Dimensionality Reduction

Farzaneh Mirzazadeh

University of California, Santa Cruz

Winter' 17

Many of the slides are from

- Prof Aarti Singh (CMU)
- Profs Richard Zemel, Raquel Urtasun and Sanja Fidler (University of Toronto)

Unsupervised and representation learning problems

- Clustering
- Coding
- **Dimensionality reduction**

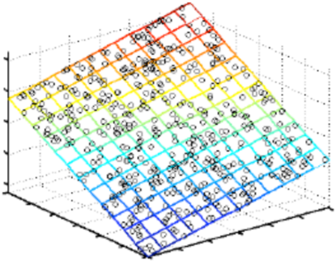
Part 3: Dimensionality Reduction

Dimensionality reduction

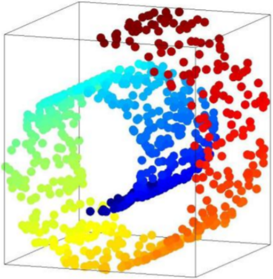
- Goal: Learning a low-dimensional representation from high dimensional data
- Applications
  - Compression of data
  - Visualization (2d or 3d only)
  - Preprocessing data for a supervised task
  - Feature extraction

Dimensionality reduction

Typically used when there is a believe that the data lie near a lower-dimensional manifold



Linear case



Non-linear case

Dimensionality reduction problem

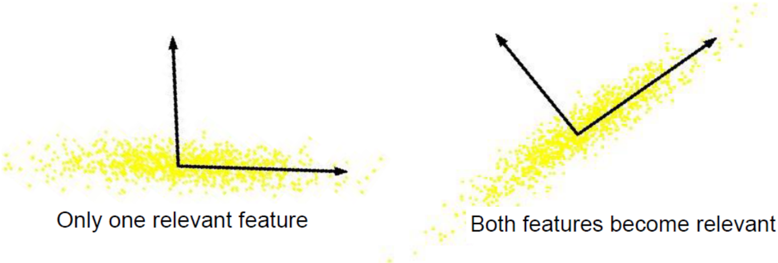
- Input: X is a $t \times n$ matrix of data as before and a dimension d where $d < n$
- Want to learn a map $X \rightarrow \Phi$ that reduces dimension
- Output: A new data matrix Φ with size $t \times d$ matrix

Note: t is the number of examples, n is the number of features

Most popular dimensionality reduction methods:
Principal Component Analysis (PCA)

PCA Idea

Most popular instance of dimensionality reduction algorithms



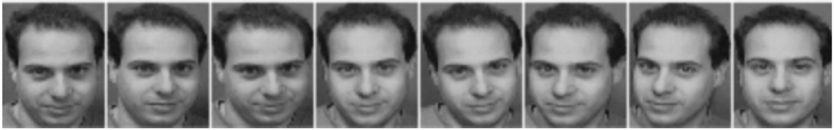
Only one relevant feature

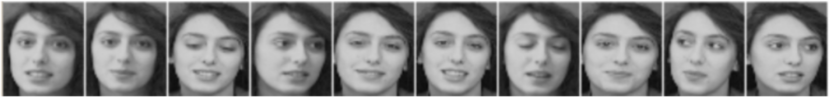
Both features become relevant

- Question that PCA asks: Can we linearly transform data so that smaller number of features become relevant?
- Forms a smaller number of features from linear transforms of the set of original features that can approximate data well: (feature extraction)
- Performs linear dimensionality reduction

PCA in practice

- What are the intrinsic latent dimensions in these two datasets?





- How can we find these dimensions from the data?

What are a small number of eigenfaces?

Two equivalent optimization problems for PCA (primal and dual)

Minimize reconstruction error of rank *d* approximation of data matrix

$$\min_{\Phi, B} \quad \sum_{i=1}^{t} \|X_{i:} - \hat{X}_{i:}\|_2^2 \qquad \hat{X} = \Phi_{t \times d}, V_{d \times n}$$

Finds the optimal rank *d* approximation of matrix *X*.

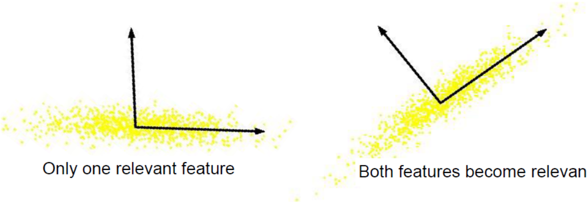
Maximizing variance of projected data into *d* dimensional space

$$\max_{v} v^\top X^\top X v \quad \text{s.t.} \quad \|v\|^2 = 1$$

Finds *d* vectors *v* (size$n \times 1$) such that projections on the vectors capture maximum variance in the data

Maximizing variance interpretation

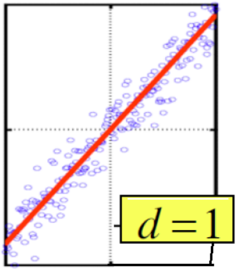
- Aim: Find a small number of directions in input space that explain (most of) variation in input data. Re-represent data by projecting along those direction.
- Important assumption: variation contains information.

