Dimensionality Reduction

Guillem & Roderic, Summer 2025

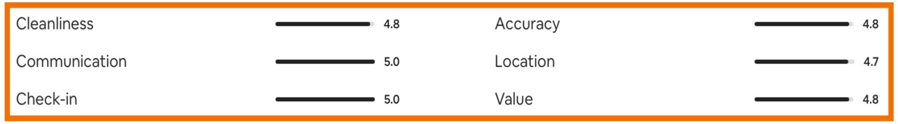
Reminder

- Modeling uses **real world data** to build **models** that **make predictions**
- It involves finding **parameters** that **max./min. objective functions**
  - **Machine learning**:
    - **computers automatically adjust parameters** in a model to improve performance based on examples.
    - Is not explicitly programmed for specific rules.
- Today: **Models to process data** (by changing the dimensionality)

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.

Example 1: Hotel Listings

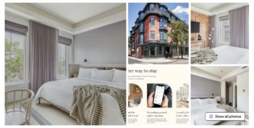


Cleanliness	4.8	Accuracy	4.8
Communication	5.0	Location	4.7
Check-in	5.0	Value	4.8

- 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

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

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



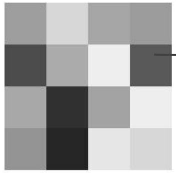
$$= \begin{bmatrix} 4.9 \\ 4.5 \\ 4.7 \\ 4.7 \\ 4.9 \\ 4.5 \end{bmatrix}$$

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

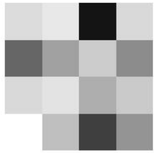
$$= \begin{bmatrix} 4.6 \\ 4.7 \\ 4.7 \\ 4.7 \\ 4.9 \\ 4.5 \end{bmatrix}$$

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 $\mathbb{R}^{16}$



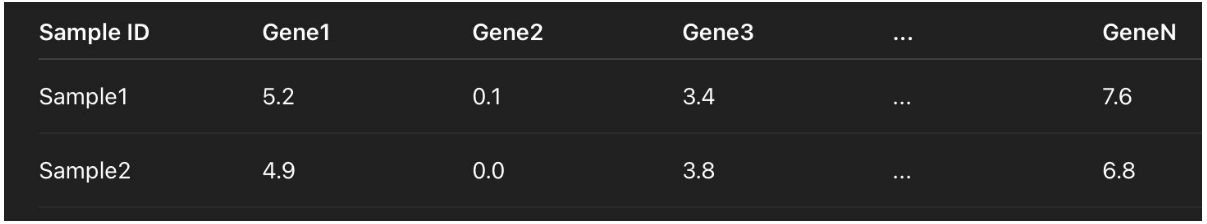
$$\longrightarrow \begin{pmatrix} 0.62 & 0.84 & 0.65 & 0.61 \\ 0.3 & 0.67 & 0.93 & 0.35 \\ 0.66 & 0.19 & 0.64 & 0.93 \\ 0.58 & 0.15 & 0.9 & 0.84 \end{pmatrix} \longrightarrow \begin{pmatrix} 0.62 \\ 0.84 \\ 0.65 \\ 0.61 \\ 0.3 \\ 0.67 \\ 0.93 \\ 0.35 \\ 0.66 \\ 0.19 \\ 0.64 \\ 0.93 \\ 0.58 \\ 0.15 \\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



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

Example 3: Gene Expression

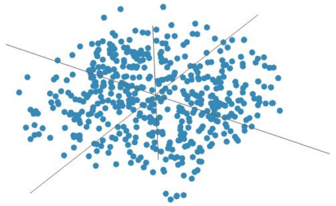
- Each sample measures expressions of N genes in distincts cells
- Each individual cell can be represented by a point in $\mathbb{R}^N$



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

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

Point Cloud Data



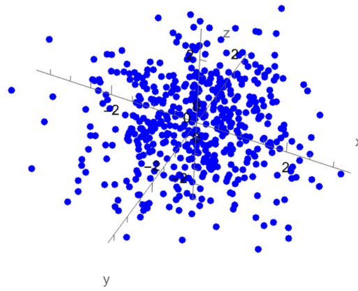
A point cloud is a collection of data points in $\mathbb{R}^m$

- 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 $\mathbb{R}^6$
  - A **4x4 pixels grayscale image** can be represented in $\mathbb{R}^{16}$
  - Samples of **expressions of 10 genes** in cells can be represented in $\mathbb{R}^{10}$

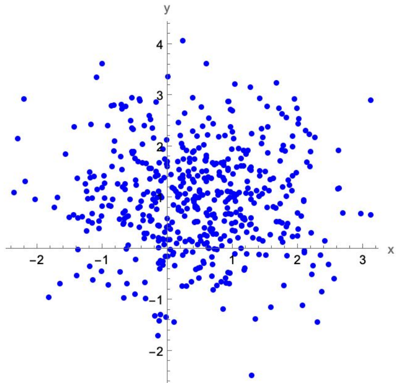
Why Dimensional Reduction?

- 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?

Point Cloud in R3



Point Cloud in R2



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

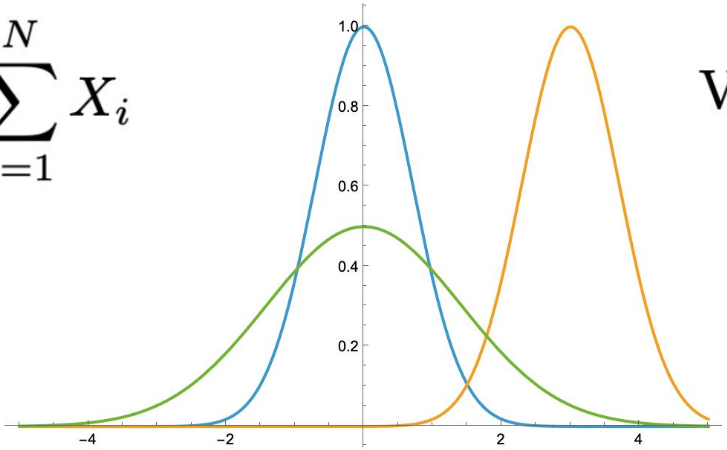
Repeated measurements of a certain feature: $X_1, X_2, \ldots, X_N$

