Dimensionality Reduction

- Olive slides: Alpaydin
- Black slides: extra content.

### 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

### Feature Selection vs Extraction

- **Feature selection**: Choosing $k < d$ important features, ignoring the remaining $d - k$
  - Subset selection algorithms
- **Feature extraction**: Project the original $x_i, i = 1, \ldots, d$ dimensions to new $k < d$ dimensions, $z_j, j = 1, \ldots, k$

### 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 $j = \arg\min_i E(F \cup x_i)$
  - 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)

Note: \(Q\) means eigenvector matrix of the covariance matrix, in Haykin slides.

Motivation

- How can we project the given data so that the variance in the projected points is maximized?

Eigenvalues/Eigenvectors

- For a square matrix \(A\), if a vector \(x\) and a scalar value \(\lambda\) exists so that
  \[(A - \lambda I)x = 0\]
  then \(x\) is called an eigenvector of \(A\) and \(\lambda\) an eigenvalue.
- Note, the above is simply
  \[Ax = \lambda x\]
- An intuitive meaning is: \(x\) is the direction in which applying the linear transformation \(A\) only changes the magnitude of \(x\) (by \(\lambda\)) but not the angle.
- There can be as many as \(n\) eigenvector/eigenvalue for an \(n \times n\) matrix.

Eigenvalue/Eigenvector Example

- Red: original data \(x\)
- Green: projected data using \(A = \begin{bmatrix} 3 & 5 \\ 2 & 1 \end{bmatrix}\).
- Blue: Eigenvectors \(v_1=(0.91, 0.42), v_2=(-0.76,0.65)\), \(\lambda_1 = 5.3, \lambda_2 = -1.3\). Octave/Matlab code: \([V,Lambda]=eig(A)\)
- Magenta: \(A\) times eigenvectors.
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: \( z = w^T x \)
- Find \( w \) such that \( \text{Var}(z) \) is maximized

\[
\text{Var}(z) = \text{Var}(w^T x) = E[(w^T x - w^T \mu)^2] = E[(w^T x - w^T \mu)(w^T x - w^T \mu)] = E[w^T (x - \mu)(x - \mu)^T w] = w^T E[(x - \mu)(x - \mu)^T] w = w^T \Sigma w
\]

where \( \text{Var}(x) = E[(x - \mu)(x - \mu)^T] = \Sigma \)

What PCA does

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

where the columns of \( W \) are the eigenvectors of \( \Sigma \) 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 $\lambda_i$ are sorted in descending order

- Typically, stop at PoV > 0.9

- Scree graph plots of PoV vs $k$, stop at “elbow”

PCA: Usage

- Project input $x$ to the principal directions:
  $$a = Q^T x.$$

- We can also recover the input from the projected point $a$:
  $$x = (Q^T)^{-1} a = Qa.$$

- Note that we don’t need all $m$ principal directions, depending on how much variance is captured in the first few eigenvalues: We can do dimensionality reduction.
PCA: Dimensionality Reduction

- **Encoding**: We can use the first \( l \) eigenvectors to encode \( \mathbf{x} \).
  \[
  [ a_1, a_2, ..., a_l ]^T = [ q_1, q_2, ..., q_l ]^T \mathbf{x}.
  \]

- Note that we only need to calculate \( l \) projections \( a_1, a_2, ..., a_l \), where \( l \leq m \).

- **Decoding**: Once \( [ a_1, a_2, ..., a_l ]^T \) is obtained, we want to reconstruct the full \( [ x_1, x_2, ..., x_l, ..., x_m ]^T \).
  \[
  \mathbf{x} = \mathbf{Q} a = [ q_1, q_2, ..., q_l ][ a_1, a_2, ..., a_l ]^T = \hat{\mathbf{x}}.
  \]

Or, alternatively
  \[
  \hat{\mathbf{x}} = \mathbf{Q} [ a_1, a_2, ..., a_l, 0, 0, ..., 0 ]^T.
  \]

\( m - l \) zeros

PCA Example

\[0.70285 \ -0.71134
0.71134 \ 0.70285
\]

\( \lambda =
\[
[0.14425 \ 0.00000
0.00000 \ 0.02161]
\]

PCA: Total Variance

- The total variance of the \( m \) components of the data vector is
  \[
  \sum_{j=1}^{m} \sigma_{j}^2 = \sum_{j=1}^{m} \lambda_j.
  \]

- The truncated version with the first \( l \) components have variance
  \[
  \sum_{j=1}^{l} \sigma_{j}^2 = \sum_{j=1}^{l} \lambda_j.
  \]

- The larger the variance in the truncated version, i.e., the smaller
  the variance in the remaining components, the more accurate the
  dimensionality reduction.