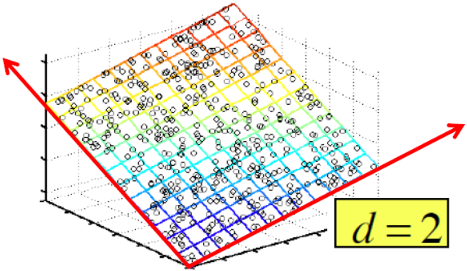


Only one relevant feature

Both features become relevant

Principal Component Analysis (PCA)





Assumption: Data lies on or near a low d-dimensional linear subspace.

Axes of this subspace are an effective representation of the data

Identifying the axes is known as Principal Components Analysis, and can be obtained by Eigen or Singular value decomposition

PCA Algorithm 1

Step 0: Preprocessing

To find the principal component directions, center the data (i.e. subtract the sample mean from each variable) to get a centered matrix X. Here X is $t \times n$ with t examples and n features.

Reason for step 1: Only for centered matrices empirical covariance is computed as the next step.

Step 1

Calculate the empirical covariance matrix of data C as $C = X^\top X$.

Background: Compare empirical covariance matrix $C = X^\top X$ with a kernel matrix $K = XX^\top$. (The order of multiplication is different)

- Both are symmetric positive semidefinite (why?).
- Kernel matrix is a $t \times t$ matrix showing similarity between examples.
- Covariance is an $n \times n$ matrix that shows relation between features.

Step 3

Perform eigenvalue decomposition on the symmetric empirical covariance matrix C from Step 2.

$$C = U \Sigma U^\top$$

Background: Eigenvalue decomposition: Any symmetric matrix C can be decomposed (factored) as product of three matrices $C = U \Sigma U^\top$. Each column of U is an eigenvector of C, Σ is a diagonal matrix, where eigenvalues of C appear on the main diagonal. For a symmetric matrix eigenvalues are real numbers. $UU^\top = I$

Step 4

Find the d eigenvectors with largest eigenvalues of C. Discard the rest. These are principal components. In other words, these are new basis for representation of our data. Form a $d \times n$ matrix U_{reduce} by discarding the rest of eigenvectors and only keeping the top d.

Step 5

Form the optimal factorization of X with Φ $t \times d$:

$$\Phi = XU_{reduce}^{\top}, \qquad V = U_{reduce}, \hat{X} = \Phi V$$

Finding the solution for max variance form

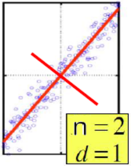
How?

A constrained optimization. Solved with Lagrange multipliers.

Let $v_1, v_2, \ldots, v_d$ denote the principal components

Orthogonal and unit norm $\quad v_i^T v_j = 0 \quad i \neq j$

$v_i^T v_i = 1$



Find vector that maximizes sample variance of projection

$$\frac{1}{n}\sum_{i=1}^{n}(\mathbf{v}^T\mathbf{x}_i)^2 = \mathbf{v}^T\ \mathbf{X}^T\mathbf{X}\ \mathbf{v}$$

Assume data are centered

Data points X

$$\max_{\mathbf{v}}\ \mathbf{v}^T\mathbf{X}^T\mathbf{X}\,\mathbf{v} \quad \text{s.t.} \quad \mathbf{v}^T\mathbf{v} = 1$$

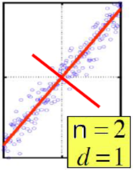
Lagrangian: $\max_{\mathbf{v}} \mathbf{v}^T\mathbf{X}^T\mathbf{X}\mathbf{v} - \lambda\mathbf{v}^T\mathbf{v}$

Wrap constraints into the objective function

$$\partial/\partial\mathbf{v} = 0 \qquad (\mathbf{X}^T\mathbf{X} - \lambda\mathbf{I})\mathbf{v} = 0 \qquad \Rightarrow \boxed{(\mathbf{X}^T\mathbf{X})\mathbf{v} = \lambda\mathbf{v}}$$

Finding the solution for max variance form

$$(\mathbf{X}^T\mathbf{X})\mathbf{v} = \lambda\mathbf{v}$$



Therefore, v is the eigenvector of sample correlation/ covariance matrix XX^T

Sample variance of projection =$\mathbf{v}^T\mathbf{X}^T\mathbf{X}\mathbf{v} = \lambda\mathbf{v}^T\mathbf{v} = \lambda$

Thus, the eigenvalue λ denotes the amount of variability captured along that dimension (aka amount of energy along that dimension).

