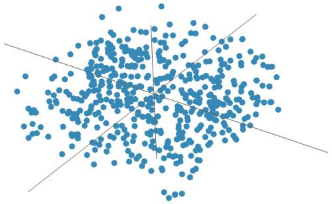
Dimensionality Reduction

Guillem & Roderic, Summer 2025

Goals

- Understand the need and purpose of dimensionality reduction algorithms.
- Giver overview of Principal Components Analysis (PCA).
- See applications of PCA in context.
- Compare to and combine with other algorithms.

Point Cloud Data



A point cloud is a collection of data points in $\mathbb{R}^m$

- Each point is represented according to its coordin $(X_1, X_2, X_3, \ldots, X_m)$
- Each coordinate represents a different numerical feature of each observation:
  - **Hotels in a city** can be represented in $\mathbb{R}^6$ according to user ratings of: cleanliness, accuracy, communication, check-in, location, value
  - A **4x4 pixels grayscale image** can be represented in $\mathbb{R}^{16}$: each pixel is represented with a unique number according to a scale from black to white
  - Samples of **expressions of 10 genes** in cells can be represented in $\mathbb{R}^{10}$

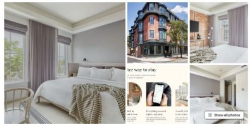
Example 1: Hotel Listings

- Each data point in $\mathbb{R}^6$ corresponds to a hotel
- Hotels are ranked according to 6 categories:
- Each individual hotel can be represented by a point in

Cleanliness	4.8	Accuracy	4.8
Communication	5.0	Location	4.7
Check-in	5.0	Value	4.8

$$= \begin{bmatrix} 3.8 \\ 4.6 \\ 4.8 \\ 4.2 \\ 4.7 \\ 4.3 \end{bmatrix}$$

$$= \begin{bmatrix} 4.8 \\ 5 \\ 5 \\ 4.8 \\ 4.7 \\ 4.8 \end{bmatrix}$$



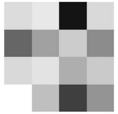
$$= \begin{bmatrix} 4.9 \\ 4.5 \\ 4.7 \\ 4.7 \\ 4.9 \\ 4.5 \end{bmatrix}$$

$$= \begin{bmatrix} 4.6 \\ 4.7 \\ 4.7 \\ 4.5 \\ 4.1 \\ 4.2 \end{bmatrix}$$

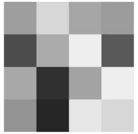
$$= \begin{bmatrix} 4.6 \\ 4.7 \\ 4.7 \\ 4.7 \\ 4.9 \\ 4.5 \end{bmatrix}$$

Example 2: Grayscale Images

- Each data point corresponds to an image of resolution 4x4
- Each of the 16 pixels is represented with a number from 0 (black) to 1 (white)
- Each image can be represented by a point in $\mathbb{R}^{16}$



$$\longrightarrow \begin{pmatrix} 0.86 & 0.91 & 0.08 & 0.85 \\ 0.4 & 0.63 & 0.8 & 0.55 \\ 0.85 & 0.89 & 0.68 & 0.79 \\ 1. & 0.75 & 0.25 & 0.58 \end{pmatrix} \longrightarrow \begin{pmatrix} 0.86 \\ 0.91 \\ 0.08 \\ 0.85 \\ 0.4 \\ 0.63 \\ 0.8 \\ 0.55 \\ 0.85 \\ 0.89 \\ 0.68 \\ 0.79 \\ 1. \\ 0.75 \\ 0.25 \\ 0.58 \end{pmatrix}$$



$$\longrightarrow \begin{pmatrix} 0.62 & 0.84 & 0.65 & 0.61 \\ 0.3 & 0.67 & 0.93 & 0.35 \\ 0.66 & 0.19 & 0.64 & 0.93 \\ 0.58 & 0.15 & 0.9 & 0.84 \end{pmatrix} \longrightarrow \begin{pmatrix} 0.62 \\ 0.84 \\ 0.65 \\ 0.61 \\ 0.3 \\ 0.67 \\ 0.93 \\ 0.35 \\ 0.66 \\ 0.19 \\ 0.64 \\ 0.93 \\ 0.58 \\ 0.15 \\ 0.9 \\ 0.84 \end{pmatrix}$$

Example 3: Gene Expression

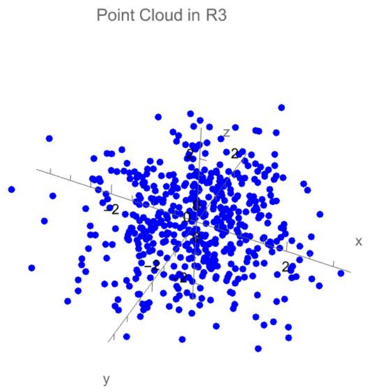
- Each sample measures expressions of N genes in distincts cells
- Each individual cell can be represented by a point in $\mathbb{R}^N$

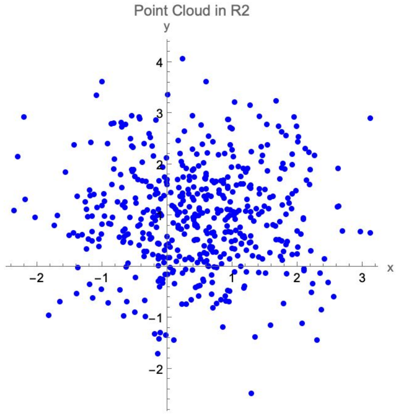
Sample ID	Gene1	Gene2	Gene3	...	GeneN
Sample1	5.2	0.1	3.4	...	7.6
Sample2	4.9	0.0	3.8	...	6.8

$$\text{sample}_1 = \begin{bmatrix} 5.2 \\ 0.1 \\ 3.4 \\ \vdots \\ 7.6 \end{bmatrix} \qquad \text{sample}_2 = \begin{bmatrix} 4.9 \\ 0.0 \\ 3.8 \\ \vdots \\ 6.8 \end{bmatrix}$$

Why Dimensional Reduction?

- Is it **practical** to work with high-dimensional data?
- Is there a way to **visualize** high-dimensional data?
- Is there a way to determine if any features more important than others?
- Are any **combinations of features** more relevant than others?





Why Dimensional Reduction?

- Enable visualization (we can only* visualize in $\mathbb{R}^2$ or $\mathbb{R}^3$)
- Reduce computational complexity
- Reduce redundancy and noise
- Reduce overfitting
- Find correlations between input features

What do we need?

