We have a small dataset with two features (dimensions):

x1	x2
2	4
3	6
4	8
5	10

These points represent a 2D scatter plot, where `x1` is the first axis and `x2` is the second.

If you plot the points, you'll see that they lie on a straight line. This indicates that the two features are highly correlated.

Step 1: centering the data

Before applying PCA, we subtract the mean from each feature to center the data. For our dataset:

- Mean of `x1` = 3.5, Mean of `x2` = 7
- Centered data:

x1 (centered)	x2 (centered)
-1.5	-3
-0.5	-1
0.5	1
1.5	3

Step 2: Covariance Matrix

The covariance matrix is calculated as:

Covariance Matrix = \begin{bmatrix} \text{Var}(x1) & \text{Cov}(x1, x2) \\ \text{Cov}(x2, x1) & \text

For this dataset:

$$CovarianceMatrix = \begin{bmatrix} 1.25 & 2.5 \\ 2.5 & 5 \end{bmatrix}$$

Step 3: Eigenvalues and Eigenvectors

PCA identifies eigenvalues and eigenvectors of the covariance matrix. These define the directions (eigenvectors) and the variance (eigenvalues) along those directions.

- Eigenvalues: λ_1 = 6.25, λ_2 = 0
- Eigenvectors:
  - For λ_1: $[0.447, 0.894]$ (direction of maximum variance)
  - For λ_2: $[-0.894, 0.447]$ (perpendicular direction)

Step 4: Transforming the data

To project the data onto the principal components, multiply the centered data by the eigenvectors.
Projected data onto PC1 (1D):

PC1
-3.35
-1.12
1.12
3.35

Step 8: Visual Interpretation

- The original data lies in 2D space.
- After PCA, the data is effectively represented along PC1 (1D), which captures all the variance (since λ2=0\lambda_2 = 0).

Takeaway for Students

- PCA identifies the "important" directions in the data (PCs).
- It reduces redundant information by compressing dimensions while retaining variance.
- In this example, PCA reduces the 2D data to 1D without significant information loss, since the data lies along a line.

Would you like a Python implementation to visualize this?