**Step by Step of Principal Component Analysis**

[[Everton Gomede, PhD](https://medium.com/@evertongomede?source=post_page-----d406f6e57ec4--------------------------------)](https://medium.com/@evertongomede?source=post_page-----d406f6e57ec4--------------------------------)

[Everton Gomede, PhD](https://medium.com/@evertongomede?source=post_page-----d406f6e57ec4--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2Fb1a64eb107f0&operation=register&redirect=https%3A%2F%2Fmedium.com%2F%40evertongomede%2Fstep-by-step-of-principal-component-analysis-d406f6e57ec4&user=Everton+Gomede%2C+PhD&userId=b1a64eb107f0&source=post_page-b1a64eb107f0----d406f6e57ec4---------------------post_header-----------)

7 min read

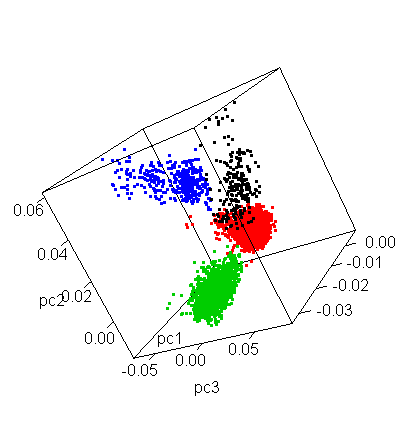
·

Nov 7

44

**Introduction**

In the realm of data analysis and machine learning, the challenge of navigating high-dimensional datasets is a ubiquitous and formidable one. Whether it be in fields such as image processing, genetics, or finance, the curse of dimensionality poses significant obstacles to comprehension, visualization, and modeling. This is where Principal Component Analysis (PCA) steps in as a powerful technique, offering a systematic approach to reduce dimensionality while preserving essential information.



PCA is not merely a mathematical procedure; it is a lens through which we can perceive complex datasets in a more accessible manner. This essay delves into the fundamental principles and practical steps of PCA, guiding the reader through the process of transforming high-dimensional data into a simpler, more interpretable form. By the end of this exploration, you will have a comprehensive understanding of how PCA works and how it benefits the analysis of multidimensional data. We will begin by elucidating the key steps involved in PCA, followed by an explanation of its applications and advantages in various domains.

*Step by step, Principal Component Analysis unveils the hidden layers of data complexity, simplifying the intricate to reveal the essential.*

**Algorithm**

Principal Component Analysis (PCA) is a widely used technique in data analysis and dimensionality reduction. It helps in identifying the most significant patterns and reducing the complexity of high-dimensional data while preserving its essential information. This essay will explain the steps involved in performing PCA:

**Step 1**: Data Collection and Standardization Before applying PCA, gather your data. Ensure that your data is numeric, as PCA is primarily suited for numerical data. If your data has categorical variables, you may need to preprocess them.

Next, standardize the data. Standardization is important because PCA is sensitive to the scales of variables. Standardization transforms the data so that each variable has a mean of 0 and a standard deviation of 1. This ensures that all variables are on the same scale.

**Step 2**: Covariance Matrix Calculation The first step in PCA is to compute the covariance matrix of the standardized data. The covariance matrix represents the relationships between variables. Each element in the matrix represents the covariance between two variables.

The formula for the covariance between two variables X and Y is:



Where:

* *N* is the number of data points.
* *Xi*​ and *Yi*​ are individual data points.
* ˉ*X*ˉ and ˉ*Y*ˉ are the means of variables X and Y, respectively.

The covariance matrix is a square matrix, with each element representing the covariance between two variables.

**Step 3:** Eigenvalue and Eigenvector Computation After computing the covariance matrix, the next step is to find the eigenvalues and eigenvectors of this matrix. These are crucial in determining the principal components.

Eigenvalues (*λ*) and eigenvectors (*v*) are obtained by solving the following equation:



In this equation, *λ* represents the eigenvalue, and *v* represents the eigenvector. You’ll have as many eigenvalues and eigenvectors as the number of variables in your data.