- Statistics
- Linear Algebra
- Topology (more advanced)

Statistics Basics

Statistics Measurements

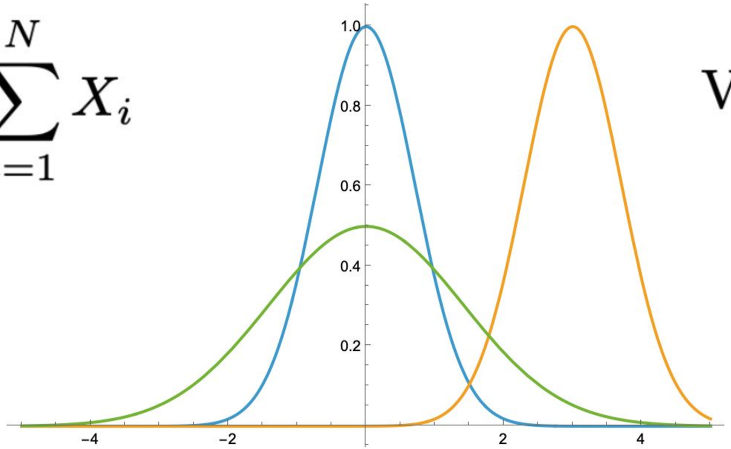
Suppose we have N measurements of a certain feature: $X_1, X_2, \ldots, X_N$

The **mean** is the central tendency or "average" of a set of numbers:

$$\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$$

The **variance** measures how spread out the values are around the mean:

$$\mathrm{Var}(X) = \frac{1}{N}\sum_{i=1}^{N}(X_i - \overline{X})^2$$



Statistics Measurements

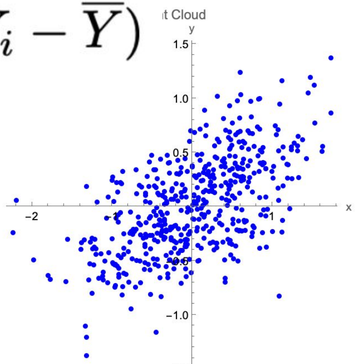
Suppose we have N measurements of two features:

$$X_1, X_2, \ldots, X_N$$
$$Y_1, Y_2, \ldots, Y_N$$

The **covariance** is a measure of how two variables change together—whether they tend to increase or decrease at the same time.

$$\mathrm{Cov}(X, Y) = \frac{1}{N} \sum_{i=1}^{N} (X_i - \overline{X})(Y_i - \overline{Y})$$

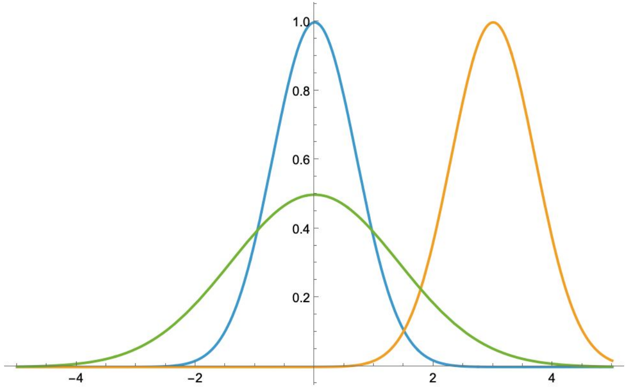
The **correlation** is the standardized version of covariance.



Statistics Measurements

When comparing variances of different numerical features in a dataset, **the higher the variance the more representative that feature is**. For example:

- All hotels are rated with an accuracy between 3 and 5, and
- are rated with a cleanliness between 1 and 5, therefore
- cleanliness is a more important feature!



Linear Algebra Basics

Linear Transformations

Any matrix can be thought of as a function via matrix-vector multiplication. A matrix with m columns and n rows “is” a function from $\mathbb{R}^m$ to $\mathbb{R}^n$.

- Example of a linear transformation from $\mathbb{R}^3$ to $\mathbb{R}^2$:

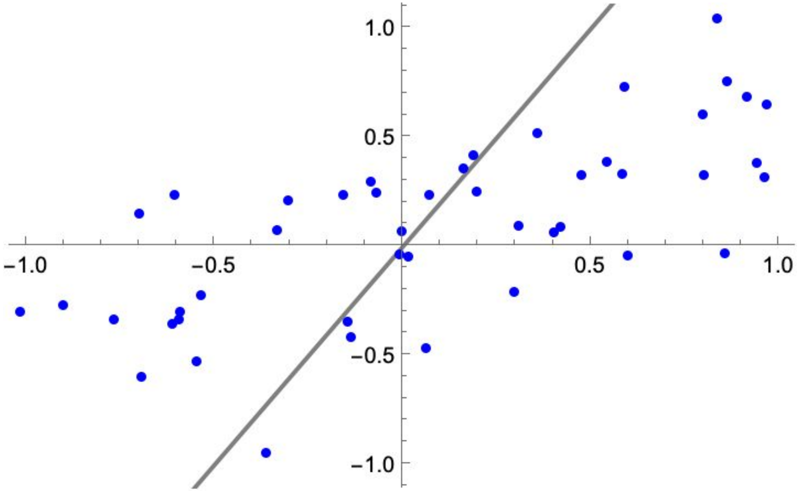
Input

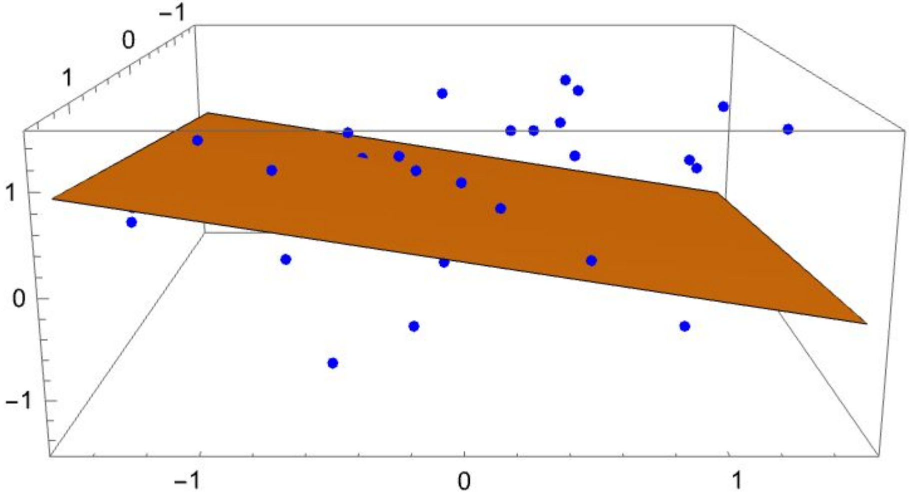
Output

$$\begin{bmatrix} 1 & -1 & 2 \\ -2 & 0 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \end{bmatrix}$$

Orthogonal Projections

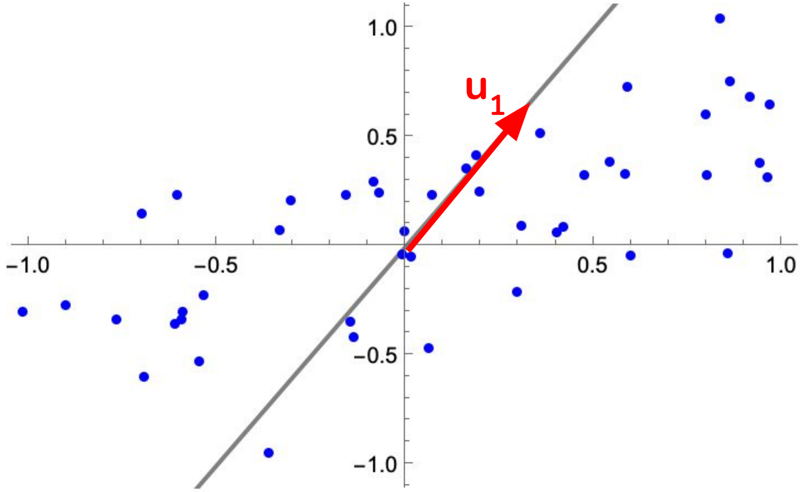
- Orthogonal projections are linear transformations (defined by a matrix)
- Let V be a n-dimensional subspace of $\mathbb{R}^n$ (line in $\mathbb{R}^2$, line or plane in $\mathbb{R}^3$, etc.)
- Orthogonal projections minimize distance between points and projections.