Eigenvalues $\lambda_1 > \lambda_2 > \lambda_3 > \ldots$

The 1st Principal component v_1 is the eigenvector of the sample covariance matrix $\mathbf{X}^T\mathbf{X}$ associated with the largest eigenvalue λ_1

The 2nd Principal component v_2 is the eigenvector of the sample covariance matrix $\mathbf{X}^T\mathbf{X}$ associated with the second largest eigenvalue λ_2

And so on …

Finding the solution for max variance form

Eigenvectors are solutions of the following equation:

$$(\mathbf{X}^T\mathbf{X})\mathbf{v} = \lambda\mathbf{v} \qquad (\mathbf{X}^T\mathbf{X} - \lambda\mathbf{I})\mathbf{v} = 0$$

Non-zero solution v ≠ 0 possible only if

$$\det(\mathbf{X}^T\mathbf{X} - \lambda\mathbf{I}) = 0$$

Characteristic Equation

This is a Dth order equation in λ, can have at most D distinct solutions (roots of the characteristic equation)

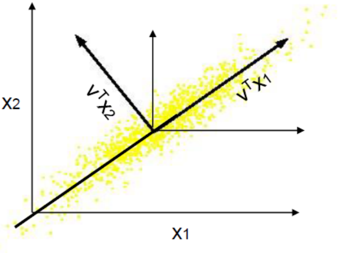
Once eigenvalues are computed, solve for eigenvectors (Principal Components) using

$$(\mathbf{X}^T\mathbf{X} - \lambda\mathbf{I})\mathbf{v} = 0$$

For symmetric matrices, eigenvectors for distinct eigenvalues are orthogonal.

So, the new axes are the eigenvectors of the matrix of sample correlations $\mathbf{X}^T\mathbf{X}$ of the data, which capture the similarities of the original features based on how data samples project to the new axes.

Transformed features are uncorrelated.



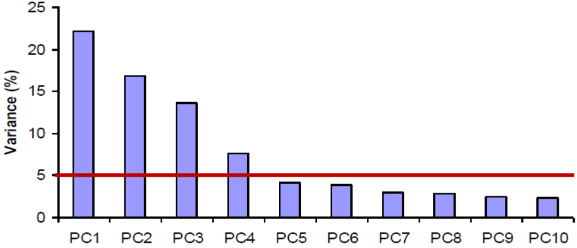
- Geometrically: centering followed by rotation
  - Linear transformation

Dimensionality Reduction using PCA

In high-dimensional problem, data usually lies near a linear subspace, as noise introduces small variability

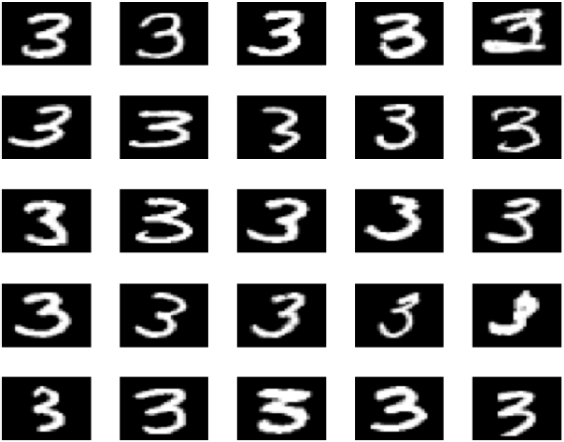
Only keep data projections onto principal components with **large** eigenvalues

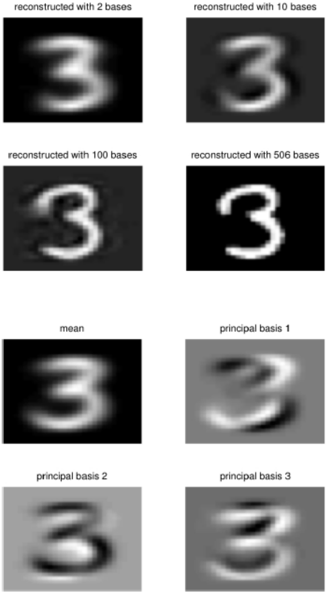
Can *ignore* the components of lesser significance.



You might lose some information, but if the eigenvalues are small, you don't lose much

Example reconstruction





reconstructed with 2 bases

reconstructed with 10 bases

reconstructed with 100 bases

reconstructed with 506 bases

mean

principal basis 1

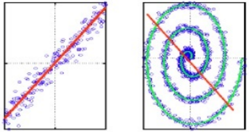
principal basis 2

principal basis 3

Use eig() in Matlab for eigen decomposition.
Use svd() in Matlab for singular value decomposition.

Properties of PCA

- Strengths
  - Eigenvector method
  - No tuning parameters
  - Non-iterative
  - No local optima



- Weaknesses
  - Limited to linear projections