Factor Analysis

- **Find a small number of factors** \( \mathbf{z} \), which when combined generate \( \mathbf{x} \):
  \[
  \mathbf{x}_i - \mu_i = \nu_{i1}z_1 + \nu_{i2}z_2 + ... + \nu_{ik}z_k + \varepsilon_i
  \]

  where \( z_i, j = 1, ..., k \) are the latent factors with
  \[
  \mathbb{E}[z_j]=0, \ \text{Var}(z_j)=1, \ \text{Cov}(z_i, z_j)=0, \ i \neq j,
  \]

  \( \varepsilon_i \) are the noise sources
  \[
  \mathbb{E}[\varepsilon_i]=\psi_i, \ \text{Cov}(\varepsilon_i, \varepsilon_j)=0, \ i \neq j, \ \text{Cov}(\varepsilon_i, z_j)=0
  \]

  and \( \nu_{ij} \) are the factor loadings
**PCA vs FA**

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

**Factor Analysis**

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

**Singular Value Decomposition and Matrix Factorization**

- Singular value decomposition: $X = V A W^T$
  - $V$ is $N \times N$ and contains the eigenvectors of $X X^T$
  - $W$ is $d \times d$ and contains the eigenvectors of $X^T X$
  - $A$ is $N \times d$ and contains singular values on its first $k$ diagonal
  
- $X = u_1 a_1 v_1^T + \ldots + u_k a_k v_k^T$ where $k$ is the rank of $X$

**Multidimensional Scaling**

- Given pairwise distances between $N$ points, $d_{ij}, i,j = 1, \ldots, N$.
  - Place on a low-dim map such that distances are preserved (by feature embedding).
  
- $z = g(x \mid \theta)$
  - Find $\theta$ that minimizes Sammon stress:
  \[
  E(\theta \mid X) = \sum_{r,s} \frac{\left( \| z_r - z_s \| - \| x_r - x_s \| \right)^2}{\| x_r - x_s \|^2}
  \]
  \[
  = \sum_{r,s} \frac{\left( g(x_r \mid \theta) - g(x_s \mid \theta) - \| x_r - x_s \| \right)^2}{\| x_r - x_s \|^2}
  \]
A topological space that is locally Euclidean (flat, not curved).

Dimensionality of the manifold = dimensionality of the Euclidean space it resembles, locally.
- Straight line, wiggly curves, etc. are 1D manifolds.
- Flat plane, surface of sphere, etc. are 2D manifolds.

Detecting curvature of space: sum of internal angles of triangle = 180°?

**Isomap**

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

Geodesic distance = Shortest path.

- A: Manifold with two points.
- B: Euclidean distance between the two points.
- C: Geodesic distance between the two points.

Isomap

Instances r and s are connected in the graph if

- \(| x^r - x^s | < \varepsilon \) or if \( x^s \) is one of the \( k \) neighbors of \( x^r \)

The edge length is \(| x^r - x^s | \)

- For two nodes \( r \) and \( s \) not connected, the distance is equal to the shortest path between them

- Once the \( N \times N \) distance matrix is thus formed, use MDS to find a lower-dimensional mapping


Locally Linear Embedding

1. Given \( x^r \) find its neighbors \( x^s_{(r)} \)
2. Find \( W_{rs} \) that minimize

\[
E(W | X) = \sum_r | x^r - \sum_s W_{rs} x^s_{(r)} |^2
\]

3. Find the new coordinates \( z^r \) that minimize

\[
E(z | W) = \sum_r | z^r - \sum_s W_{rs} z^s_{(r)} |^2
\]
LLE on Optdigits


References