of All Face Images

We want to construct a low-dimensional linear subspace that best explains the variation in the set of face images



Principal Component Analysis

- Given: N data points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ in R^d
- We want to find a new set of features that are linear combinations of original ones:

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

($\boldsymbol{\mu}$: mean of data points)

- What unit vector $\mathbf{u}$ in R^d captures the most variance of the data?

Principal Component Analysis

Direction that maximizes the variance of the projected data:

$$\begin{aligned} var(u) &= \frac{1}{N}\sum_{i=1}^{N} \underbrace{\mathbf{u}^{\mathrm{T}}(\mathbf{x}_i - \mu)}_{\text{Projection of data point}}(\mathbf{u}^{\mathrm{T}}(\mathbf{x}_i - \mu))^{\mathrm{T}} \\ &= \mathbf{u}^{\mathrm{T}} \underbrace{\left[\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^{\mathrm{T}}\right]}_{\text{Covariance matrix of data}} \mathbf{u} \\ &= \mathbf{u}^{\mathrm{T}}\Sigma\mathbf{u} \end{aligned}$$

The direction that maximizes the variance is the eigenvector associated with the largest eigenvalue of Σ

Principal Component Analysis

- The direction that captures the maximum covariance of the data is the eigenvector corresponding to the largest eigenvalue of the data covariance matrix
- Furthermore, the top *k* orthogonal directions that capture the most variance of the data are the *k* eigenvectors corresponding to the *k* largest eigenvalues

Eigenfaces: Key idea

- Assume that most face images lie on a low-dimensional subspace determined by the first k (k<d) directions of maximum variance
- Use PCA to determine the vectors or "eigenfaces" $\mathbf{u}_1,\ldots\mathbf{u}_k$ that span that subspace
- Represent all face images in the dataset as linear combinations of eigenfaces

M. Turk and A. Pentland, [Face Recognition using Eigenfaces](), CVPR 1991

Eigenfaces (1)

- Calculation of Eigenfaces
  - Calculate average face : v
  - Collect difference between training images and average face in matrix A (M by N), where M is the number of pixels and N is the number of images

$$A = [\boldsymbol{u}_1^1 - v, \ldots, \boldsymbol{u}_n^1 - v, \ldots, \boldsymbol{u}_1^p - v, \ldots, \boldsymbol{u}_n^p - v]$$

  - The eigenvectors of covariance matrix C (M by M) give the eigenfaces
- M is usually big, so this process would be time consuming
- What to do? $C = AA^T$

Eigenfaces (2)

- Calculation of Eigenvectors of C
  - If the number of data points is smaller than the dimension (N<M), then there will be only N-1 meaningful eigenvectors.
  - Instead of directly calculating the eigenvectors of C, we can calculate the eigenvalues and the corresponding eigenvectors of a much smaller matrix L (N by N)

$$L = A^T A$$

  - If λi are the eigenvectors of L then A λi are the eigenvectors for C
    - The eigenvectors are in the descent order of the corresponding eigenvalues

Eigenfaces (3)

- Representation of Face Images using Eigenfaces
- The training face images and new face images can be represented as linear combination of the eigenfaces.
- When we have a face image u :

$$u = \sum_i a_i \phi_i$$

- Since the eigenvectors are orthogonal :

$$a_i = u^T \phi_i$$

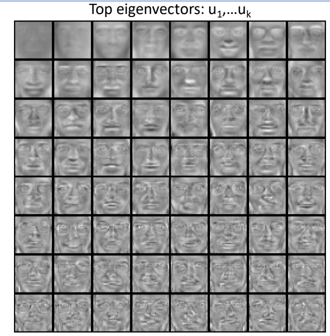
Eigenfaces Example

Training images $\mathbf{x}_1,\ldots,\mathbf{x}_N$

Eigenfaces Example

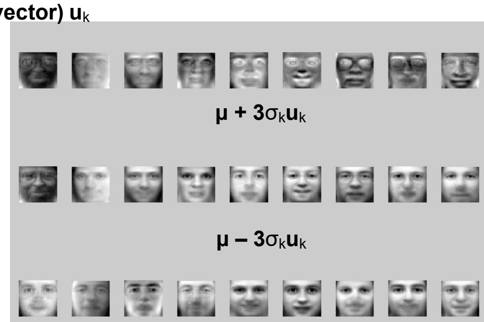
Mean: μ

Top eigenvectors: $u_1,\ldots u_k$



Eigenfaces Example

Principal component (eigenvector) $\mathbf{u}_k$



Eigenfaces Example

Face **x** in "face space" coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu),\ldots,\mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)]$$
$$= \quad w_1,\ldots,w_k$$

Reconstruction:

 =  + 

$\hat{\mathbf{x}} = \boldsymbol{\mu} + w_1\mathbf{u}_1 + w_2\mathbf{u}_2 + w_3\mathbf{u}_3 + w_4\mathbf{u}_4 + \ldots$

Eigenfaces Example

Face **x** in "face space" coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x} - \mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x} - \mu)]$$
$$= \quad w_1, \ldots, w_k$$

Eigenfaces Example

First three eigenfaces

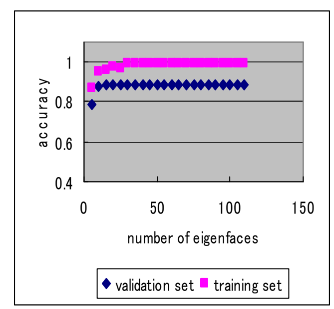
Recognition with Eigenfaces

- Process labeled training images:
  - Find mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
  - Find k principal components (eigenvectors of $\boldsymbol{\Sigma}$) $\mathbf{u}_1, \ldots \mathbf{u}_k$
  - Project each training image $\mathbf{x}_i$ onto subspace spanned by principal components:
    $(w_{i1},\ldots,w_{ik}) = (\mathbf{u}_1^T(\mathbf{x}_i - \boldsymbol{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x}_i - \boldsymbol{\mu}))$
- Given novel image **x**:
  - Project onto subspace:
    $(w_1,\ldots,w_k) = (\mathbf{u}_1^T(\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x} - \boldsymbol{\mu}))$
  - Optional: check reconstruction error $\mathbf{x} - \mathbf{x}$ to determine whether image is really a face
  - Classify as closest training face in k-dimensional subspace

Classification Using Nearest Neighbor

- Save average coefficients for each person. Classify new face as the person with the closest average
- Recognition accuracy increases with number of eigenfaces till 15.
- Later eigenfaces do not help much with recognition

- Best recognition rates
  - Training set 99%
  - Test set 89%

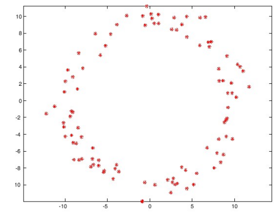


Limitations

Global appearance method: not robust to misalignment, background variation

Limitations

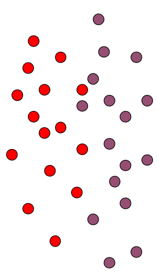
- PCA assumes that the data has a Gaussian distribution (mean μ, covariance matrix Σ)



- The shape of this dataset is not well described by its principal components

Limitations

The direction of maximum variance is not always good for classification



Thanks for listening!