Lecture Slides for

INTRODUCTION TO

Machine Learning

ETHEM ALPAYDIN
© The MIT Press, 2004

Edited for CS 536 Fall 2005 – Rutgers University
Ahmed Elgammal
alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml

CHAPTER 6:
Dimensionality Reduction
Why Reduce Dimensionality?

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

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)
Subset Selection

- There are $2^d$ subsets of $d$ features
- Forward search: Add the best feature at each step
  - Set of features $F$ initially $\emptyset$.
  - 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 $l$)

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: $z = w^T x$
- Find $w$ such that $\text{Var}(z)$ is maximized
  $$ \text{Var}(z) = \text{Var}(w^T x) = \text{E}[(w^T x - w^T \mu)^2] $$
  $$ = \text{E}[(w^T x - w^T \mu)(w^T x - w^T \mu)] $$
  $$ = \text{E}[w^T (x - \mu)(x - \mu)^T w] $$
  $$ = w^T \text{E}[(x - \mu)(x - \mu)^T] w = w^T \Sigma w $$
  where $\text{Var}(x) = \text{E}[(x - \mu)(x - \mu)^T] = \Sigma$
Maximize \( \text{Var}(z) \) subject to \( ||w||=1 \)

\[
\max_{w_1} w_1^T \Sigma w_1 - \alpha (w_1^T w_1 - 1)
\]

\( x w_1 = \alpha w_1 \) that is, \( w_1 \) is an eigenvector of \( x \)
Choose the one with the largest eigenvalue for \( \text{Var}(z) \) to be max

Second principal component: Max \( \text{Var}(z_2) \), s.t., \( ||w_2||=1 \) and orthogonal to \( w_1 \)

\[
\max_{w_2} w_2^T \Sigma w_2 - \alpha (w_2^T w_2 - 1) - \beta (w_2^T w_1 - 0)
\]

\( x w_2 = \alpha w_2 \) that is, \( w_2 \) is another eigenvector of \( x \) and so on.

What PCA does

\[
z = W^T (x - m)
\]

where the columns of \( W \) are the eigenvectors of \( x \), and \( m \) is sample mean

Centers the data at the origin and rotates the axes

\[
x_1$
\]

\[
z_2$
\]

\[
z_1$
\]
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 \( \lambda_i \) are sorted in descending order
- Typically, stop at PoV > 0.9
- Scree graph plots of PoV vs k, stop at “elbow”
Principle Component Analysis

PCA

- Given a set of points \( \{x_1, x_2, \ldots, x_N\}, x_i \in \mathbb{R}^d \)
- We are looking for a linear projection: a linear combination of orthogonal basis vectors

\[
x \approx A \cdot c,
\]

\[
R^d, m \ll d
\]

What is the projection that minimizes the reconstruction error? 

\[
E = \sum |x_i - Ac_i|
\]
Principle Component Analysis (PCA)

- Given a set of points
  \[ \{x_1, x_2, \cdots, x_N\}, x_i \in \mathbb{R}^d \]
- Center the points: compute
  \[ \mu = \frac{1}{N} \sum x_i \]
  \[ P = [x_1 - \mu, x_2 - \mu, \cdots, x_N - \mu], x_i \in \mathbb{R}^d \]
- Compute covariance matrix
  \[ Q = PP^T \]
- Compute the eigenvectors for \( Q \)
  \[ Qe_k = \lambda_k e_k \]
- Eigenvectors are the orthogonal basis we are looking for

Singular Value Decomposition

- SVD: If \( A \) is a real \( m \times n \) matrix then there exist orthogonal matrices
  \( U (m \times m) \) and \( V (n \times n) \) such that
  \[ U^TAV = \Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_p) \]
  \[ p = \min\{m, n\} \]
  \[ U^TAV = \Sigma \]
  \[ A = U \Sigma V^T \]
- **Singular values**: Non negative square roots of the eigenvalues of \( AA^T \).
  Denoted \( \sigma_i, i=1, \ldots, n \)
- \( AA^T \) is symmetric \( \Rightarrow \) eigenvalues and singular values are real.
- Singular values arranged in decreasing order.
  \[ A^T A = (U \Sigma V^T)^T U \Sigma V^T = V \Sigma^T U^T U \Sigma V^T = V \Sigma^T \Sigma V^T = V \Sigma^2 V^{-1} \]
  \[ (A^T A)V = \Sigma^2 \]
  \[ (A^T A)v = v \lambda \]
SVD for PCA

- SVD can be used to efficiently compute the image basis
  \[ PP' = (U \Sigma V')(U \Sigma V')' = U \Sigma V' V \Sigma' U' = U \Sigma^2 U' = U \Sigma^2 U^{-1} \]
  \[ (PP')U = U \Sigma^2 \]
  \[ (PP')v = \nu \lambda \]

- U are the eigen vectors (basis)
- Most important thing to notice: Distance in the eigen-space is an approximation to the correlation in the original space
  \[ \| x_i - x_j \| \approx \| c_i - c_j \| \]

PCA

- Most important thing to notice: Distance in the eigen-space is an approximation to the correlation in the original space
  \[ \| x_i - x_j \| \approx \| c_i - c_j \| \]
Eigenface