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:

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



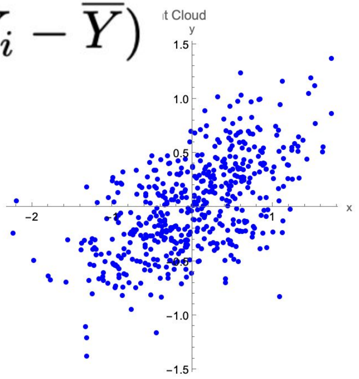
Statistics Measurements

N measurements of two features:
$$X_1, X_2, \ldots, X_N$$
$$Y_1, Y_2, \ldots, Y_N$$

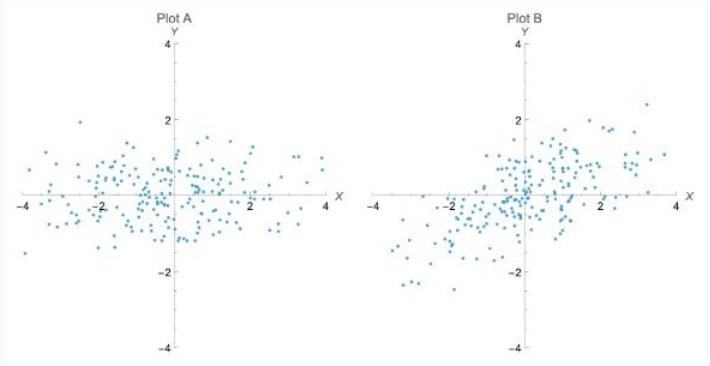
The **covariance** is a measure of how two variables change together—whether they tend to increase or decrease at the same time.

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

correlation ≈ covariance



What is true about these plots?



$\mathrm{Var}(X_A) >> \mathrm{Var}(X_B)$

0%

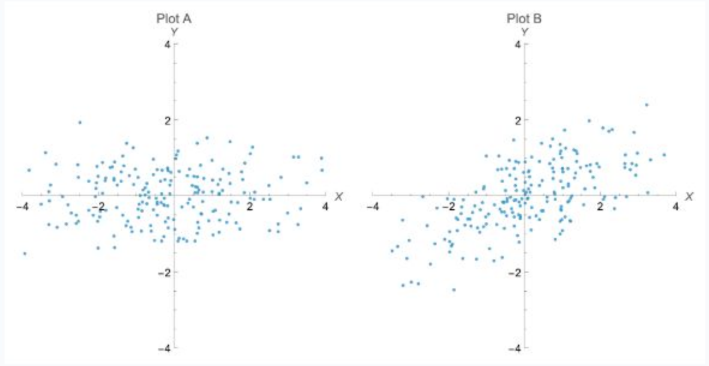
$\mathrm{Var}(X_A) << \mathrm{Var}(X_B)$

0%

$\mathrm{Var}(X_A) \simeq \mathrm{Var}(X_B)$

0%

What is true about these plots?



$\mathrm{Cov}(X_A, Y_A) >> \mathrm{Cov}(X_B, Y_B)$

0%

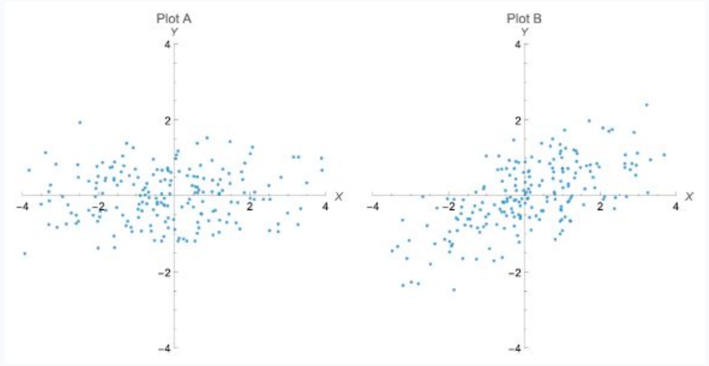
$\mathrm{Cov}(X_A, Y_A) << \mathrm{Cov}(X_B, Y_B)$

0%

$\mathrm{Cov}(X_A, Y_A) \simeq \mathrm{Cov}(X_B, Y_B)$

0%

What is true about these plots?



$\mathrm{Cov}(X_A, Y_A) > 0$

0%

$\mathrm{Cov}(X_A, Y_A) < 0$

0%

$\mathrm{Cov}(X_A, Y_A) \simeq 0$

0%

Linear Algebra Basics

Linear Transformations

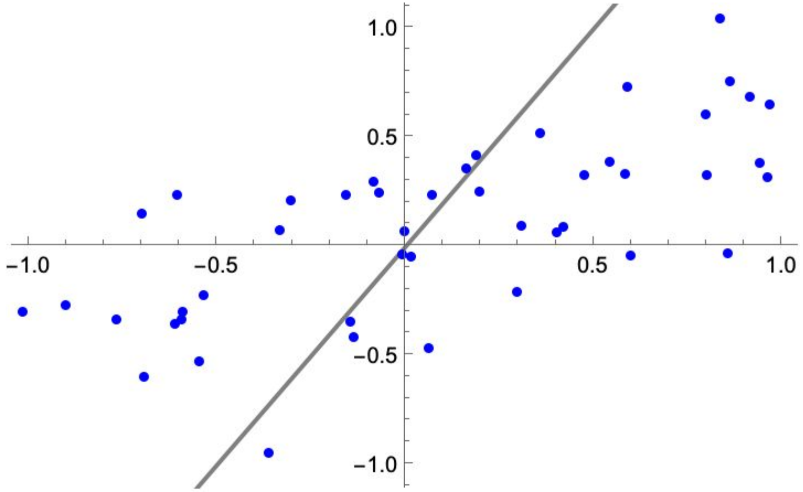
Any matrix can be thought of as a function via matrix-vector multiplication. A matrix with m columns and n rows “is” a function from $\mathbb{R}^m$ to $\mathbb{R}^n$.