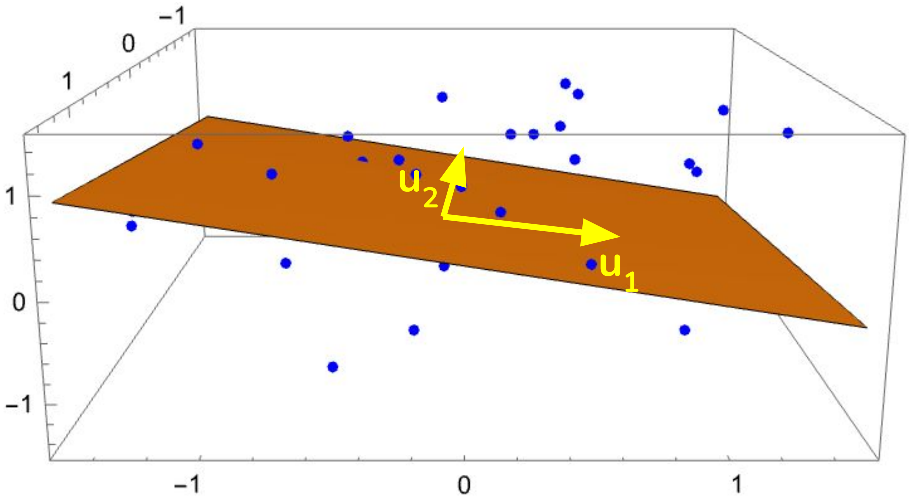




Orthogonal Projections

- Projection formula $\text{proj}(\vec{v}) = (\vec{u}_1 \cdot \vec{v})\vec{u}_1 + (\vec{u}_2 \cdot \vec{v})\vec{u}_2 + \cdots + (\vec{u}_k \cdot \vec{v})\vec{u}_k$
- Need a basis of perpendicular unit vectors (orthonormal basis)





Eigenvalues & Eigenvectors

Eigenvectors and eigenvalues are specific properties of square (nxn) matrices.

- Eigenvectors are vectors that scale by a constant when transformed.
- Eigenvalues can be real or complex.
- Not all matrices have the "expected" number of eigenvectors.
- If a matrix has a basis of eigenvectors, then is it **diagonalizable**

Suppose A is an $x \times x$ matrix. A nonzero vector $\vec{v}$ in $\mathbb{R}^n$ is an **eigenvector** of A of **eigenvalue** λ if

$$A\vec{v} = \lambda\vec{v}.$$

Symmetry, SVD & Eigendecomposition

- A square matrix M is symmetric is M^T=M
- Symmetric matrices satisfy
  - All eigenvalues are real
  - As many (l.i.) eigenvectors as their dimension
  - The eigenvectors can be chosen to be orthogonal
  - In summary: are **orthogonally diagonalizable**

$$\begin{bmatrix} 1 & -1 & 2 \\ -1 & 0 & 3 \\ 2 & 3 & 4 \end{bmatrix}$$

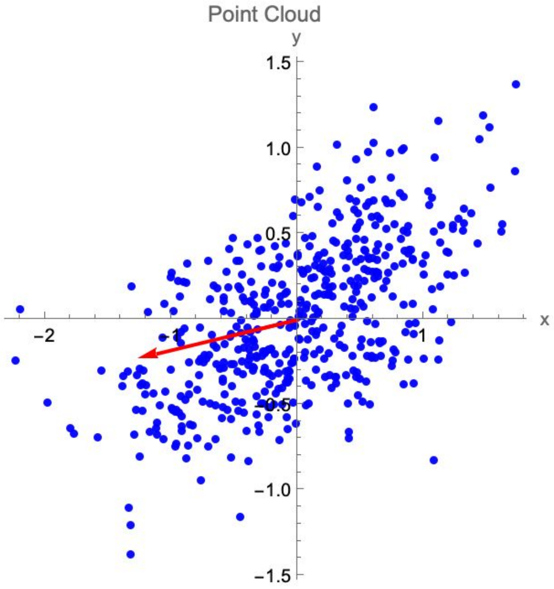
Linear Algebra Summary

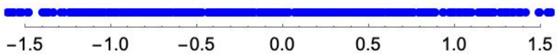
- **Projections** are linear transformations that require a basis of perpendicular unit vectors (orthonormal basis)
- **Symmetric Matrices** are orthogonally diagonalizable, that means it has a basis of **perpendicular unit eigenvectors**.
- Diagonalizing a symmetric matrix is efficient computationally.
- PCA uses symmetric matrices to find directions in which data is more spread, and projection

Principal Components Analysis

Principal Component Analysis

- PCA **finds the direction(s) in which the data varies the most** (i.e., is most spread out), and
- **projects the data onto those directions** to reduce dimensionality while preserving as much variance as possible.





Step 0: Gathering The Data

- Consider a multidimensional dataset consisting of **N** observations of **m** different characteristics X_i :

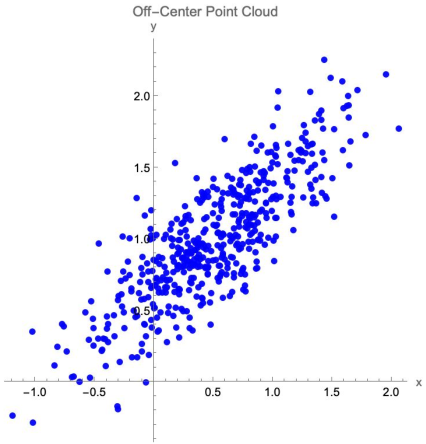
$$(X_1^{(1)}, X_2^{(1)}, X_3^{(1)}, \ldots, X_m^{(1)})$$
$$(X_1^{(2)}, X_2^{(2)}, X_3^{(2)}, \ldots, X_m^{(2)})$$
$$\vdots$$
$$(X_1^{(N)}, X_2^{(N)}, X_3^{(N)}, \ldots, X_m^{(N)})$$

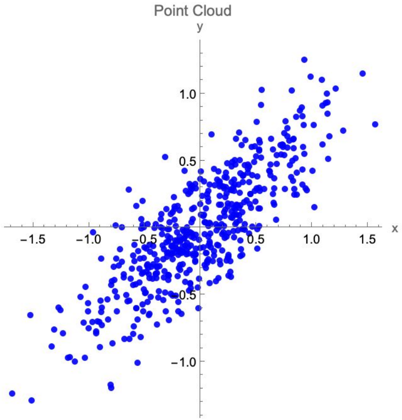
- This data lives in a high-dimensional space $\mathbb{R}^m$, that is "impossible" for us to visualize

