Unsupervised Learning

HI 743

Ryan Gallagher

Department of Health Informatics and Administration
Zilber College of Public Health
University of Wisconsin - Milwaukee

March 25th, 2025

Overview

1. **Review Final Project Rubric**

2. **Overview**

3. **Principal Component Analysis (PCA)**
   3.1 USArrests Data

4. **Missing Values & Matrix Completion**

Title

Let’s look at Final Project.pdf

Unsupervised Learning

What is Unsupervised Learning?

- In supervised learning, we observe features $X_1, X_2, \ldots, X_p$ and a response Y, and our goal is to predict Y using the X's.
- In **unsupervised learning**, we only observe features $X_1, X_2, \ldots, X_p$—*no response variable* Y.
- The goal is not prediction, but **exploration**: to discover interesting patterns or structures in the data.

Key Questions:

- Is there a useful way to **visualize** the data?
- Can we identify **groups of similar observations or variables**?

Challenges of Unsupervised Learning

”Exploratory Data Analysis” - Inference is Subjective

- In supervised learning, model performance can be evaluated using Y
- In unsupervised learning, there is no obvious way to check if results are “correct”

Implications:

- Many different valid ways to define structure or clusters
- Results depend heavily on:
  - Method used
  - Distance or similarity metrics
  - Data scaling and preprocessing

Requires careful interpretation and domain knowledge

Principal Component Analysis (PCA)

What is PCA?

- A method to reduce the dimensionality of a dataset.
- Finds new features (called **principal components**) that are:
  - Linear combinations of the original variables.
  - Uncorrelated with each other.
  - Ranked by how much variance they explain in the data.
- Often used for:
  - Visualization of high-dimensional data.
  - Preprocessing before supervised learning.

What is a Principal Component?

Definition:

- A **principal component (PC)** is a direction in feature space along which the data varies the most.
- The first PC is the direction of **maximum variance**.
- The second PC is orthogonal to the first and explains the next highest variance, and so on.

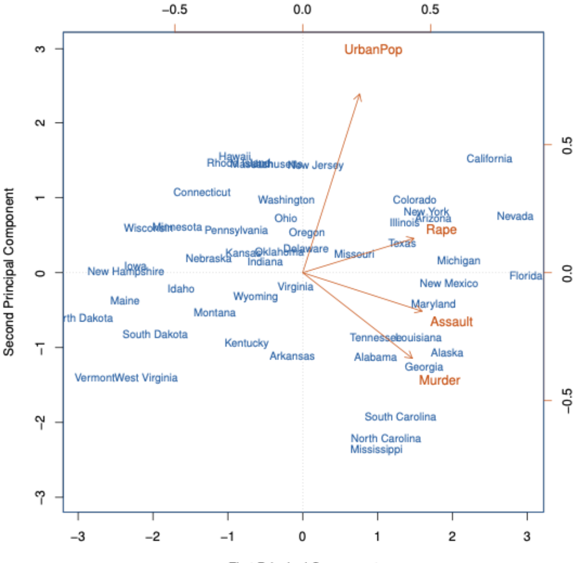
Mathematically:

- $\mathrm{PC}_1 =$ direction v_1 such that

$$v_1 = \arg\max_{\|v\|=1} \mathrm{Var}(Xv)$$

- The projection of each observation onto v_1 gives the first principal component score.

Visualizing PCA - USArrests



USArrests data:

- Data points in 2D (e.g., X_1 and X_2)
- **First Principal Component (PC1)**: direction of greatest variance
- **Second Principal Component (PC2)**: orthogonal to PC1, captures remaining variance

PCA on USArrests Data (Figure 12.1)

Context:

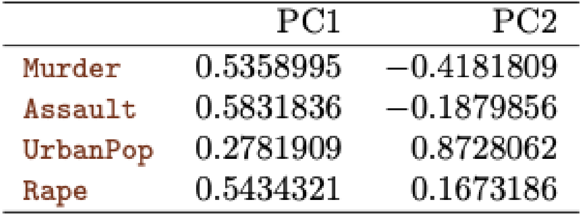
- Dataset: `USArrests` — crime statistics (Murder, Assault, UrbanPop, Rape) for 50 U.S. states
- PCA applied after standardizing the variables

Results shown in Figure 12.1:

- **PC1** accounts for the largest variance:
  - Separates states with high rates of violent crimes (Murder, Assault, Rape)
- **PC2** captures variation more related to UrbanPop
- States like California and Florida have high PC1 scores — indicating higher crime rates

PCA reveals underlying structure: crime-heavy vs. low-crime states in fewer dimensions.