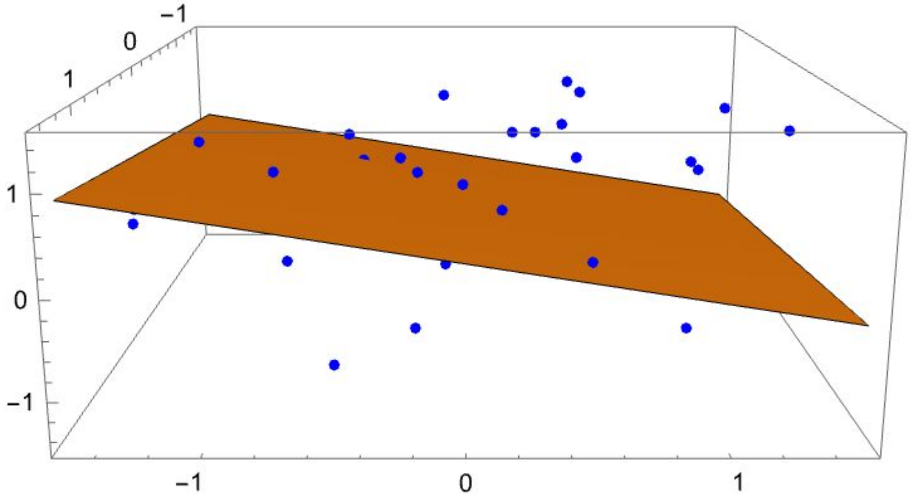
- Example of a linear transformation from $\mathbb{R}^3$ to $\mathbb{R}^2$:

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

Orthogonal Projections

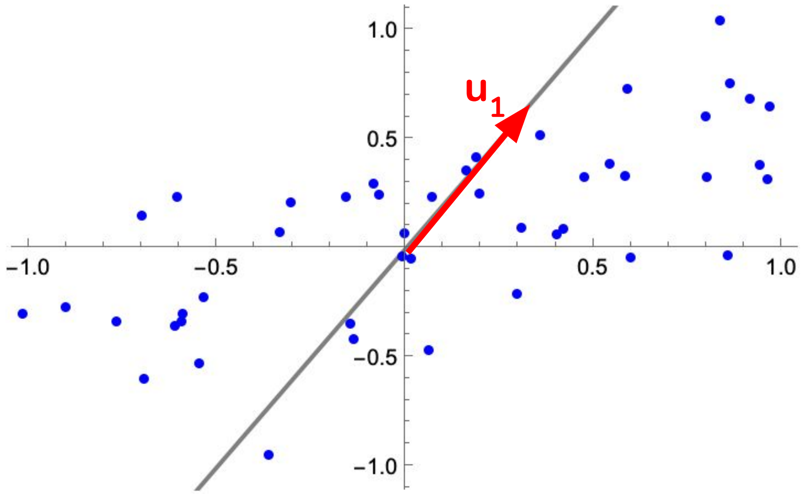
- Are linear transformations (defined by a matrix)
- Project data onto a k-dimensional subspace of $\mathbb{R}^n$ (line in $\mathbb{R}^2$, line or plane in $\mathbb{R}^3$, etc.)

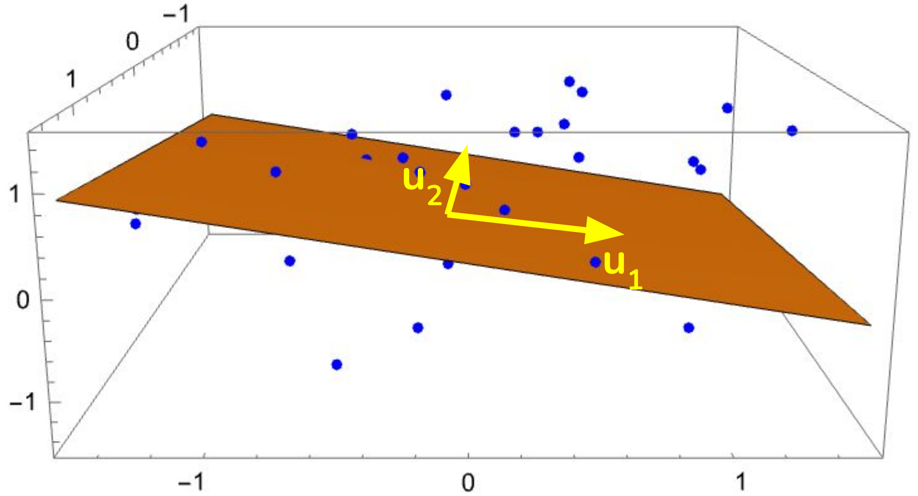




Orthogonal Projections

- 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)

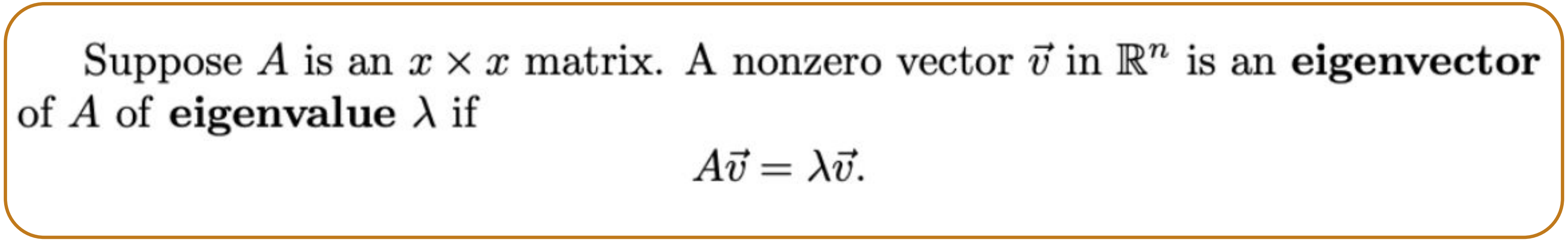




Eigenvalues & Eigenvectors

Eigenvectors are specific vectors of square (nxn) matrices.