Step 1: Standardizing The Data

- Center the data: $X_i - \overline{X}$
- Standardize (typically): $\dfrac{X_i - \overline{X}}{\sigma}$

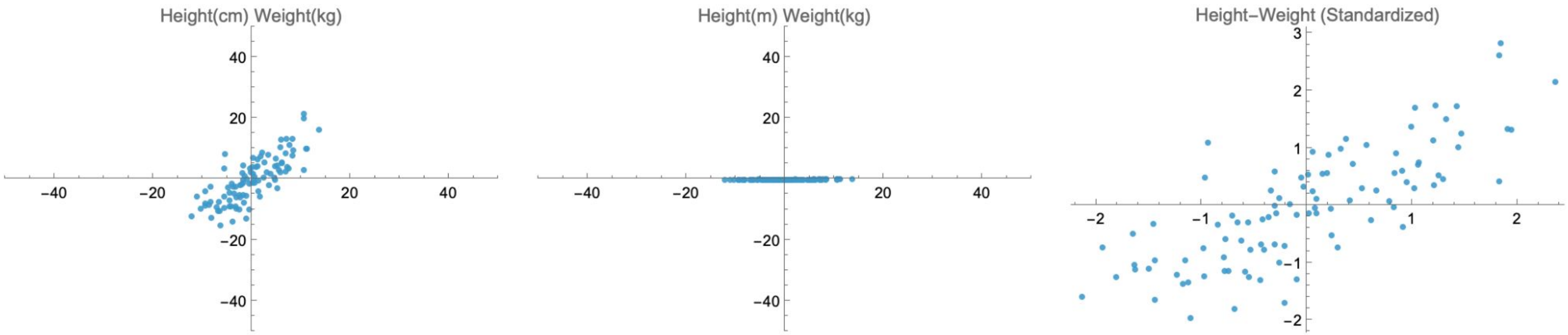
- Point cloud before/after centering





Step 1: Standardizing The Data

- Visual: why is it important to standardize?
  - Remove dependency on units
  - Get rid of scaling differences



Height vs weight of a 100 person sample

Step 2: Finding Covariance Matrix

- Find the covariance matrix:

$$\mathrm{Cov}(\vec{X}) = \begin{bmatrix} \mathrm{Var}(X_1) & \mathrm{Cov}(X_1, X_2) & \cdots & \mathrm{Cov}(X_1, X_m) \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Var}(X_2) & \cdots & \mathrm{Cov}(X_2, X_m) \\ \vdots & & \ddots & \vdots \\ \mathrm{Cov}(X_m, X_1) & \mathrm{Cov}(X_m, X_2) & \cdots & \mathrm{Var}(X_m) \end{bmatrix}$$

- Computational shortcut: if M is the matrix of your standardized data. Then

$$\mathrm{Cov}(\vec{X}) = \frac{1}{N} M^T M$$

Step 3: Eigenvectors and Eigenvalues

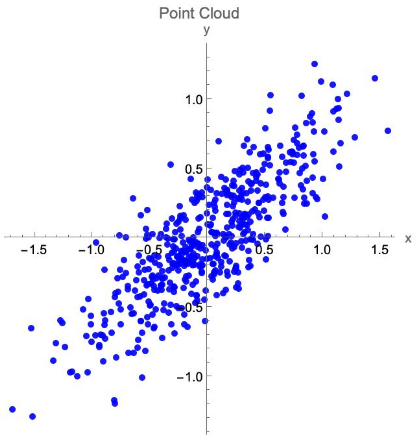
When data is standardized, the eigenvalues of the covariance matrix $\mathrm{Cov}(\vec{X})$ measure the proportion of the variance in the direction of the corresponding eigenvectors.

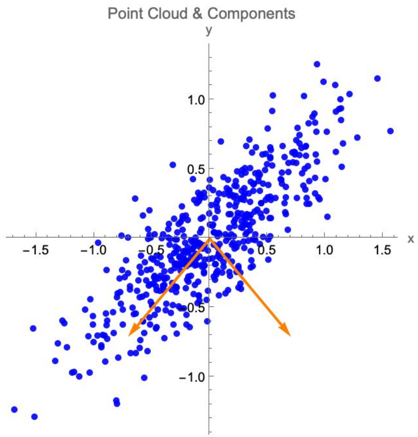
$$\sum_{i=1}^{N} \lambda_i = \sum_{i=1}^{N} \mathrm{Var}(X_i) = N$$

Eigenvector of largest eigenvalue <-> first principal component,
Eigenvector of second largest eigenvalue <-> second principal component,
Etc.

Step 4: Choose Number of Principal Components

- There are as man principal components as dimensions of your initial data.
- Choose a number N of principal components to project onto
- Pick the eigenvectors with the largest N eigenvalues