PCA Loadings: Table 12.1



	PC1	PC2
`Murder`	0.5358995	−0.4181809
`Assault`	0.5831836	−0.1879856
`UrbanPop`	0.2781909	0.8728062
`Rape`	0.5434321	0.1673186

Loadings

- Loadings are coefficients that define each principal component.
- PC_1 emphasizes `Murder`, `Assault`, and `Rape`.
- PC_2 loads strongly on `UrbanPop` and contrasts with `Rape`.

Proportion of Variance Explained (PVE)

PVE

- Each principal component captures a different “direction” of variation in the data.
- PCA ranks these directions from most to least variation.
- The Proportion of Variance Explained (PVE) tells us how much of the total variation is captured by each component.

What insight does it give?

- It helps us understand which components are most important for representing the data.
- A high PVE for the first few components means the data can be summarized well with fewer dimensions.
- PVE is typically visualized using a scree plot: a chart showing how much variation each component explains.

Missing Values and Matrix Completion

Missing Data is common. In many datasets, some entries are missing. This is especially challenging in unsupervised settings where no outcome Y is available to help fill in the blanks.

Matrix Completion:

- Unsupervised learning can estimate missing values using the observed data.
- It assumes the true data matrix has some underlying structure (e.g., low-rank).
- Common in recommendation systems (e.g., Netflix: users rate only a few movies).

Fill in Missing Values via PCA (Algorithm 12.1)

Goal: Estimate the missing entries in a data matrix using the observed ones.

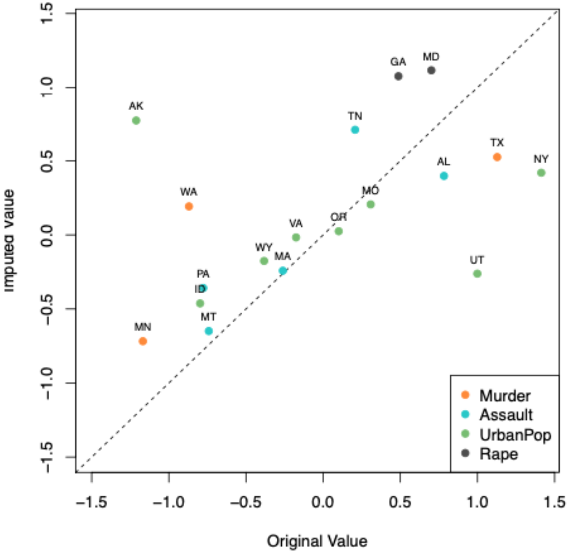
Idea Behind the Algorithm:

- We assume the complete data lies close to a low-dimensional space (like in PCA).
- Even with missing values, we can iteratively estimate the full matrix.

How It Works — In Plain Terms:

1. **Start by filling in** the missing values with simple guesses (like column means).
2. **Apply PCA** to the filled-in matrix to find the best low-rank approximation.
3. **Replace the missing entries** with the values predicted by the PCA.
4. **Repeat** until the estimates stop changing much.

Figure 12.5: Matrix Completion with PCA



- Plot of the `USArrests` dataset with some values removed at random.
- Missing values were filled using PCA-based matrix completion.
- X & Y axis agreement show accurate fit

K-Means Clustering - Quick Review

Group observations into K distinct, non-overlapping clusters based on similarity.

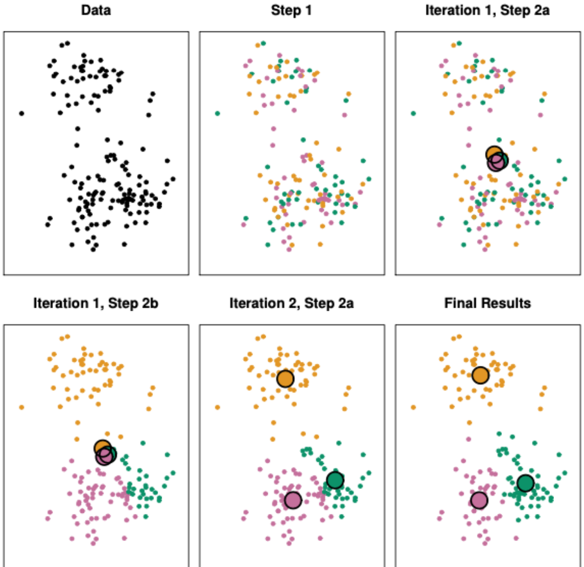
Algorithm

1. Choose the number of clusters, K.
2. Randomly assign each observation to one of the K clusters.
3. Compute the center (mean) of each cluster.
4. Reassign observations to the nearest cluster center.
5. Repeat steps 3–4 until assignments stop changing.