- Eigenvectors scale by a constant when transformed.
- Eigenvalues can be real or complex.
- 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

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

Which of these is an eigenvalue-vector pair for the matrix $\begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$

0

$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\lambda = 1$

0%

$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$ and $\lambda = 5$

0%

$\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ and $\lambda = 0$

0%

Symmetry, SVD & Eigendecomposition

- A square matrix M is symmetric is M^T=M
- Symmetric matrices satisfy
  - All eigenvalues are real
  - 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

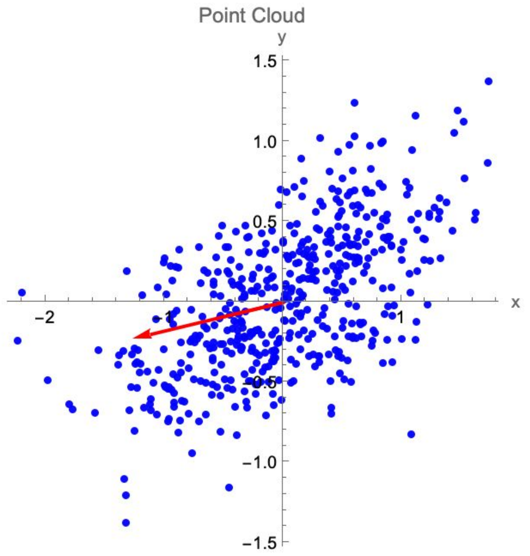
Linear Algebra Summary

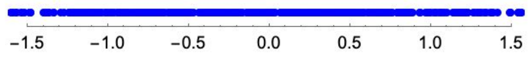
- **Projections** are easy and interpretable transformations
- **Symmetric Matrices** have nice properties
- **PCA** uses symmetric matrix, eigenvectors and projections to reduce the dimension of the data.

Principal Components Analysis

Principal Component Analysis

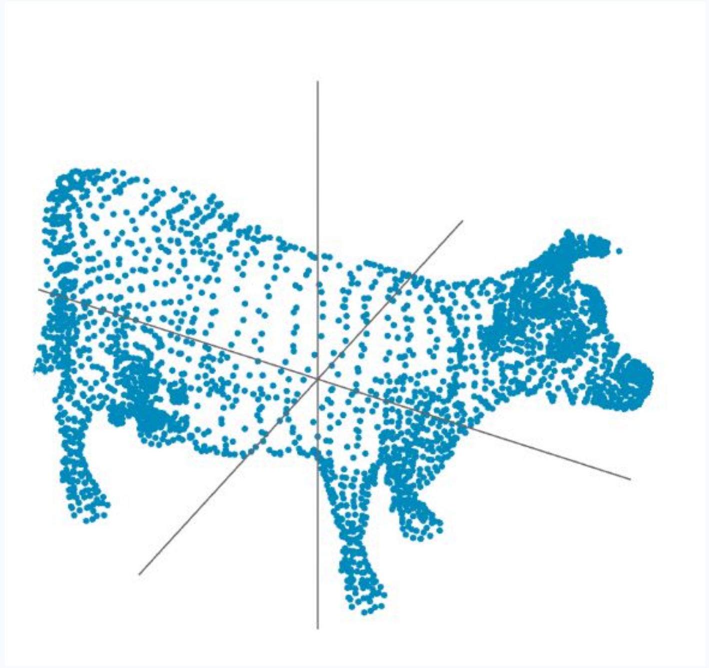
- PCA **finds the direction(s) in which maximize the variance,** and
- **projects the data onto those directions** to reduce dimensionality while preserving as much variance as possible.

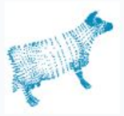




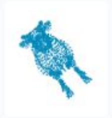
Which of the following is the PCA 2D projection of this cow?

0

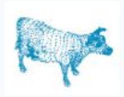




0%



0%



0%

Step 0: Gathering The Data

- Dataset consisting of **N** observations of **m** different characteristics X_i :

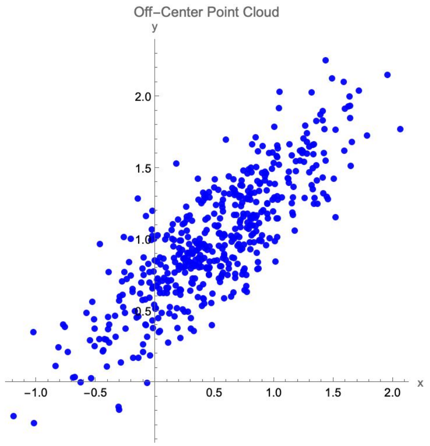
$$(X_1^{(1)}, X_2^{(1)}, X_3^{(1)}, \ldots, X_m^{(1)})$$
$$(X_1^{(2)}, X_2^{(2)}, X_3^{(2)}, \ldots, X_m^{(2)})$$
$$\vdots$$
$$(X_1^{(N)}, X_2^{(N)}, X_3^{(N)}, \ldots, X_m^{(N)})$$

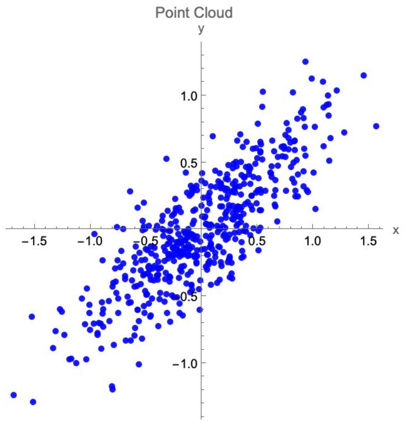
- Data lives in a high-dimensional space $\mathbb{R}^m$, that is “impossible” for us to visualize