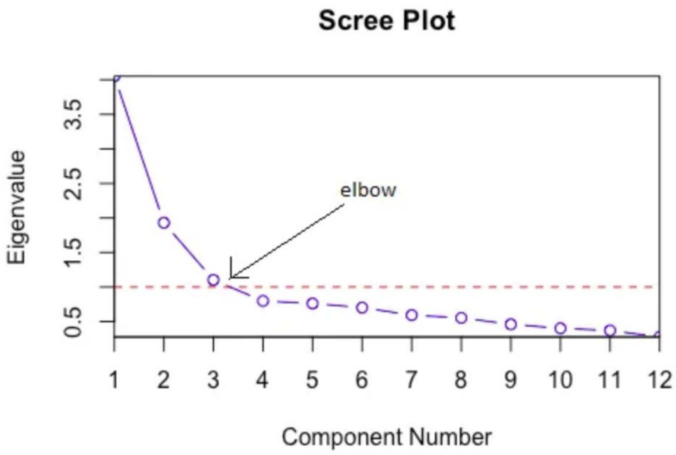


Principal components of a 2D point cloud

Step 4: Choosing Number of Principal Components

Deciding the number of components onto which the

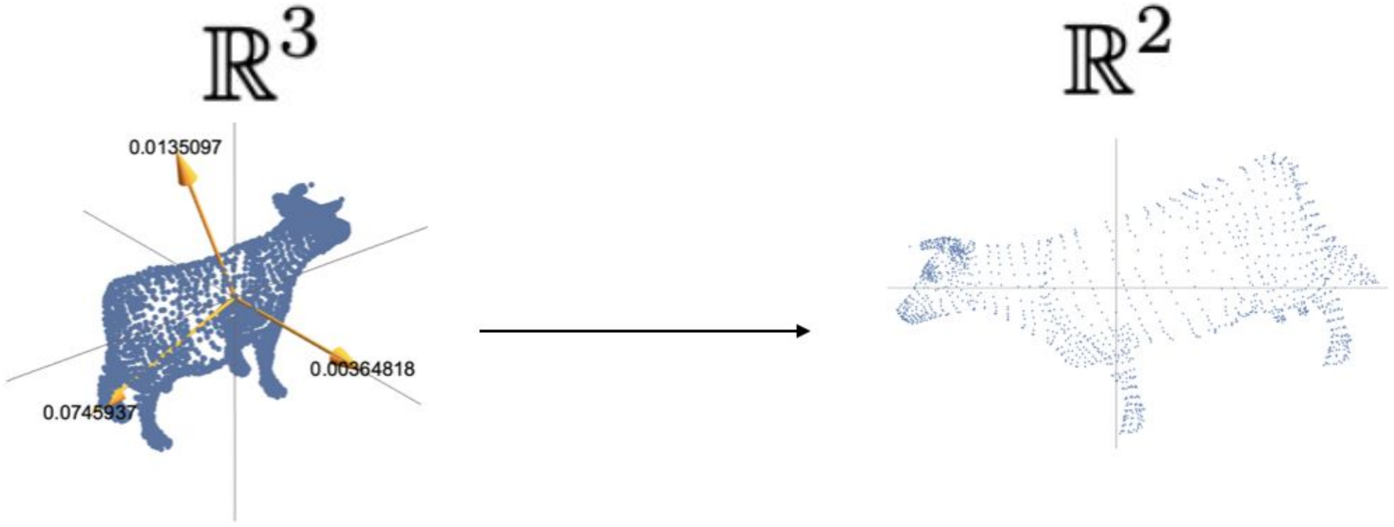
- For visualization purposes 2 or 3 (obvious reasons)
- Elbow Rule: (shown below)
- Kaiser Rule: pick eigenvectors with eigenvalue greater than 1



Step 5: Project The Data

Example: a cow from $\mathbb{R}^3$ to $\mathbb{R}^2$

Projection onto the **first two** principal components.

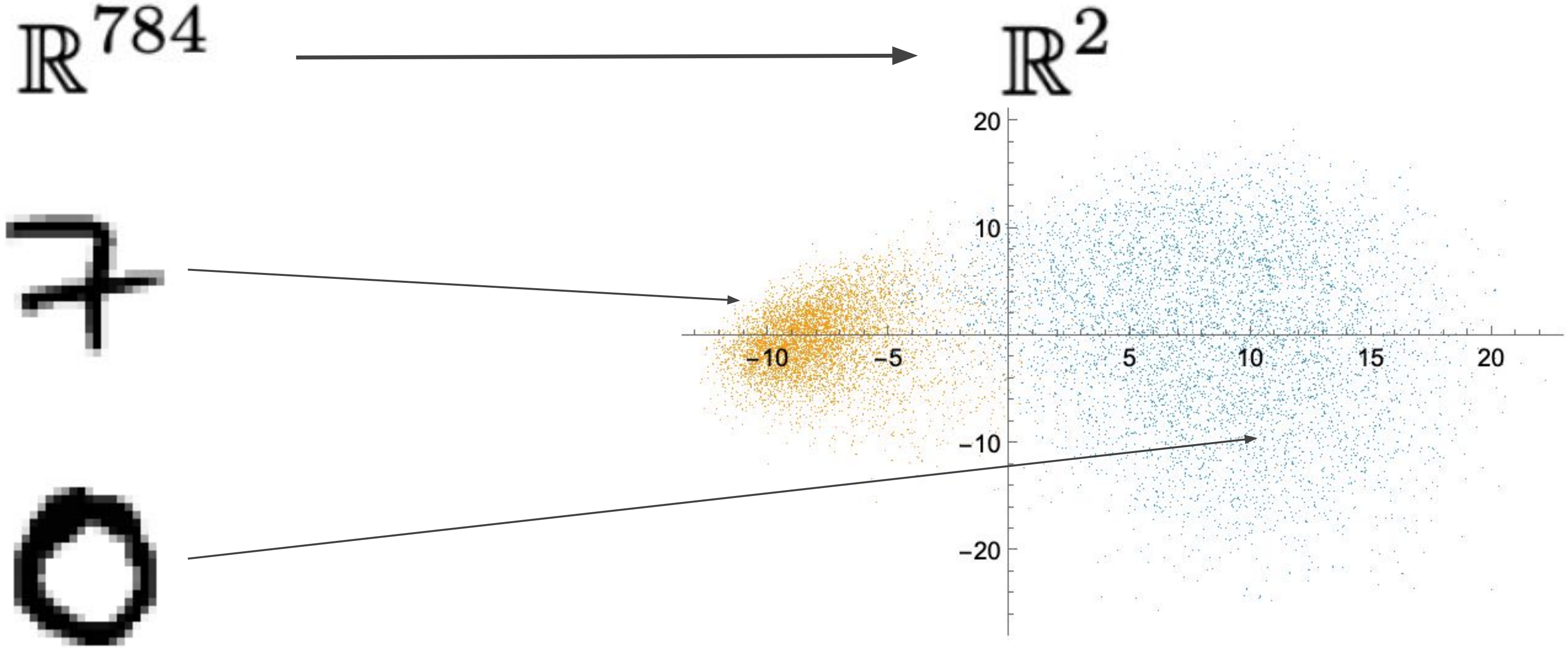


3D Cow

2D Cow

Step 5: Project The Data

Example: Classifying 28x28 handwritten digits

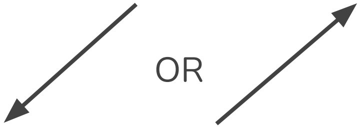


Principal Component Analysis: Summary

- Standardize (or center) each feature
- Compute covariance matrix
- Find eigenvectors and eigenvalues of the covariance matrix
  - The eigenvalues represent the proportion of overall variance in the direction of the eigenvector
  - Select a number of eigenvectors according to their eigenvalues
  - Project the data onto those eigenvectors
  - Read off the combinations of features that are more relevant

