Dimensionality Reduction with PCA

Ke Tran

May 24, 2011

Why dimensionality reduction?

- To discover or to reduce the dimensionality of the data set.

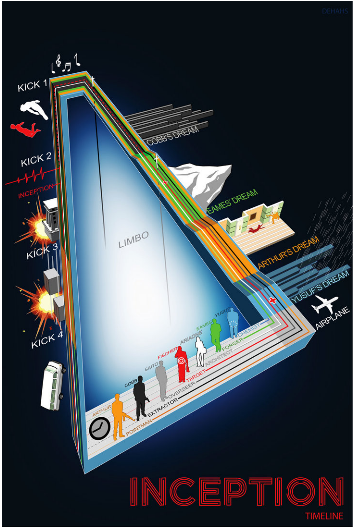
Why dimensionality reduction?

- To discover or to reduce the dimensionality of the data set.
- To identify new meaningful underlying variables.

Why dimensionality reduction?

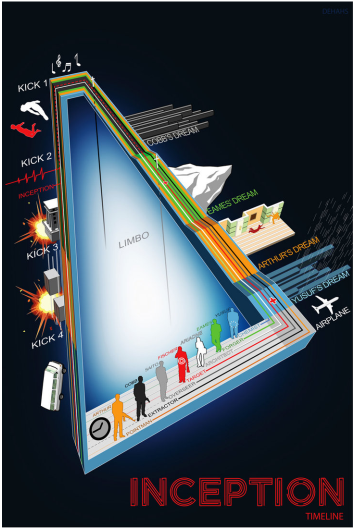
- To discover or to reduce the dimensionality of the data set.
- To identify new meaningful underlying variables.
- Curse of dimensionality: some problems become intractable as the number of the variables increases.

Why dimensionality reduction?



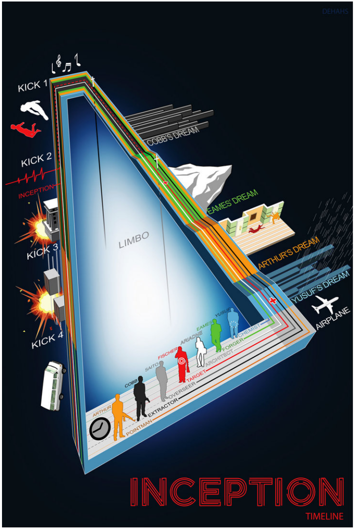
- To reason about or obtain insights from.

Why dimensionality reduction?



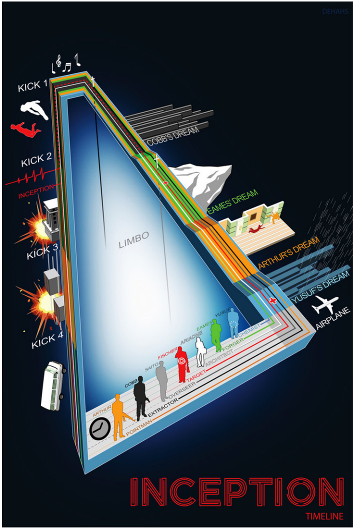
- To reason about or obtain insights from.
- Too much noise in the data.

Why dimensionality reduction?



- To reason about or obtain insights from.
- Too much noise in the data.
- To visualize

Why dimensionality reduction?



- To reason about or obtain insights from.
- Too much noise in the data.
- To visualize
- Can build more effective data analyses on the reduced-dimensional space: classification, clustering, pattern recognition.

PCA - Basic Idea

- Projection
- Can be used to determine how many real dimensions there are in the data.

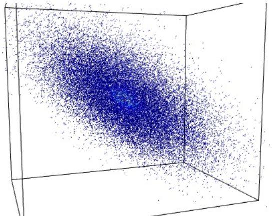


Figure: The data forms a cluster of points in a 3D space

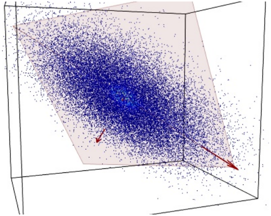


Figure: The covariance eigenvectors identified by PCA are shown in red. The plane defined by the 2 largest eigenvectors is shown in light red.