Step 1: Standardizing The Data

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

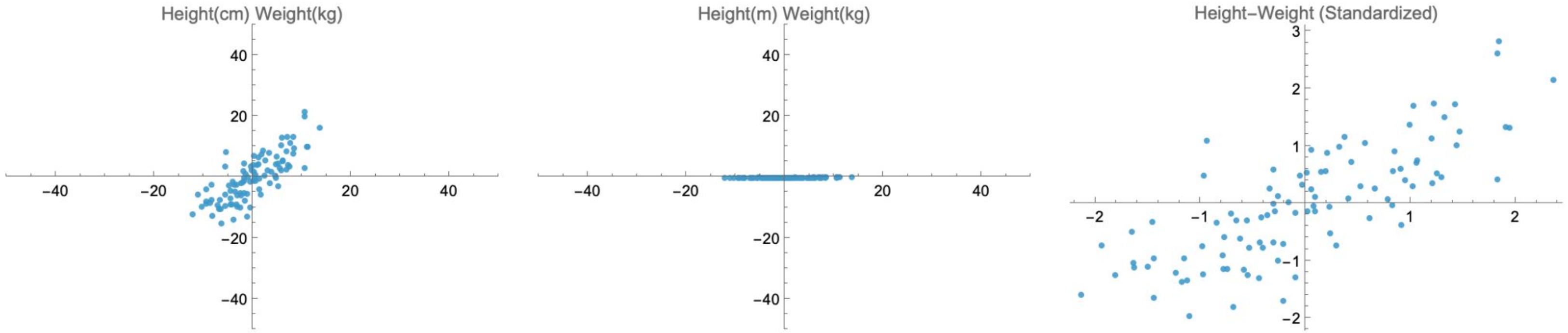
- Point cloud before/after centering





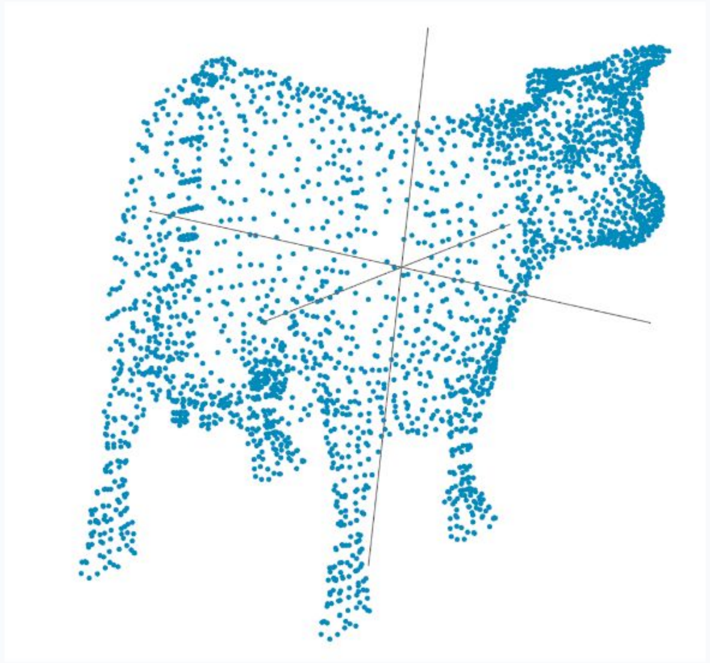
Step 1: Standardizing The Data

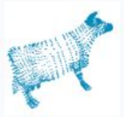
- Visual: why is it important to standardize?
  - Remove dependency on units
  - Get rid of scaling differences



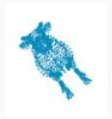
Height vs weight of a 100 person sample

Which of the following is the PCA 2D projection of the *standardized* cow?

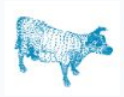




0%



0%



0%

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

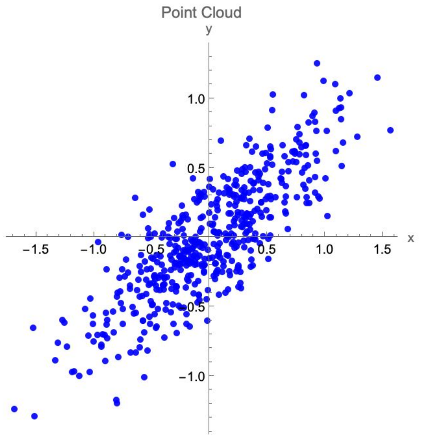
When data is standardized, the eigenvalues of the covariance matrix $\mathrm{Cov}(\vec{X})$ measure the proportion of the variance in the direction of the corresponding eigenvectors:

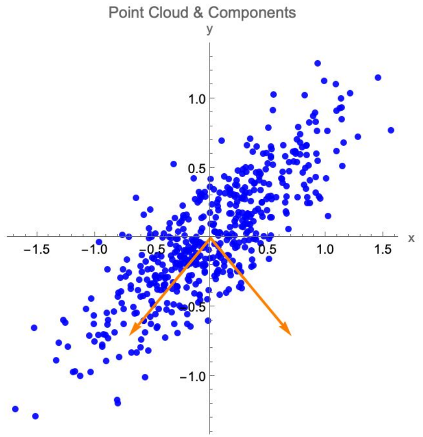
$$\sum_{i=1}^{N} \lambda_i = \sum_{i=1}^{N} \mathrm{Var}(X_i) = N$$

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 of principal components to project onto
- Pick the eigenvectors with the largest eigenvalues

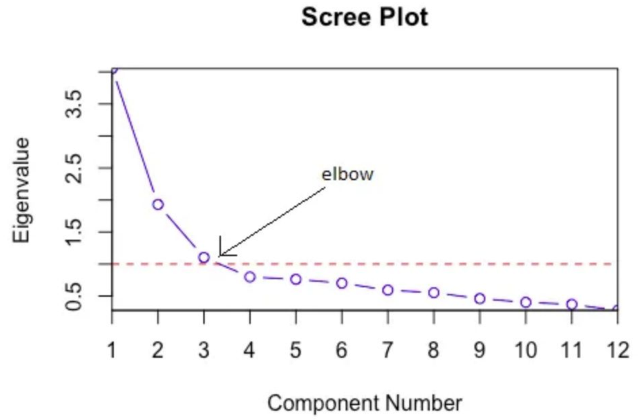




Step 4: Choosing Number of Principal Components

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

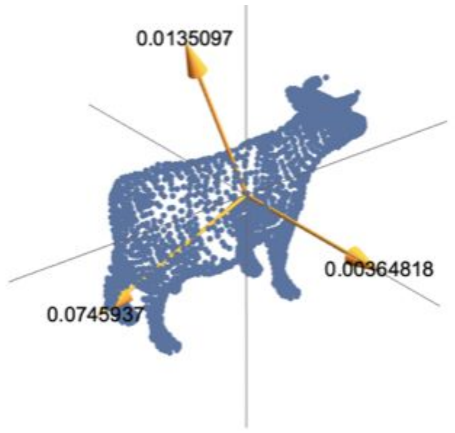


Step 5: Project The Data

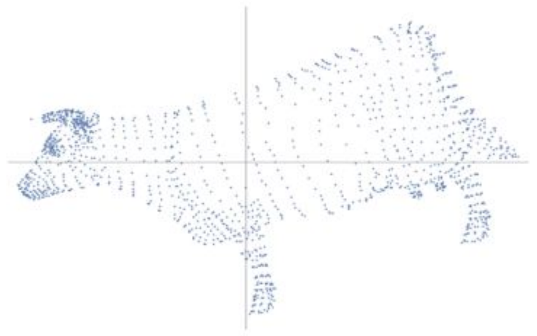
Example: a cow from $\mathbb{R}^3$ to $\mathbb{R}^2$

