Data Analysis Lab

Component Analysis Dimensionality Reduction

Fátima Leal

DCT DEPARTAMENTO CIÊNCIA E TECNOLOGIA

UPT UNIVERSIDADE PORTUCALENSE

Outline

- Component Analysis
- Dimensionality reduction
- Example

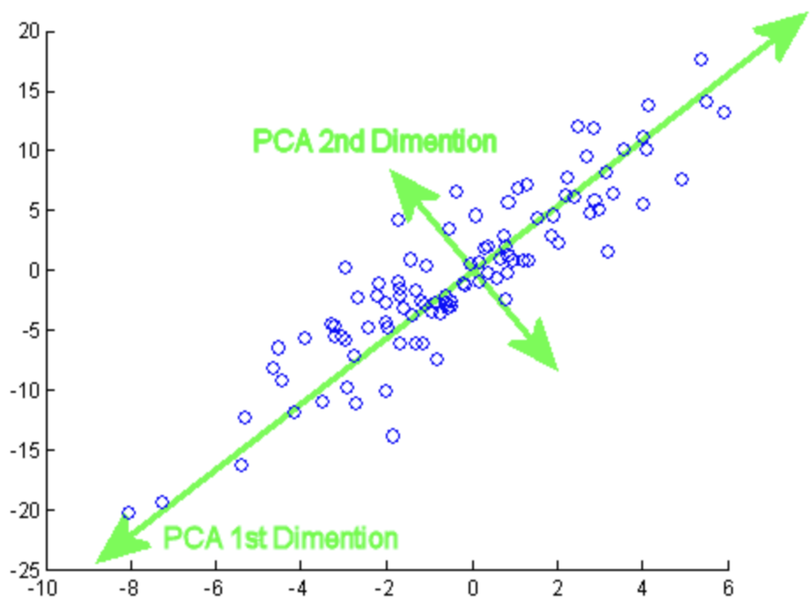
Principal Component Analysis

- A common approach to the problem of reducing the dimensionality of a high-dimensional dataset is based on the assumption that, normally, the total variance is not explained equally by all components. If p_{data} is a multivariate Gaussian distribution with covariance matrix $\sum$, then the entropy (which is a measure of the amount of information contained in the distribution) is as follows: $H(p) = \frac{1}{2} logdet2\pi e \sum$
- Therefore, if some components have a very low variance, they also have a limited contribution to the entropy, and provide little additional information. Hence, they can be removed without a high loss of accuracy.

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

Principal Component Analysis

- Unsupervised dimensionality reduction technique
- cluster the similar data points based on the feature correlation between them without any supervision (or labels)



Principal Component Analysis

- Just as we've done with factor analysis, let's consider a dataset drawn from $p_{data} \approx N(0, \sum)$: $X = \{\bar{x}_1, \bar{x}_2, ..., \bar{x}_M\}$ where $\bar{x}_i \in \mathrm{R}^n$
- Our goal is to define a linear transformation, $\bar{z} = A^T \bar{x}$ (a vector is normally considered a column, therefore, $\bar{x}$ as a shape $(nx1)$), such as the following: $dim \bar{z}_i << n$ and $H(p(\bar{z})) \approx H(p(\bar{x}))$
- As we want to find out the directions where the variance is higher, we can build our transformation matrix A, starting from the eigen decomposition of the input covariance matrix $\sum$ (which is real, symmetric, and positive definite): $\sum = V \Omega V^T$
- V is an (nxn) matrix containing the eigenvectors (as columns), while Ω is a diagonal matrix containing the eigenvalues. Moreover, V is also orthogonal, hence the eigenvectors constitute a basis.

Where yo apply PCA?

- Data visualisation
- Speeding Up a Machine Learning (ML) Algorithm

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

```
from sklearn.datasets import load_breast_cancer
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA

breast = load_breast_cancer()
breast_data = breast.data
breast_labels = breast.target

scaler = StandardScaler()
scaler.fit(breast_data)
scaled_data =scaler.transform(breast_data)


pca = PCA(n_components=2)
pca.fit(scaled_data)
x_pca = pca.transform(scaled_data)
print(scaled_data.shape)
print(x_pca.shape)

plt.figure(figsize=(8,6))
plt.scatter(x_pca[:,0],x_pca[:,1],c=breast_labels,cmap='prism')
plt.xlabel('First Principle Component')
plt.ylabel('Second Principle Component')
plt.show()
```

Principal Component Analysis

- An alternative approach is based on the Singular Value Decomposition (SVD), which has an incremental variant. There are also algorithms that can perform a decomposition truncated at an arbitrary number of components, speeding up the convergence process. In this case, it's immediately noticeable that the sample covariance is as follows (if $M >> 1, M \approx M - 1$) the estimation is almost unbiased): $\sum_S = \frac{1}{M} X^T X$ where $X \in \mathrm{R}^{M*n}$ and $\sum_S \in \mathrm{R}^{n*n}$

- If we apply the SVD to the matrix X (each row represents a single data point with a shape $(1xn)$), we obtain the following: $X = U \Lambda V^T$ where $U \in \mathrm{R}^{M*M}$, $\Lambda \in diag(nxn)$ and $V \in \mathrm{R}^{M*M}$

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

```python
from sklearn.datasets import load_iris
from sklearn.decomposition import TruncatedSVD

iris = load_iris()
iris_data = iris.data
iris_target = iris.target
svd = TruncatedSVD(n_components=2)
iris_transformed = svd.fit_transform(iris_data)


print(iris_data)
print(iris_transformed)
```

Exercise

- Pick one dataset and apply:
- A regressor or classifier without and with dimensionality reduction
- See the diferences in accuracy and time

References

- Bonaccorso, G. (2020). Mastering Machine Learning Algorithms: Expert techniques for implementing popular machine learning algorithms, fine-tuning your models, and understanding how they work. Packt Publishing Ltd.
- Bonaccorso, G. (2018). Machine Learning Algorithms: Popular algorithms for data science and machine learning. Packt Publishing Ltd.
- Lee, W. M. (2019). Python machine learning. John Wiley & Sons.
- Hauck, T. (2014). scikit-learn Cookbook. Packt Publishing Ltd.
- Montgomery, D. C., Peck, E. A., & Vining, G. G. (2021). Introduction to linear regression analysis. John Wiley & Sons.
- Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression (Vol. 398). John Wiley & Sons.

UNIVERSIDADE
PORTUCALENSE

Do conhecimento à prática.