Subtleties

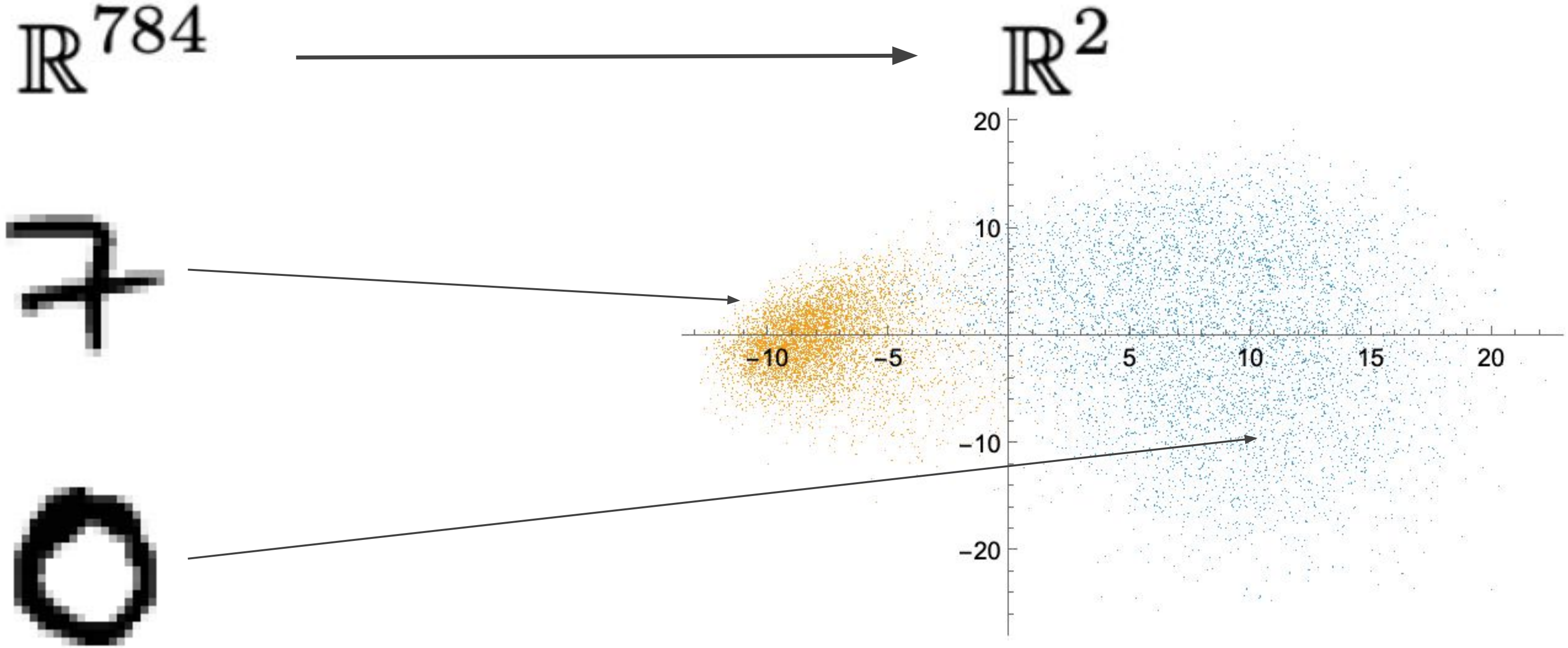
- Built-in algorithms will center your data, but (typically) won't standardize it.
- There is a sign ambiguity when choosing the eigenvectors.



Questions?

PCA + Other Algorithms

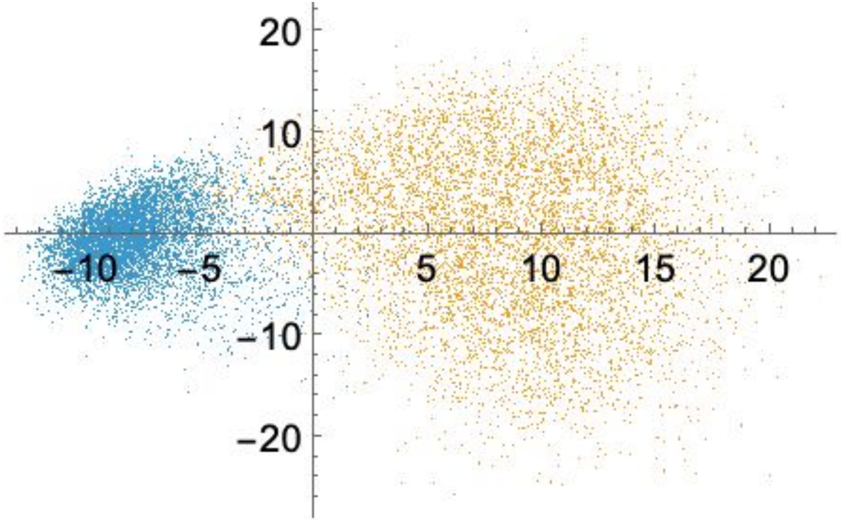
Running Example: Classifying 0s and 7s

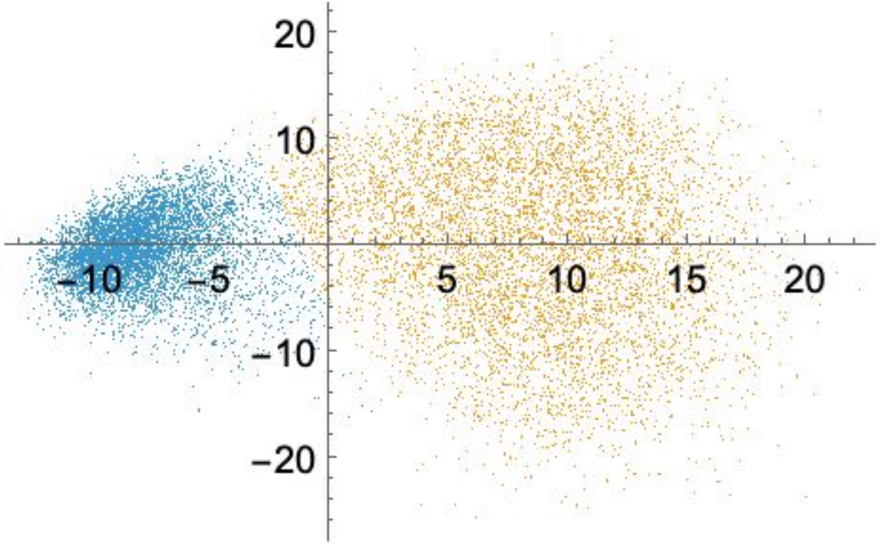


PCA + Logistic Regression

- First run Logistic Regression
- Then apply PCA
- Timing: 27s

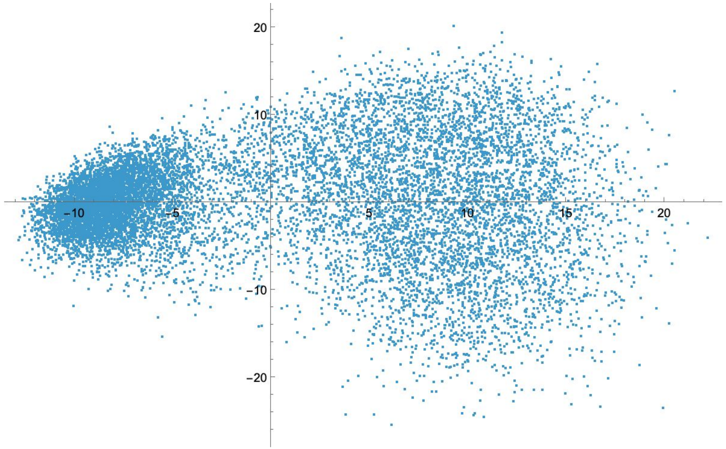
- First apply PCA
- Then run Logistic Regression
- Timing: 1.9s