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

$$\mathbb{R}^3 \xrightarrow{\qquad\qquad} \mathbb{R}^2$$



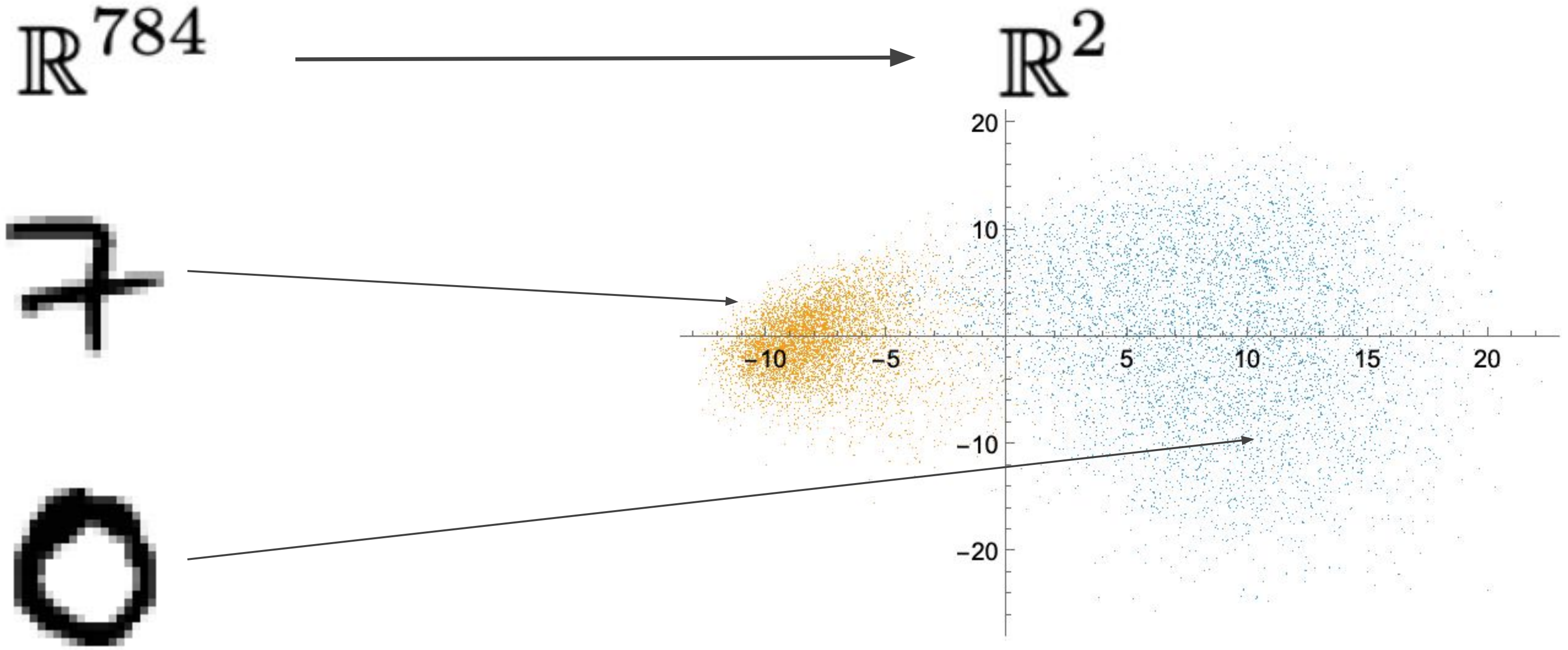
3D Cow



2D Cow

Step 5: Project The Data

Example: Classifying 28x28 handwritten digits



Step 6: Interpretation

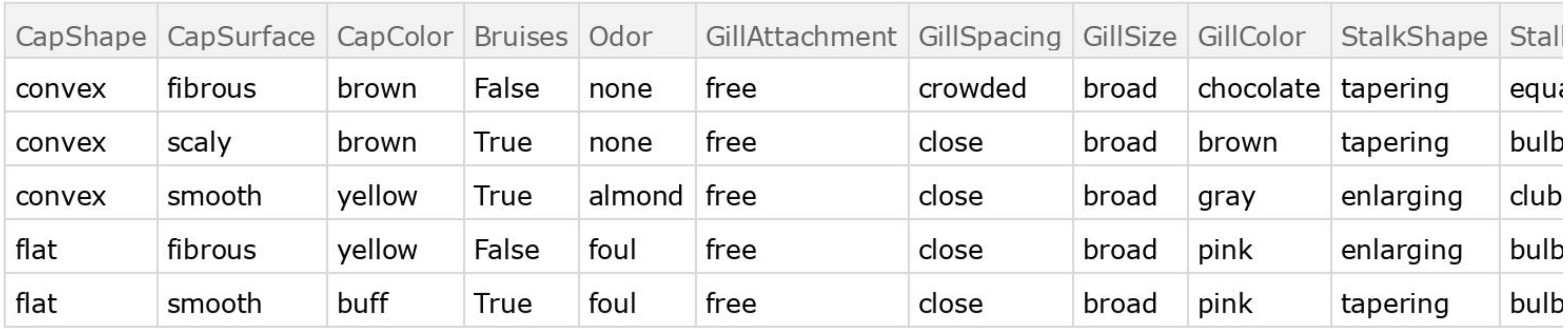
- Let $X_1, X_2, \dots, X_n$ in denote the original features in $\mathbb{R}^n$ and $Y_1, Y_2, \dots, Y_d$ the principal components in $\mathbb{R}^d$:

  - $Y_1 = a_{11}X_1 + a_{12}X_2 + \dots + a_{1n}X_n$
  - $Y_2 = a_{21}X_1 + a_{22}X_2 + \dots + a_{2n}X_n$
  - ...
  - $Y_d = a_{d1}X_1 + a_{d2}X_2 + \dots + a_{dn}X_n$