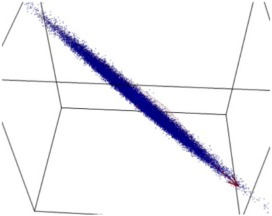


Figure: If we look at the data in the plane identified by PCA, it is clear that it was mostly 2D

Linear Transformation

1. Let X be the original data set, where each column is a single sample, X is an $m \times n$ matrix
2. Let Y be another $m \times n$ matrix related by a linear transformation P
3. X is the original recorded data set and Y is a new representation of that data set.

$$PX = \begin{bmatrix} p_1 \\ \vdots \\ p_m \end{bmatrix} [x_1 \cdots x_n] \ Y = \begin{bmatrix} p_1 x_1 & \cdots & p_1 x_n \\ \vdots & \ddots & \vdots \\ p_m x_1 & \cdots & p_m x_n \end{bmatrix}$$

1. What is the best way to re-express X?
2. What is a good choice of basis P?

Variance and the Goal

What is the best way to re-express X?

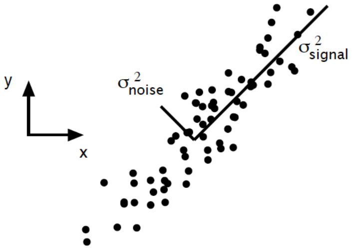


Figure: The signal and noise variances σ^2_{signal} and σ^2_{noise} are graphically represented by the two lines subtending the cloud of data

- Signal-to-noise ratio (SNR)
- $SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}}$
- A high SNR indicates a high precision measurement, while a low SNR indicates very noisy data.

Variance and the Goal

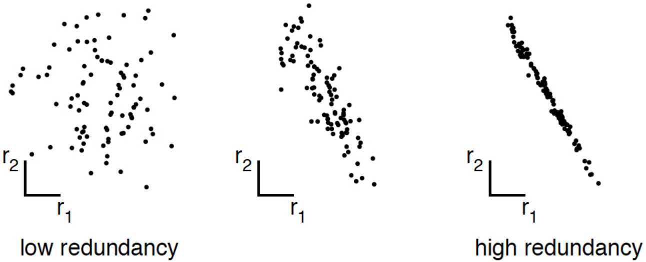


Figure: A spectrum of possible redundancies in data from the two separate measurements r_1 and r_2. The two measurements on the left are uncorrelated because one can not predict one from the other. Conversely, the two measurements on the right are highly correlated indicating highly redundant measurements.

Assumption behind PCA

1. Linearity
2. Large variances have important structure.
3. The principal components are orthogonal: $p_i \times p_j = 0$

PCA algorithm

1. Select a normalized direction in m-dimensional space along which the variance in X is maximized. Save this vector as p_1.
2. Find another direction along which variance is maximized, however, because of the orthonormality condition, restrict the search to all directions orthogonal to all previous selected directions. Save this vector as p_i
3. Repeat this procedure until m vectors are selected.

The resulting ordered set of p's are the principal components.

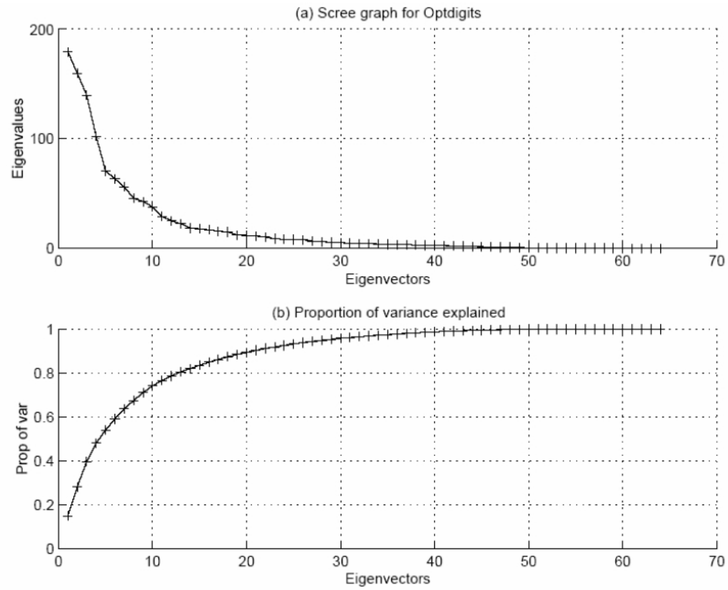
PCA algorithm - Computational Trick

1. Compute covariance matrix C_x, $C_X \equiv \frac{1}{n}XX^T$
2. We select the matrix P to be a matrix where each row p_i is an eigenvector of $\frac{1}{n}XX^T$
3. If A is a square matrix, a non-zero vector v is an eigenvector of A if there is a scalar λ such that $Av = \lambda v$
4. Reduction: there are m eigenvectors, we reduce from m dimensions to k dimensions by choosing k eigenvectors related with k largest eigenvalues λ

How to choose k?