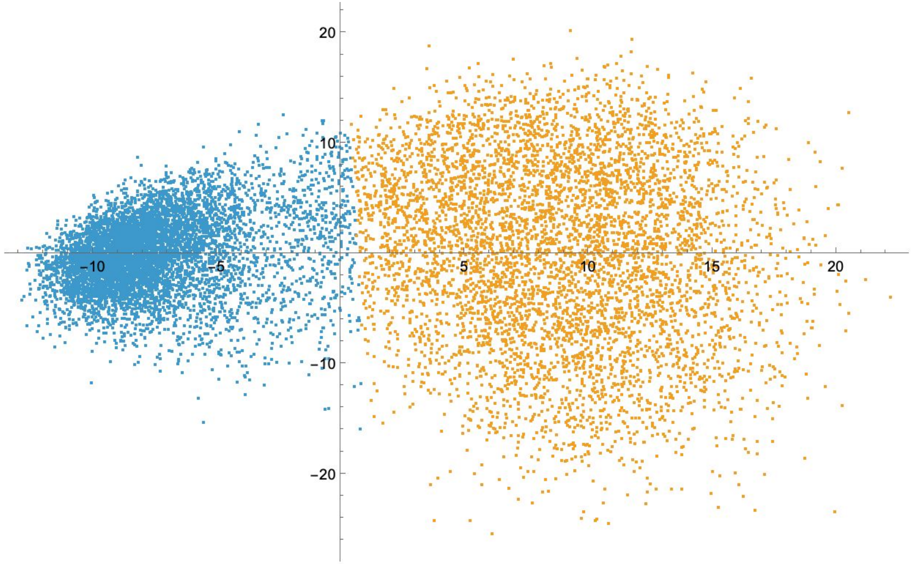


PCA + Clustering

- First find 2 clusters
- Apply PCA
- Timing: 2.6s
- Finds 1 cluster (and 1 singleton)



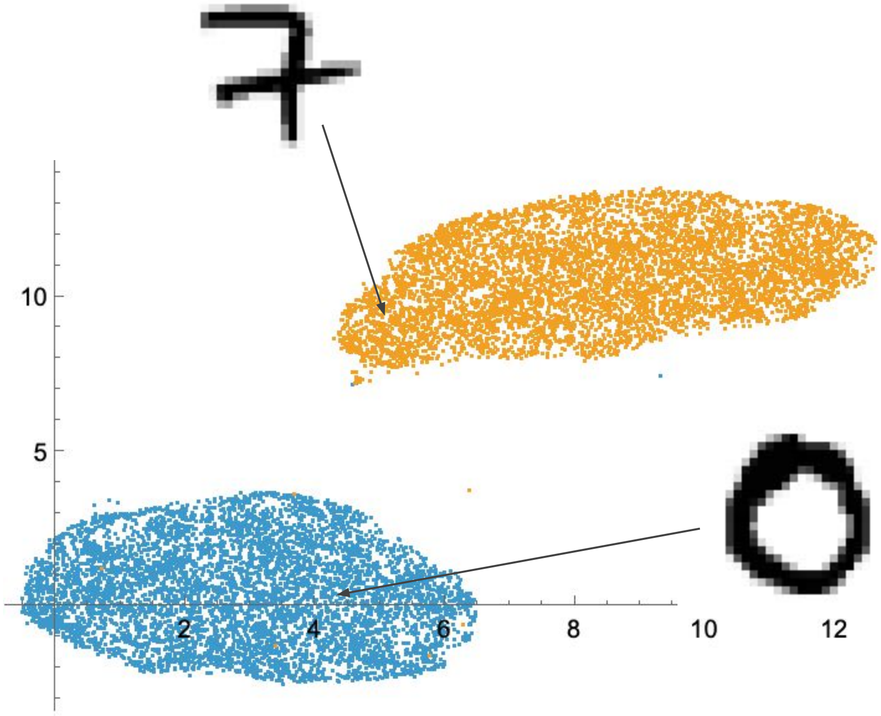
- First apply PCA
- Find 2 clusters
- Timing: 0.5s



Other Dimensional Reduction Algorithms

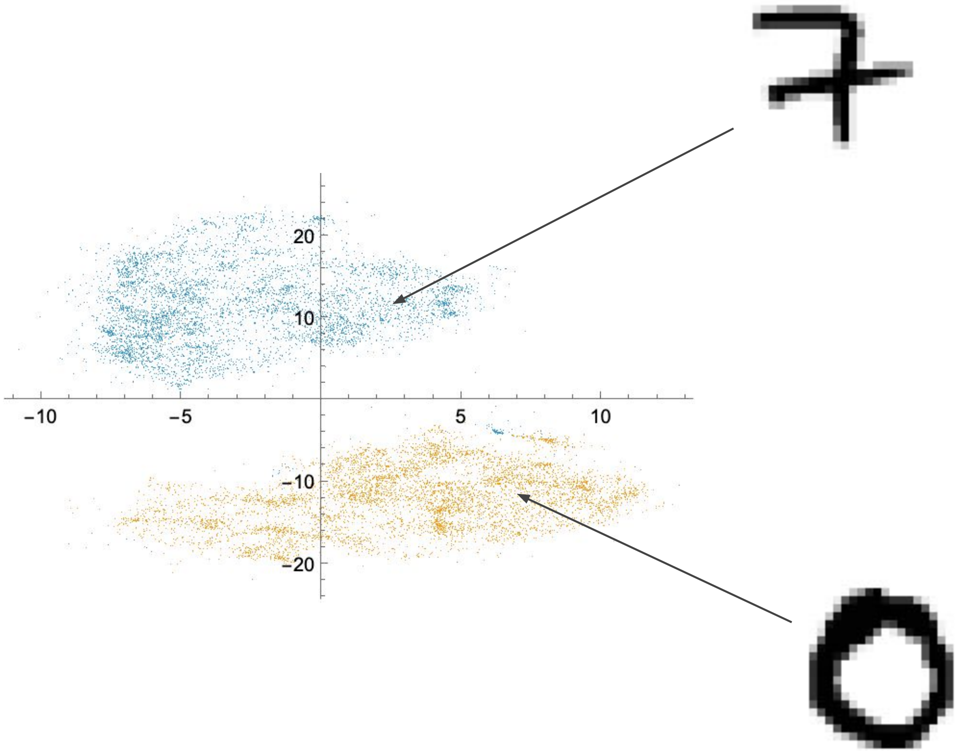
UMAP

UMAP captures both **local** neighborhoods and some **global** relationships in the data.