Key Idea:

- Each cluster groups together observations that are close in terms of their features.
- The algorithm tries to minimize the variation within clusters.

Figure 12.8: K-Means Clustering



- A toy dataset in two dimensions.
- Points are grouped into three clusters using K-means.
- Shows steps toward achieving the clustering.
- Each point is colored by its assigned cluster.

Hierarchical Clustering

Concept Overview

- Hierarchical clustering groups similar observations based on their distance.
- It creates a tree-like structure called a **dendrogram**, which shows how clusters are merged at different levels.
- Observations that are more similar are combined earlier (lower in the tree).
- Unlike K-means, you do not need to specify the number of clusters in advance.

The Algorithm (Produce a Dendrogram)

1. Start with each observation in its own cluster.
2. Compute the distances between all pairs of clusters.
3. Merge the two clusters that are closest together.
4. Repeat steps 2–3 until all observations are merged into a single cluster.

Hierarchical Clustering: Building a Tree of Similarity

Concept Overview

- Groups similar observations based on a notion of distance or similarity.
- Builds a tree-like structure called a **dendrogram**.
- Observations that are more similar are joined earlier in the tree.

Why Use It?

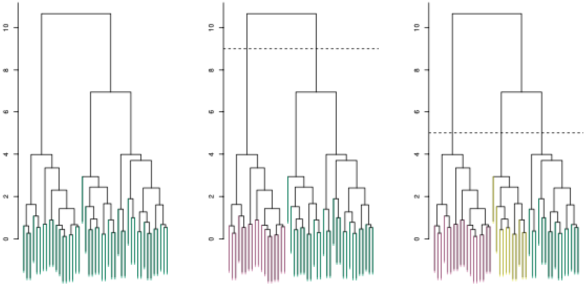
- No need to choose the number of clusters in advance.
- Helps visualize nested groupings and relationships.

Cutting the Dendrogram

- By slicing the dendrogram at a chosen height, you can form distinct clusters.

Hierarchical clustering reveals the structure of similarity in your data at multiple levels.

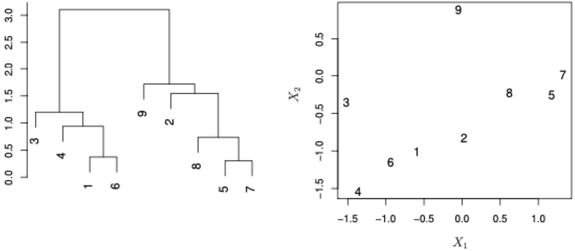
Figure 12.11: Cutting a Dendrogram for Clusters



- **Left:** Full dendrogram from hierarchical clustering (complete linkage, Euclidean distance).
- **Center:** Cutting at height 9 forms **two clusters**.
- **Right:** Cutting at height 5 forms **three clusters**.

- The **height of the cut** determines the number of clusters.
- One dendrogram allows us to explore multiple clustering solutions.

Figure 12.12: Interpreting Dendrogram Structure



- **Left panel:** A dendrogram built from 9 observations using complete linkage.
- **Right panel:** The raw data used to generate the dendrogram, shown in 2D space.
- For example, observations 9 and 2 appear near each other, but are not more similar than 9 is to 5, 7, or 8.

Practical Issues: Scaling and Setup Choices

Clustering isn't automatic — small choices can matter a lot.

Decisions that affect results:

- **Should the variables be standardized?**
  - Variables with larger scales (e.g., annual sock purchases vs. laptops) can dominate distance calculations.
  - Scaling to standard deviation 1 gives each variable equal weight.
- **Hierarchical Clustering Choices:**
  - What dissimilarity measure should we use? (e.g., Euclidean, correlation)
  - What type of linkage? (e.g., complete, average, single)
  - Where should we cut the dendrogram?
- **K-means Choices:**
  - How many clusters (K) should we choose?

Practical Issues: Validating and Interpreting

Are the clusters meaningful? Or just random patterns?

Validating the Clusters Obtained:

- Clustering methods *will* produce groups—even from random data.
- We must ask: are these clusters real or noise?
- One approach: apply clustering to a dataset with no actual group structure (e.g., data drawn from a single Gaussian) and compare results.

Other Considerations:

- Cluster analysis is more about **exploration** than strict inference.
- Results may depend heavily on:
  - Method used (e.g., K-means vs. hierarchical)
  - Distance metric
  - Preprocessing choices (e.g., scaling)
- It's common to try multiple methods and compare for robustness and interpretability.