Principal Component Analysis

Machine Learning
Summer 2015

Dr. Joschka Boedecker

Acknowledgement
Slides courtesy of Manuel Blum
Motivation

dimensionality reduction transforms a $n$-dimensional dataset to a $k$-dimensional dataset with $k < n$

- dataset compression
  - less memory storage consumption
  - machine learning algorithms run faster on low-dimensional data

- data visualization
  - high-dimensional data can be transformed to 2D or 3D for plotting
Principal Component Analysis

- most commonly used dimensionality reduction method
- projects the data on $k$ orthogonal bases vectors $u$ that minimize the projection error

Example:

- original 2D dataset containing features weight and height
- projection on vector $u$
PCA Algorithm

input: $x^{(1)}$, $x^{(2)}$, ..., $x^{(m)}$

preprocessing:
- mean normalization
  1. compute mean of each feature $j$
     \[
     \mu_j = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)}
     \]
  2. subtract the mean from data
     \[
     x_j^{(i)} \leftarrow x_j^{(i)} - \mu_j
     \]
- feature scaling
  \[
  x_j^{(i)} \leftarrow a_j x_j^{(i)}
  \]
PCA Algorithm

compute covariance matrix  \[ \Sigma = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} x^{(i)T} \]

diagonalize covariance matrix (using SVD)
\[ S = U^{-1} \Sigma U \]

\( U \) is the matrix of Eigenvectors
\( S \) a diagonal matrix containing the Eigenvalues

dimensionality reduction from \( n \) to \( k \) dimensions:
project the data onto the Eigenvectors corresponding to the \( k \) largest Eigenvalues

\[ z^{(i)} = U^T_{reduce} x^{(i)} \]
Reconstruction

\[ z(i) = U^T_{reduce} x^{(i)} \]

\[ x_{approx} = U_{reduce} \star z^{(i)} \]

the reconstruction of compressed data points is an approximation of the original data
Choosing $k$

average squared projection error:

$$\frac{1}{m} \sum_{i=1}^{m} \left\| x(i) - x_{approx} \right\|^2$$

total variation in the data:

$$\frac{1}{m} \sum_{i=1}^{m} \left\| x(i) \right\|^2$$

to retain 99% of the variance, choose $k$ to be the smallest value, such that

$$\frac{1}{m} \sum_{i=1}^{m} \left\| x(i) - x_{approx} \right\|^2 \leq 0.01$$

$$\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \geq 0.99$$
Example using Real-world Data

http://archive.ics.uci.edu/ml/

- offers 223 datasets
- datasets can be used for the evaluation of ML methods
- results can be compared to those of other researchers
**Iris Data Set**

*Download: Data Folder, Data Set Description*

**Abstract:** Famous database; from Fisher, 1936

<table>
<thead>
<tr>
<th>Data Set Characteristics:</th>
<th>Multivariate</th>
<th>Number of Instances:</th>
<th>150</th>
<th>Area:</th>
<th>Life</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Characteristics:</td>
<td>Real</td>
<td>Number of Attributes:</td>
<td>4</td>
<td>Date Donated</td>
<td>1988-07-01</td>
</tr>
<tr>
<td>Associated Tasks:</td>
<td>Classification</td>
<td>Missing Values?</td>
<td>No</td>
<td>Number of Web Hits:</td>
<td>348488</td>
</tr>
</tbody>
</table>

**Source:**

Creator: R.A. Fisher

Donor: Michael Marshall (MARSHALL%PLU '@' io.arc.nasa.gov)

**Data Set Information:**

This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. (See Duda & Hart, for example.) The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

Predicted attribute: class of iris plant.

This is an exceedingly simple domain.

This data differs from the data presented in Fishers article (identified by Steve Chadwick, spchadwick '@' espeedaz.net ). The 35th sample should be: 4.9,3.1,1.5,0.2,"Iris-setosa" where the error is in the fourth feature. The 38th sample: 4.9,3.6,1.4,0.1,"Iris-setosa" where the errors are in the second and third features.

**Attribute Information:**

1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm
5. class: Iris Setosa, Iris Versicolour, Iris Virginica
PCA on the Iris dataset

given: data matrix $X$

preprocessing:
- mean normalization
- feature scaling

compute covariance matrix:

$$
\Sigma = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} x^{(i)T}
$$

compute eigenvectors and eigenvalues:

$$
U = \begin{pmatrix}
-0.5224 & -0.3723 & 0.7210 & 0.2620 \\
0.2634 & -0.9256 & -0.2420 & -0.1241 \\
-0.5813 & -0.0211 & -0.1409 & -0.8012 \\
-0.5656 & -0.0654 & -0.6338 & 0.5235
\end{pmatrix}
$$

$$
S = \begin{pmatrix}
2.8914 & 0 & 0 & 0 & 0 \\
0 & 0.9151 & 0 & 0 & 0 \\
0 & 0 & 0.1464 & 0 & 0 \\
0 & 0 & 0 & 0.0205
\end{pmatrix}
$$

reduce $U$ to $k$ components

$$
\tilde{x}^{(i)} = U_{\text{reduce}}^T x^{(i)}
$$
Final Remarks

- PCA can only realize linear transformations
- there exist nonlinear extensions (Kernel PCA)
- PCA-transformed data is uncorrelated
- PCA assumes that most of the information is contained in the direction with the highest variance
- PCA is often used to reduce the noise in a signal
- PCA is an unsupervised method - when used as a preprocessing step for supervised learning the performance can drop significantly