主成分分析 (Principle Component Analysis,PCA)

Jingbo Xia

Huazhong Agricultural University

xiajingbo.math@gmail.com

2026-03-06

开宗明义 I

- 对矩阵的操作必要性在于数据本身的矩阵存储形式。
- PCA同样是对存储数据的矩阵进行操作。我们介绍该算法的两个应用——第一是数据的降维，第二是数据的重构。
- 这个教学材料中包括一个图片处理的示例，它能帮助同学们较为生动地理解PCA算法中的“维”。

Table of contents I

Outline

主成分分析/Principal Component Analysis

投影/Projection!

The data is linearly transformed onto a new coordinate system such that the directions (**principal component（主成分）)** capturing the largest variation in the data can be easily identified.
The principal components of a collection of points in a real coordinate space are a sequence of J unit vectors, where the j-th vector is the direction of a line that best fits the data while being orthogonal to the first $j-1$ vectors. "[a].

[a] https://en.wikipedia.org/wiki/Principal_component_analysis

要素：

- Data: $\{X_i\}$, a set of observations（观测） of possibly correlated variables, $X_i \in R^d, \; i = 1, 2, ..., n.$
- A real coordinate space: $V_j \in R^d$, that "captures the largest variation in the data". V_j is orthogonal to the first $j-1$ vectors , $j = 1, 2, ..., J$.

主成分分析/Principal Component Analysis

投影/Projection!

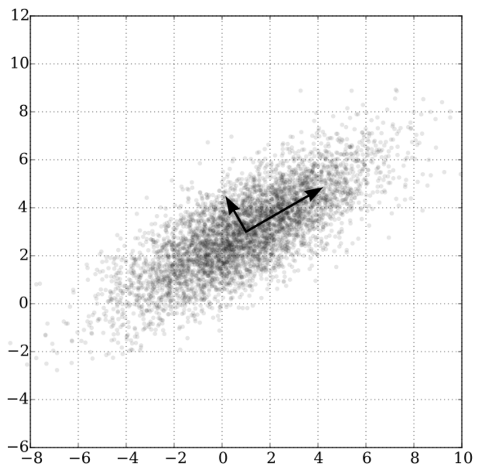


图 1: PCA of a multivariate Gaussian distribution（多维高斯分布） centered at $(1, 3)$ with a standard deviation（标准差） of 3 in roughly the $(0.866, 0.5)$ direction and of 1 in the orthogonal direction. The vectors shown are the eigenvectors of the covariance matrix（协方差矩阵） scaled by the square root（平方根） of the corresponding eigenvalue, and shifted so their tails are at the mean.

主成分分析/Principal Component Analysis

投影/Projection!

如何选取投影向量？

As shown in figure 2, different selection of projection line leads to difference in the variance. It's no hard to observe that the left projection brings greater variance than the right projection does.

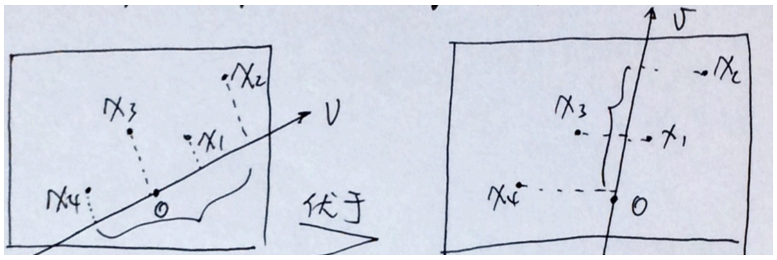


图 2: An example of four samples/points which are projected to different vector/directions.

Then, how to calculate the projection coordinates（投影坐标）?

主成分分析/Principal Component Analysis
模型求解

计算准备：向量α在向量β上投影。

Let's discuss it in a $\mathbb{R}^n$ space. If α project to β, then the projection[1] is

$$\frac{(\alpha,\beta)}{(\beta,\beta)}\beta = \frac{(\alpha,\beta)}{|\beta|}\frac{\beta}{|\beta|},$$

while the projection coordinate[2] in β is

$$\frac{(\alpha,\beta)}{|\beta|}.$$

Without the loss of the generality（不失一般性）, we assume $|\beta| = 1$, and then the projection coordinate is (α,β).