Combinations of the **original features** that are more relevant (preserve the maximum variance of the original data)

Example: Identifying Mushrooms

- Predicting whether a mushroom is edible or poisonous according to 22 binary characteristics:



CapShape	CapSurface	CapColor	Bruises	Odor	GillAttachment	GillSpacing	GillSize	GillColor	StalkShape	Stal
convex	fibrous	brown	False	none	free	crowded	broad	chocolate	tapering	equa
convex	scaly	brown	True	none	free	close	broad	brown	tapering	bulb
convex	smooth	yellow	True	almond	free	close	broad	gray	enlarging	club
flat	fibrous	yellow	False	foul	free	close	broad	pink	enlarging	bulb
flat	smooth	buff	True	foul	free	close	broad	pink	tapering	bulb

```
{convex, smooth, brown, True, pungent, free, close, narrow, black, enlarging, equal, smooth,
  smooth, white, white, partial, white, one, pendant, black, scattered, urban} → poisonous
{convex, smooth, yellow, True, almond, free, close, broad, black, enlarging, club, smooth,
  smooth, white, white, partial, white, one, pendant, brown, numerous, grasses} → edible
```

Which characteristics are more distinguishing of mushrooms?

0

Cap Shape: convex, bell, sunken or flat

Gill color: black, brown, gray, pink, white, chocolate, purple, red, buff, green, yello…

Ring number: 1,2 or 3

Stalk surface: smooth, fibrous, scaly or silky

Example: Identifying Mushrooms

- Which characteristics are more distinguishing between mushrooms?
- First principal component:

$$Y_1 = 0.48 * gill\_color + 0.38 * spore\_print\_color + 0.35 * stalk\_color\_above\_ring + \ldots$$

Which characteristics are more distinguishing of mushrooms?

✓ 0

Cap Shape: convex, bell, sunken or flat	0%
Gill color: black, brown, gray, pink, white, chocolate, purple, red, buff, green, yellow or orange	0%
Ring number: 1,2 or 3	0%
Stalk surface: smooth, fibrous, scaly or silky	0%

Which characteristics are more distinguishing of mushrooms?

✓ 0

Cap Shape: convex, bell, sunken or flat

0%

Gill color: black, brown, gray, pink, white, chocolate, purple, red, buff, green, yellow or orange

0%

Ring number: 1,2 or 3

0%

Stalk surface: smooth, fibrous, scaly or silky

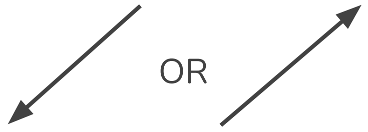
0%

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

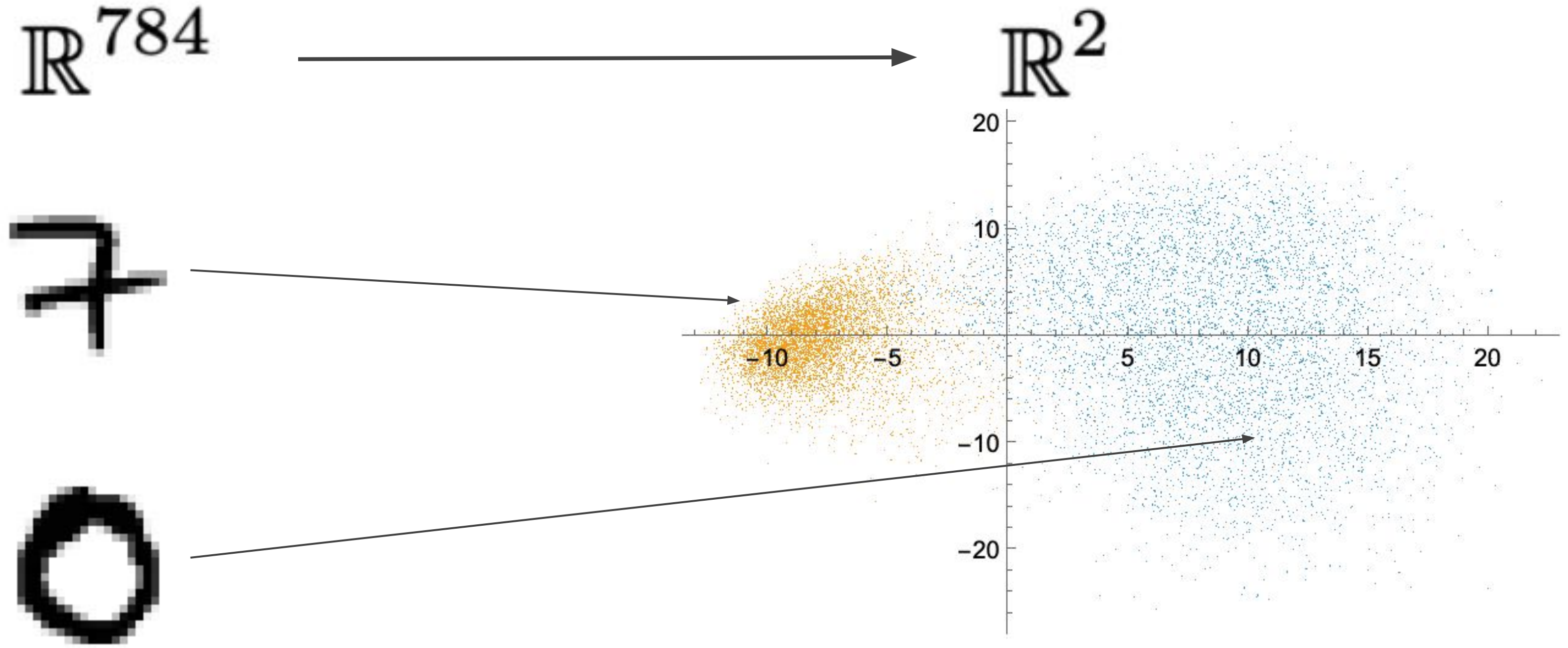
- Built-in algorithms will center your data, but (typically) won’t standardize it.
- There is a sign ambiguity when choosing the eigenvectors.



Questions?