1. Proportion of Variance (PoV) explained
   $\frac{\lambda_1+\lambda_2+\cdots\lambda_k}{\lambda_1+\lambda_2+\cdots\lambda_k+\cdots+\lambda_m}$
   when λ_i are sorted in descending order.
2. Typically, stop at $PoV \geq 0.9$
3. Scree graph plots of PoV vs k, stop at "elbow"

Scree graph



(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

Yale Faces B - The Dataset

standard face recognition test data set containing

- 10 subjects
- in 585 different positions and lighting conditions each

  $\rightarrow$ database of 5850 images
- representation: Matlab type
- image dimension: 30 x 40

  $\rightarrow$ image representation: 1200-dimensional vector
- split randomly into training and test set

Yale Faces B - The Dataset

Figure: Yale Faces B: selected sample pictures for one test person

SVM experiment

Procedure:

- again: random division of Yale Faces B into training and test set:
  - training set: 1800 images of 10 classes
  - test set: 4050 (remaining) images
    $\rightarrow$ 30 % training, 70 % testing

Eigenfaces or Eigenvectors

1. Using PCA reduces to 10 dimensions
2. Classification with SVM: 97,5 % correctness

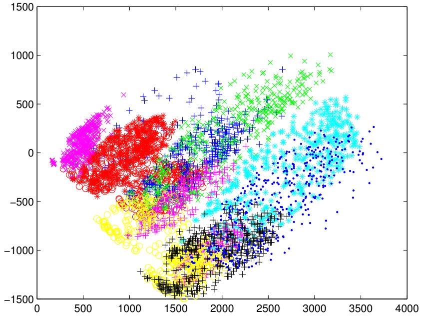
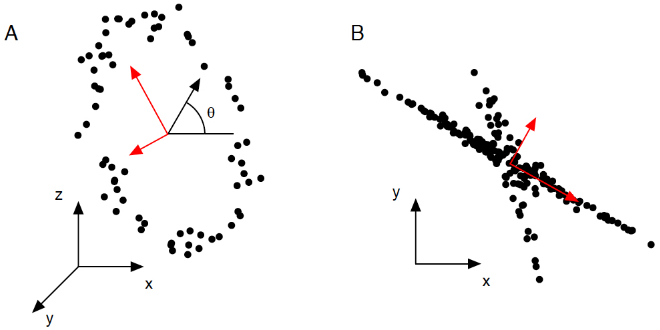


Figure: 2-D Visualization of data encoded into Eigenfaces

When does PCA fail?



- non-linear data
- non Gaussian distribution
- variance due to error

PCA or not?

1. depend on the problem
2. depend on computational resource
3. there are many better methods for dimensionality reduction

PCA: 97,5 % correctness

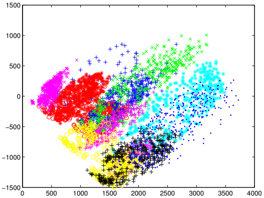


Figure: Visualization of 2-D **projection onto Eigenfaces** showing linear separability

PCA or not?

1. depend on the problem
2. depend on computational resource
3. there are many better methods for dimensionality reduction

PCA: 97,5 % correctness

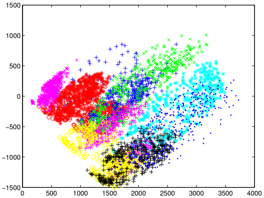


Figure: Visualization of 2-D **projection onto Eigenfaces** showing linear separability

Autoencoder: 99,8 % correctness

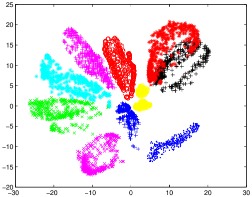


Figure: Comparison: Visualization of 2-D **autoencoded data** showing better linear separability

Questions?

Questions?

Thanks!