[1]Note: This is a vector, and this vector is with the same/ opposite direction of β
[2]Note: This is a scalar value.

主成分分析/Principal Component Analysis I
模型求解

To obtain an optimized V^* w.r.t. the maximization of the variance (方差) of the projection coordinate.

$$V^* = \underset{V \in R^d}{\arg\max}\{\sigma^2\}.$$

Now let us discuss the variance. Assume there are l samples $X_i \in \mathbb{R}^d$, $(i = 1, 2, \cdots, l)$, and V is a vector to which X_i projects.

For simplicity, we assume that the mean of samples $\bar{X} = 0$. Denote σ^2 as the variance of all of the projection coordinates.

$$\sigma^2 = ?$$

主成分分析/Principal Component Analysis II
模型求解

Compute σ^2.

$$\begin{aligned}\sigma^2 &= \frac{1}{l}\sum_{i=1}^{l}(V^T X_i - 0)^2 = \frac{1}{l}\sum_{i=1}^{l}(V^T X_i)(V^T X_i)^T \\ &= \frac{1}{l}\sum_{i=1}^{l} V^T X_i X_i^T V = V^T(\frac{1}{l}\sum_{i=1}^{l} X_i X_i^T)V \\ &:= V^T C V\end{aligned} \quad (1)$$

Note 1: C is a covariance matrix in a zero-mean case.
Note 2: 把关于V的目标函数整理成上述表达式有何益处?

主成分分析/Principal Component Analysis III

模型求解

Note: The general form of the covariance matrix（协方差矩阵）.

As a general form, i.e., if we remove the assumption that the mean of X_i equals to zero, the above deduction leads to:

$$\begin{aligned} C &= \frac{1}{l}\sum_{i=1}^{l}(X_i - \bar{X})(X_i - \bar{X})^T. \\ &= E[(X-\mu)(X-\mu)^T] \qquad (\mu = E[X]) \end{aligned} \tag{2}$$

Here, C is the covariance matrix.

主成分分析/Principal Component Analysis IV

模型求解

Solve PCA model. Computation steps based on optimization theory and matrix calculus.

The optimization problem with PCA is

$$V = \underset{V\in\mathbb{R}^d, |V|=1}{\arg\max}\ V^T C V \tag{3}$$

The Lagrangean function is

$$f(v,\lambda) = V^T C V - \lambda(V^T V - 1).$$

——————计算环节：$\frac{\partial f}{\partial V} = 0$.——————

主成分分析/Principal Component Analysis V

模型求解

寻优的过程，等同于对协方差矩阵C的特征向量求解。

By solving $\frac{\partial f}{\partial V} = 0$, we have

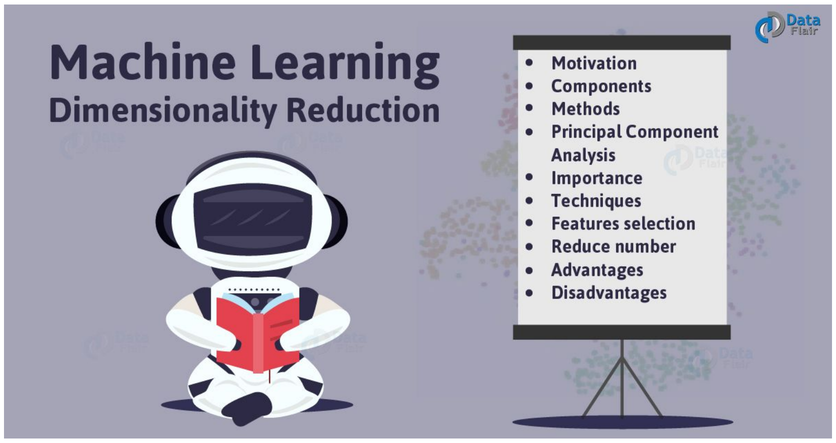
$$CV = \lambda V.$$

This means that V, as a projection vector, is the eigenvector of the covariance matrix.

