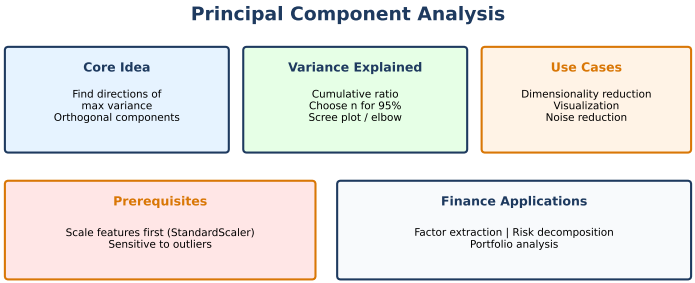
Lesson 31 Summary: PCA

Data Science with Python – Key Concepts

Data Science Program

PCA Overview



*PCA(n_components=2).fit_transform(X_scaled) | pca.explained_variance_ratio_*

PCA finds the directions of maximum variance

The Core Idea

Dimensionality reduction:

- **Find:** Directions with most variance
- **Project:** Data onto these directions
- **Result:** Fewer dimensions, preserved structure

Principal components are orthogonal to each other

Variance Explained

Choosing number of components:

- **Cumulative ratio:** Sum of explained variance
- **Rule of thumb:** Keep 95% of variance
- **Scree plot:** Look for elbow

First few PCs often capture most of the variance

Scaling is Critical

Why standardize:

- **Problem:** PCA sensitive to scale
- **Solution:** StandardScaler before PCA
- **Result:** Equal contribution from all features

Always scale features before PCA

Interpreting Components

Understanding PCs:

- **Loadings:** Weights of original features
- **High loadings:** Important for that PC
- **Sign:** Direction of relationship

Access loadings via pca.components_

Visualization

Common uses:

- **2D scatter:** PC1 vs PC2
- **Color by label:** See cluster separation
- **Biplot:** Show feature contributions

PCA is a powerful visualization tool

Limitations

When PCA struggles:

- **Non-linear:** Only captures linear relationships
- **Outliers:** Can distort principal components
- **Interpretation:** PCs may not be meaningful

Consider t-SNE or UMAP for non-linear data

Finance Applications

Use cases:

- **Factor extraction:** Find market factors
- **Risk decomposition:** Identify risk sources
- **Portfolio compression:** Reduce dimensionality

PCA is widely used in quantitative finance

Quick Reference

Essential Commands:

Task	Code
Scale data	`StandardScaler().fit_transform(X)`
Fit PCA	`PCA(n_components=2).fit(X_scaled)`
Transform	`pca.transform(X_scaled)`
Variance ratio	`pca.explained_variance_ratio_`
Loadings	`pca.components_`

PCA is a fundamental unsupervised technique