**Step 4**: Sorting and Selecting Principal Components The eigenvalues represent the amount of variance in the data that each eigenvector explains. To reduce the dimensionality, sort the eigenvalues in descending order. The eigenvector corresponding to the largest eigenvalue explains the most variance and is the first principal component. The second largest eigenvalue corresponds to the second principal component, and so on.

Typically, you’ll select a subset of the top eigenvalues/eigenvectors that explain most of the variance in the data while reducing the dimensionality. You can decide on the number of principal components to keep based on a variance explained threshold (e.g., 95% of the total variance).

**Step 5**: Data Transformation To reduce the dimensionality of your data, create a projection matrix using the selected eigenvectors (principal components). This matrix represents the transformation needed to project the data into the new reduced-dimensional space. You multiply your standardized data by this projection matrix to obtain the new data in the principal component space.

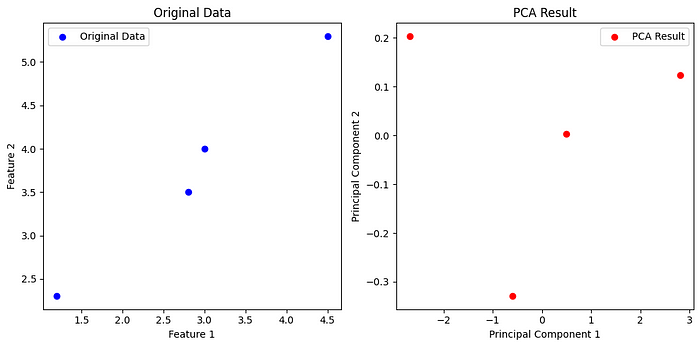
**Step 6**: Interpretation and Analysis Once the data is transformed, you can interpret the principal components and their relationships to the original variables. This is crucial for understanding the most significant patterns in the data.

**Code**

Here’s a Python code example that performs Principal Component Analysis (PCA) step by step using the popular Python libraries NumPy and scikit-learn. We’ll walk through each step of the process.

import numpy as np  
from sklearn.decomposition import PCA  
from sklearn.preprocessing import StandardScaler  
import matplotlib.pyplot as plt  
  
# Step 1: Data Collection and Standardization  
# Example data (replace this with your own dataset)  
data = np.array([[1.2, 2.3, 4.0, 3.8],  
 [2.8, 3.5, 6.5, 5.0],  
 [3.0, 4.0, 7.9, 6.2],  
 [4.5, 5.3, 9.0, 7.8]])  
  
# Standardize the data  
scaler = StandardScaler()  
data\_standardized = scaler.fit\_transform(data)  
  
# Step 2: Covariance Matrix Calculation  
cov\_matrix = np.cov(data\_standardized, rowvar=False)  
  
# Step 3: Eigenvalue and Eigenvector Computation  
eigenvalues, eigenvectors = np.linalg.eig(cov\_matrix)  
  
# Step 4: Sorting and Selecting Principal Components  
# Sort eigenvalues and their corresponding eigenvectors in descending order  
eigenvalue\_eigenvector\_pairs = [(eigenvalues[i], eigenvectors[:, i]) for i in range(len(eigenvalues))]  
eigenvalue\_eigenvector\_pairs.sort(key=lambda x: x[0], reverse=True)  
  
# Choose the number of principal components to keep (e.g., 2 components)  
num\_components = 2  
selected\_eigenpairs = eigenvalue\_eigenvector\_pairs[:num\_components]  
  
# Create the projection matrix from selected eigenvectors  
projection\_matrix = np.column\_stack((selected\_eigenpairs[0][1], selected\_eigenpairs[1][1]))  
  
# Step 5: Data Transformation  
# Project the standardized data onto the new reduced-dimensional space  
data\_pca = data\_standardized.dot(projection\_matrix)  
  