主成分分析/Principal Component Analysis

特征值排序带来的应用

Sorting of the eigenvalues (or singular values) leads to two main application scenarios: feature reduction（特征约简）, or dimentionality reduction（维度约简）[3],

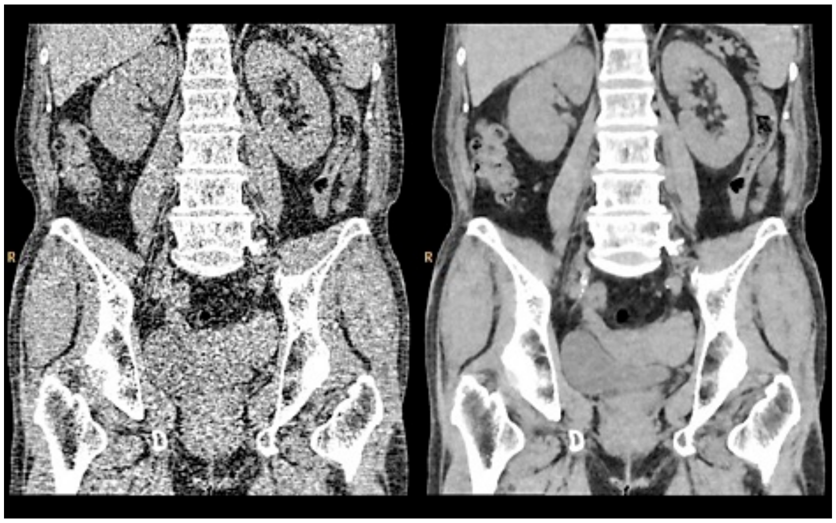


[3]https://data-flair.training/blogs/dimensionality-reduction-tutorial/.

主成分分析/Principal Component Analysis

特征值排序带来的应用

and data reconstruction（数据重构）[4].



[4]https://www.usa.philips.com/healthcare/product/HCNCTD449/iterative-model-reconstruction-reconstruction-technology

主成分分析/Principal Component Analysis I

特征值排序带来的应用

对特征值λ_i进行排序，依次获得不同的投影向量V_i。

By sorting the singular values, or eigenvalues, with a descending order, one can select different V_j, and project $X_i \in \mathbb{R}^d$ to different V_j and obtain corresponding projection coordinate $(X_i, V_j) = X_i^T V_j$.

主成分分析/Principal Component Analysis II

特征值排序带来的应用

Feature reduction（特征约简）

Let's fix the first J V_j with greatest eigenvalues, and replace X_i with a J-tuple vector, $\hat{X}_i = (X_i^T V_1, X_i^T V_2, \cdots, X_i^T V_J)^T$. [a]

[a] We now convert the d-tuple vector X_i into a J-tuple one, i.e., $\hat{X}_i$, thus reducing the dimentionality of the data.

Note: 特征约简，是不是就是特征选择？

主成分分析/Principal Component Analysis III

特征值排序带来的应用

Data reconstruction（数据重构）

If we represented the sample X_i with an approximation, $\tilde{X}_i = (X_i^T V_1)V_1 + (X_i^T V_2)V_2 + \cdots + (X_i^T V_J)V_J$. [a]

[a] Here, $\tilde{X}_i$ is a low-rank approximate of X_i, which reconstructs the original data.

Note 1：上述公式中，何谓“主成分”，或者“主成分向量”？
Note 2: 数据重构，还有哪些类似算法？

Outline

PCA的若干有趣阐释 I

SVD on PCA

SVD and principle component analysis have close relevance.
Assume the matrix $A \in \mathbb{R}^{m\times n}$ contains m "n-tuple" sample vectors A_i $(i = 1, 2, .., \cdots, m)$, i.e.,

$$A = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_m \end{pmatrix}, \tag{4}$$

When performing low-rank approximation, the purpose of PCA is to find several (let's say r) "n-tuple" principle components $V_1^T, \cdots, V_r^T$.

$V_j\ (j = 1, 2, \cdots, r)$ are the eigenvectors of co-variance matrix A^TA, s.t.,

$$A_i \approx b_{i1}V_1^T + b_{i2}V_2 + \cdots + b_{ir}V_r^T.$$

PCA的若干有趣阐释 II

SVD on PCA

To put all A_i into A, we have
$$A = BV^T.$$

Interestingly, SVD of A provides solution of PCA:
$$A = U\Sigma V^T.$$

By collecting $U\Sigma$ and V, SVD provides B and V.