- Use PCA and subspace projection to perform face recognition
- How to describe a face as a linear combination of face basis
- Matthew Turk and Alex Pentland “Eigenfaces for Recognition” 1991

Face Recognition - Eigenface

- MIT Media Lab - Face Recognition demo page http://vismod.media.mit.edu/vismod/demos/facerec/
Factor Analysis

- Find a small number of factors $z$, which when combined generate $x$:
  \[ x_i - \mu_i = v_{i1}z_1 + v_{i2}z_2 + \cdots + v_{ik}z_k + \epsilon_i \]

  where $z_j, j = 1,...,k$ are the latent factors with
  \[ E[z_j] = 0, \text{Var}(z_j) = 1, \text{Cov}(z_i, z_j) = 0, i \neq j, \]
  \[ \epsilon_i \] are the noise sources
  \[ E[\epsilon_i] = \psi_i, \text{Cov}(\epsilon_i, \epsilon_j) = 0, i \neq j, \text{Cov}(\epsilon_i, z_j) = 0, \]
  and $v_{ij}$ are the factor loadings

PCA vs FA

- PCA From $x$ to $z$ \[ z = W^T(x - \mu) \]
- FA From $z$ to $x$ \[ x - \mu = Vz + \epsilon \]
Factor Analysis

- In FA, factors $z_j$ are stretched, rotated and translated to generate $x$.

Multidimensional Scaling

- Given pairwise distances between $N$ points, $d_{ij}, i,j = 1, ..., N$.
  Place on a low-dim map s.t. distances are preserved.

- $z = g(x \mid \theta)$

Find $\theta$ that min Sammon stress:

$$E(\theta \mid X) = \sum_{r,s} \frac{\left\| z_r^\tau - z_s^\tau \right\|^2 - \left\| x_r^\tau - x_s^\tau \right\|^2}{\left\| x_r^\tau - x_s^\tau \right\|^2}$$

$$= \sum_{r,s} \frac{\left\| g(x_r^\tau \mid \theta) - g(x_s^\tau \mid \theta) \right\|^2 - \left\| x_r^\tau - x_s^\tau \right\|^2}{\left\| x_r^\tau - x_s^\tau \right\|^2}$$
Map of Europe by MDS

Linear Discriminant Analysis

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

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

$$m_1 = \frac{\sum_{t} w^T x^t r^t}{\sum_{t} r^t}$$

$$s_1^2 = \sum_{t} (w^T x^t - m_1)^2 r^t$$
Between-class scatter:
\[(m_1 - m_2)^2 = (w^T m_1 - w^T m_2)^2\]
\[= w^T (m_1 - m_2)(m_1 - m_2)^T w\]
\[= w^T S_B w \text{ where } S_B = (m_1 - m_2)(m_1 - m_2)^T\]

Within-class scatter:
\[s_1^2 = \sum_t (w^T x_t^i - m_1)^2 r_t^i\]
\[= \sum_t w^T (x_t^i - m_1)(x_t^i - m_1)^T w r_t^i = w^T S_1 w\]

where \(S_1 = \sum_t (x_t^i - m_1)(x_t^i - m_1)^T r_t^i\)
\[s_2^2 = w^T S_w w \text{ where } S_w = S_1 + S_2\]

Fisher’s Linear Discriminant

Find \(w\) that max
\[J(w) = \frac{w^T S_B w}{w^T S_w w} = \frac{|w^T (m_1 - m_2)|^2}{w^T S_w w}\]

LDA soln:
\[w = c \cdot S_w^{-1}(m_1 - m_2)\]

Parametric soln:
\[W = \Sigma^{-1} (\mu_1 - \mu_2)\]
when \(p(x | C_i) \sim N(\mu_i, \Sigma)\)
K>2 Classes

- Within-class scatter:
  \[ S_W = \sum_{i=1}^{K} S_i \quad S_i = \sum_{c} r_i^t (x_i - m_i) (x_i - m_i)^T \]

- Between-class scatter:
  \[ S_B = \sum_{i=1}^{K} N_i (m_i - m) (m_i - m)^T \quad m = \frac{1}{K} \sum_{i=1}^{K} m_i \]

- Find \( W \) that max
  \[ J(W) = \frac{|W^T S_B W|}{|W^T S_W W|} \]
  The largest eigenvectors of \( S_W^{-1} S_B \)
  Maximum rank of K-1
Separating Style and Content

- Objective: Decomposing two factors using linear methods
  - Content: which character
  - Style: which font
- “Bilinear models”
- J. Tenenbaum and W. Freeman “Separating Style and Content with Bilinear Models” Neural computation 2000

Bilinear Models

- Symmetric bilinear model
  \[ y^{sc} = \sum_{i,j} w_{ij} a_i^s b_j^c \]
Bilinear models

- Asymmetric bilinear model: use style dependent basis vectors

\[ y^{sc} = A^s b^c \]

Head pose as style factor
person as content

Person as style factor
pose as content

Figures from J. Tenenbaum and W. Freeman “Separating Style and Content with Bilinear Models” Neural computation 2000