Module 23: Unsupervised Learning: Part Two: Principal Component Analysis

Learning outcomes

1. Identify the principal components that capture the most variance in a data set.
2. Determine how data characteristics and preprocessing choices affect PCA results.
3. Perform PCA on a data set using Python.
4. Analyse real-world applications of PCA and the impact of ML methods in a specific industry.

Variance and principal components

- **Variance** measures how much data points differ from the mean.
- **Principal components** are new axes that capture the directions of **maximum variance** in the data.
- Each component is a **linear combination** of the original features weighted by **loadings** that indicate each feature's contribution.
- The first component explains the most variance, the second explains the next most while being **orthogonal** (uncorrelated) to the first, and so on.

Explained variance

- Explained variance tells you how much information each component retains.
- The explained variance ratio for component i given by:

$$Explained\ variance\ ratio\ (PC_i) = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}$$

  Where λ_i is the eigenvalue for principal component i, and the denominator is the sum of all eigenvalues (total variance).
- Use a **scree plot** or **cumulative variance plot** to decide how many components to keep – typically, the number before the 'elbow' point on the curve.

Data preprocessing for PCA

PCA assumes features are measured on comparable scales:

- **Mean centring:** subtract the mean so each feature has mean 0.
- **Scaling:** divide by the standard deviation so each feature has variance 1.

Without these steps, features with large numeric ranges dominate the analysis.

Kernel PCA

When data relationships are **non-linear**, standard PCA may not capture the structure effectively.
Kernel PCA uses a kernel function (e.g. RBF, polynomial or cosine) to implicitly map data into a higher-dimensional feature space before applying PCA.

Typically, kernel PCA:

- Reveals curved or complex patterns
- Is commonly used for **image denoising**, **pattern recognition** and **non-linear separation** tasks

Applications

PCA is used across domains for dimensionality reduction, noise filtering and data interpretation, including:

- **Finance:** condensing large covariance matrices to uncover underlying risk factors
- **Manufacturing:** monitoring processes via PCA-based fault detection using Q-residuals and Hotelling's T^2 statistics
- **Healthcare:** identifying latent gene or imaging patterns for diagnosis
- **Face recognition:** using eigenfaces for compact visual representation
- **General ML workflows:** reducing collinearity and improving training efficiency

When to use PCA

Consider using PCA in the following situations:

- When you have **many correlated features**
- To **simplify visualisation** of high-dimensional data (2D/3D plots)
- To **reduce overfitting** by eliminating redundant information
- As a **preprocessing step** before clustering or supervised learning