PCA的若干有趣阐释 I

An example to visualize the principle component

Generally speaking, it is not that direct to figure out how A_i and V_j look like. A nice visualization example is recommended here, http://www.infoq.com/cn/articles/matrix-decomposition-of-recommend-system.

In this example, A_i is a 4096-tuple vector which presents a picture, and V_js are so-called "creepy faces" which represent different characteristics of faces. Here, these weighted sum of "creepy faces" reconstruct A_I.

PCA的若干有趣阐释 II

An example to visualize the principle component

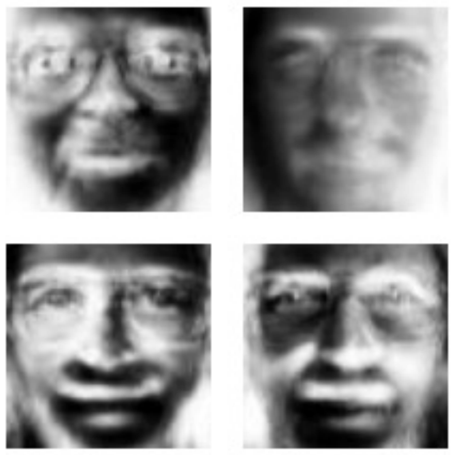
Each A_i is a 4096-tuple vector, which represents a face figure.

Each A_i is approximated by a weighted sum of the below "creepy faces":

PCA的若干有趣阐释 III

An example to visualize the principle component

V_j^T, "creepy faces":



PCA的若干有趣阐释

An example to visualize the principle component

例 (Python Codes)

```
from sklearn.datasets import fetch_olivetti_faces
from sklearn.decomposition import PCA

import matplotlib.pyplot as plt
%matplotlib inline
faces = fetch_olivetti_faces()
print(faces.DESCR)
```

There are ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement).

PCA的若干有趣阐释

An example to visualize the principle component

例 (Python Codes)

```
# Here are the first ten guys of the dataset
fig = plt.figure(figsize=(10, 10))
for i in range(5):
    ax = plt.subplot2grid((1, 5), (0, i))
    ax.imshow(faces.data[i * 10].reshape(64, 64), cmap=plt.cm.gray)
    ax.axis('off')

# Let's compute the PCA
pca = PCA()
pca.fit(faces.data)
```

PCA的若干有趣阐释

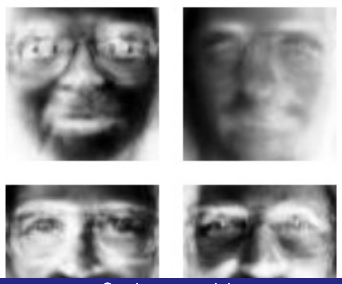
An example to visualize the principle component

例 (Python Codes)

```
# Now, the creepy guys are in the components_ attribute.
# Here are the first four ones:

fig = plt.figure(figsize=(10, 10))
for i in range(4):
    ax = plt.subplot2grid((2, 2), (0, i))

    ax.imshow(pca.components_[i].reshape(64, 64), cmap=plt.cm.gray)
    ax.axis('off')
```



PCA的若干有趣阐释

例 (Python Codes)

```
# Reconstruction process

from skimage.io import imsave
face = faces.data[0]  # we will reconstruct the first face

# During the reconstruction process we are actually computing,
# at the kth frame, a rank k approximation of the face.
# To get a rank k approximation of a face,
# we need to first transform it into the 'latent space', and then
# transform it back to the original space
# Step 1: transform the face into the latent space.
# It's now a vector with 400 components. The kth component gives
# the importance of the kth  creepy guy

trans = pca.transform(face.reshape(1, -1))
# Reshape for scikit learn
```

PCA的若干有趣阐释

An example to visualize the principle component

例 (Python Codes)

```
# Step 2: reconstruction. To build the kth frame, we use all the cre
# up until the kth one.
# Warning: this will save 400 png images.

for k in range(400):
    rank_k_approx = trans[:, :k].dot(pca.components_[:k])
            + pca.mean_
    imsave('{:>03}'.format(str(k))
            + '.jpg', rank_k_approx.reshape(64, 64))
```