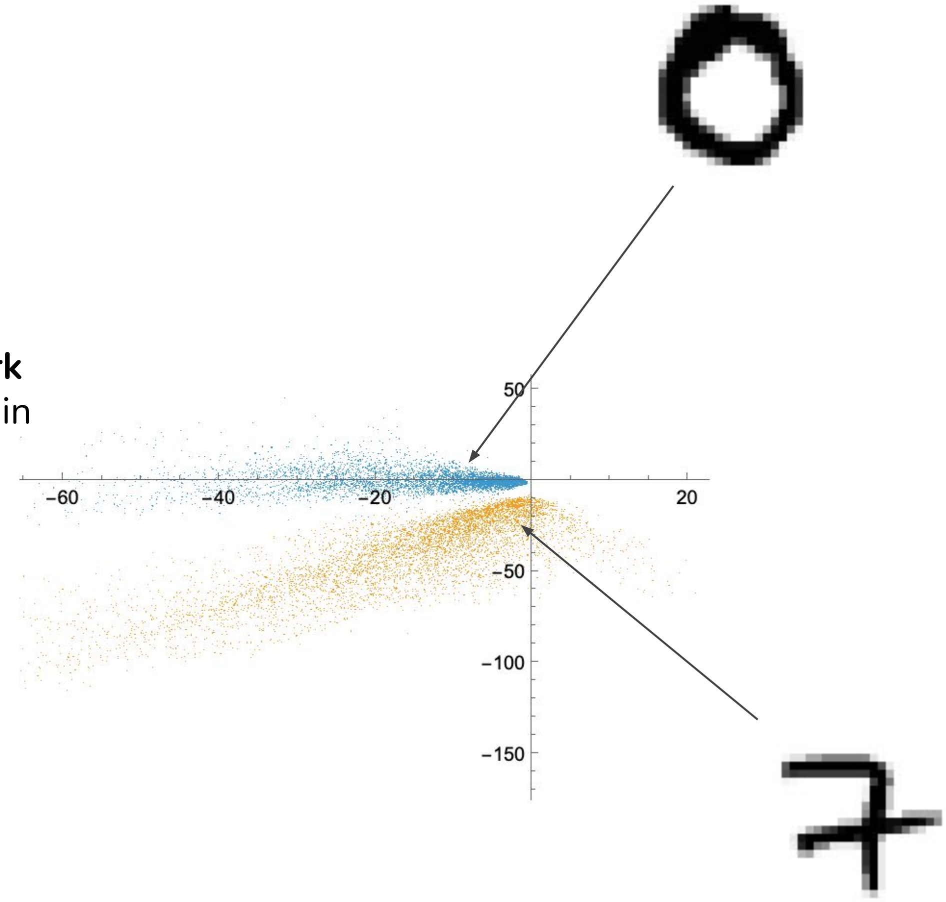
t-SNE

t-SNE is designed to keep similar points close together in the low-dimensional space. It's **excellent at revealing clusters and local groupings** in complex, high-dimensional data.



Autoencoder

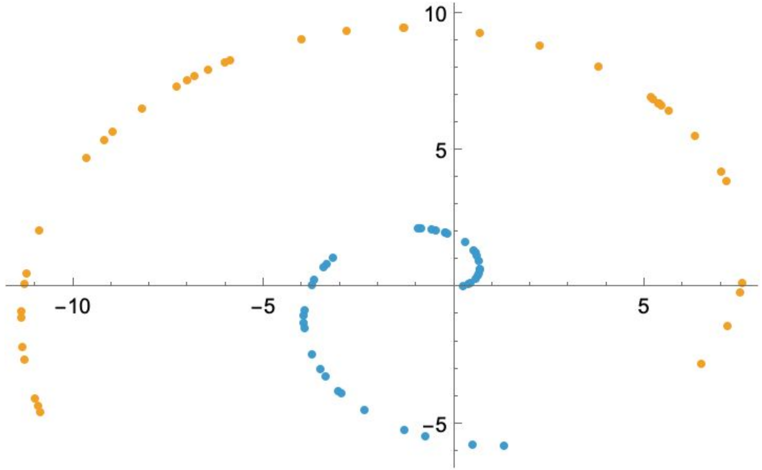
An autoencoder is a type of **neural network** that automatically identifies main features in data



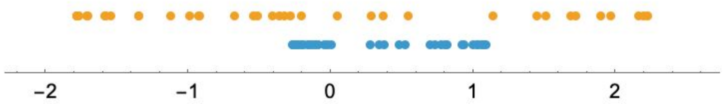
Complicated Geometries

Pathological case I:

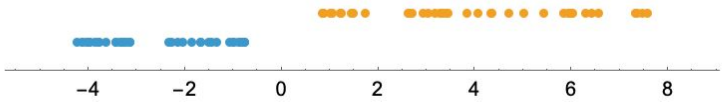
- Data is distributed in a spiral
- t-SNE detects the geometry better than PCA



PCA



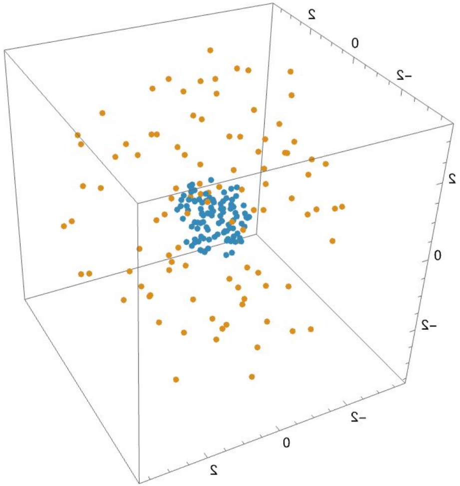
t-SNE



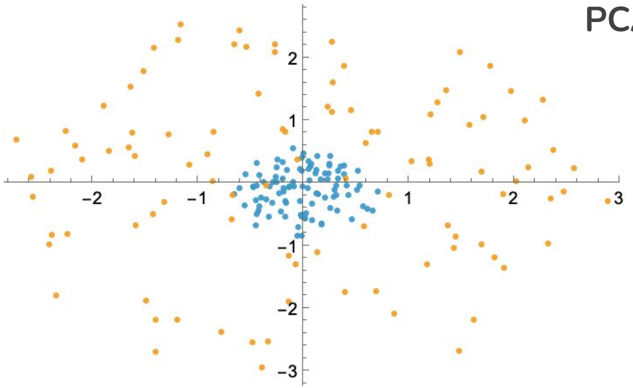
Complicated Geometries

Pathological case II:

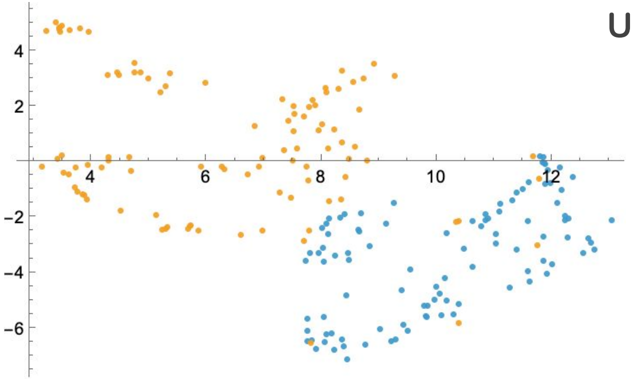
- Data is grouped in nested spheres
- UMap detects the geometry better than PCA



PCA



UMAP



Summary

	PCA	t-SNE	UMAP
Type	Linear	Non-linear	Non-linear
Preserves	Global structure	Local structure	Local & some global
Mathematical Basis	Linear Algebra	Local Topology	Global Topology
Speed	Fast	Slow	Faster than t-SNE
Scalability	Good	Poor	Good
Distance Interpretability	Yes	No	Yes*
Reproducibility	Yes (deterministic)	No (random init)	No (varies slightly)