# Step 6: Plot the Original Data and PCA Results  
plt.figure(figsize=(10, 5))  
  
# Plot the original data  
plt.subplot(1, 2, 1)  
plt.scatter(data[:, 0], data[:, 1], c='b', label='Original Data')  
plt.xlabel('Feature 1')  
plt.ylabel('Feature 2')  
plt.title('Original Data')  
plt.legend()  
  
# Plot the PCA results  
plt.subplot(1, 2, 2)  
plt.scatter(data\_pca[:, 0], data\_pca[:, 1], c='r', label='PCA Result')  
plt.xlabel('Principal Component 1')  
plt.ylabel('Principal Component 2')  
plt.title('PCA Result')  
plt.legend()  
  
plt.tight\_layout()  
plt.show()

This code first collects your data and standardizes it, then calculates the covariance matrix, computes the eigenvalues and eigenvectors, sorts and selects the principal components, and finally, transforms the data. Please make sure to replace the example data with your actual dataset for a meaningful PCA analysis.



The two charts in the example represent the original data and the result of applying Principal Component Analysis (PCA). Let’s discuss the key differences and benefits of using PCA for these charts:

**Original Data (Left Chart):**

1. **Dimensionality**: The left chart shows the original data in its native feature space. In this example, we have two features (Feature 1 and Feature 2), and the data points are plotted in this 2D space.
2. **Complexity**: The original data chart represents the data in its raw form, which might contain redundant or correlated information. In this case, the two features are plotted directly, and there’s no attempt to reduce dimensionality.
3. **Interpretability**: The original data chart is straightforward to interpret because it represents the data as it is. Each axis corresponds to a specific feature, and the relationships between data points are clear.

**PCA Result (Right Chart):**

1. **Dimensionality Reduction**: The right chart represents the result of applying PCA to the data. In this case, we’ve reduced the dimensionality from the original 2D space to a new 2D space formed by the first two principal components.
2. **Simplification**: PCA identifies the directions (principal components) in the data space where the most variance is explained. By projecting the data onto these components, PCA simplifies the data while preserving as much of the original variance as possible.
3. **Orthogonality**: The principal components are orthogonal (uncorrelated), meaning they capture different patterns in the data. The first component captures the most significant variance, and the second component captures the second most significant variance while being orthogonal to the first.

**Benefits of PCA:**

1. **Dimensionality Reduction:**PCA is particularly useful when dealing with high-dimensional data. It reduces the number of features, making it easier to analyze and visualize complex datasets.
2. **Noise Reduction**: By retaining the most significant variance and discarding less important variance, PCA can help reduce the impact of noise in the data.
3. **Data Visualization**: PCA provides a way to project high-dimensional data into a lower-dimensional space, making it easier to visualize and interpret the data. It’s especially valuable when dealing with multi-dimensional datasets.
4. **Feature Engineering**: PCA can be used as a feature engineering technique to create new variables (principal components) that capture the most important patterns in the data.
5. **Decorrelation**: PCA decorrelates the data by ensuring that the principal components are orthogonal, which can be beneficial for certain algorithms that assume uncorrelated features.
6. **Preservation of Variance**: PCA ensures that the first principal component explains the most variance in the data, the second component explains the second most, and so on. This means that a high percentage of the total variance can often be retained with a relatively small number of principal components.

In summary, PCA is a powerful technique for dimensionality reduction and feature extraction. It simplifies data while retaining essential information, making it easier to work with and interpret complex datasets. The right chart in the example demonstrates the benefits of PCA by showing how it simplifies the data representation while preserving the most important patterns and relationships.

**Conclusions**

In conclusion, PCA is a valuable technique for reducing the dimensionality of data while retaining essential information. It involves data collection and standardization, covariance matrix calculation, eigenvalue and eigenvector computation, sorting and selecting principal components, data transformation, and interpretation and analysis of the results. PCA is widely used in various fields, such as machine learning, image processing, and data analysis, to simplify data and extract meaningful patterns.