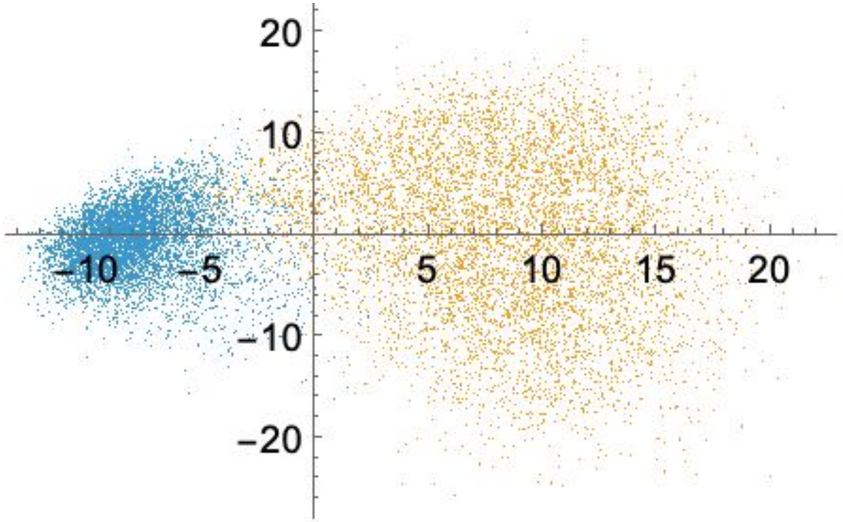
PCA + Other Algorithms

Running Example: Classifying 0s and 7s

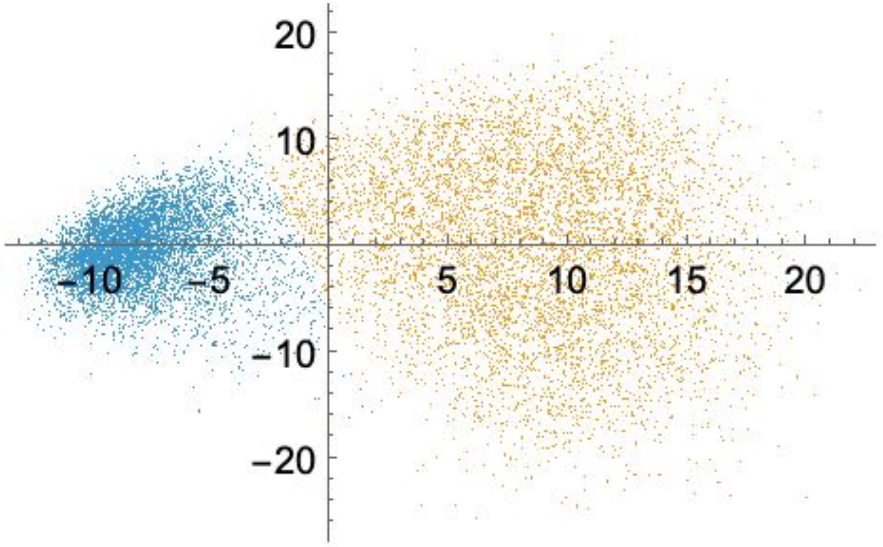


PCA + Logistic Regression

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

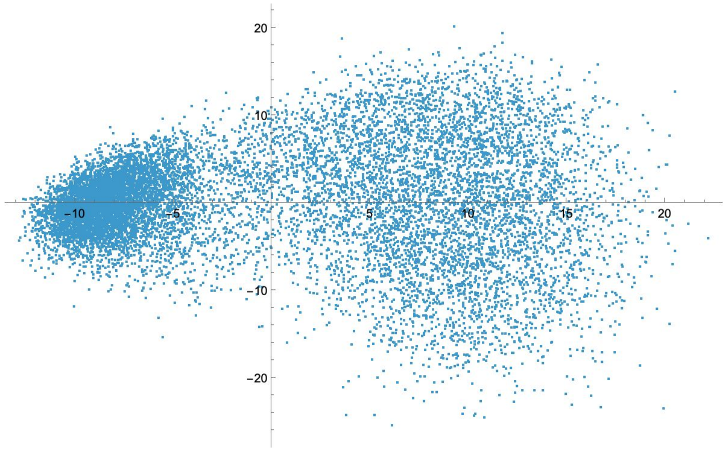


- First apply PCA
- Then run Logistic Regression
- Timing: 1.9s

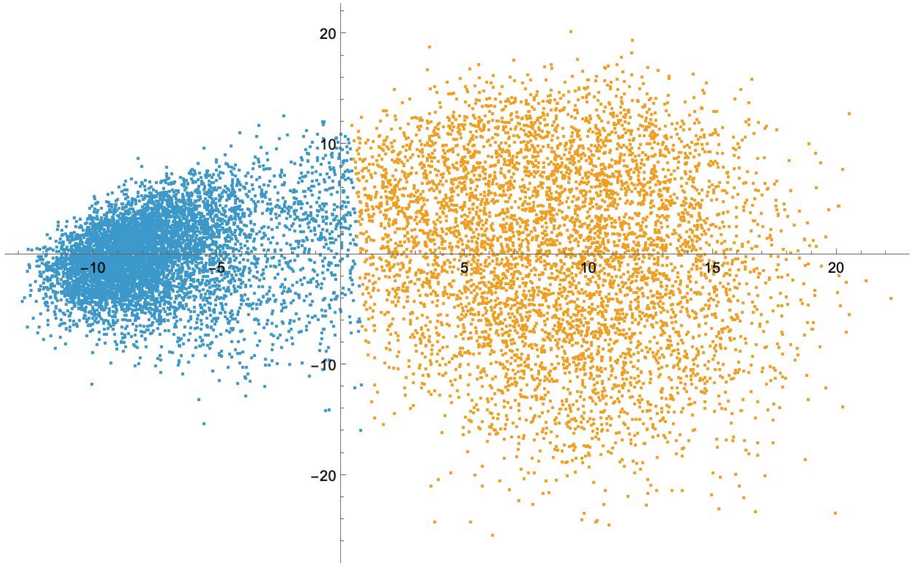


PCA + Clustering

- 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

Local vs Global Structure

Local structure captures relationships between a point and its **nearby neighbors**.

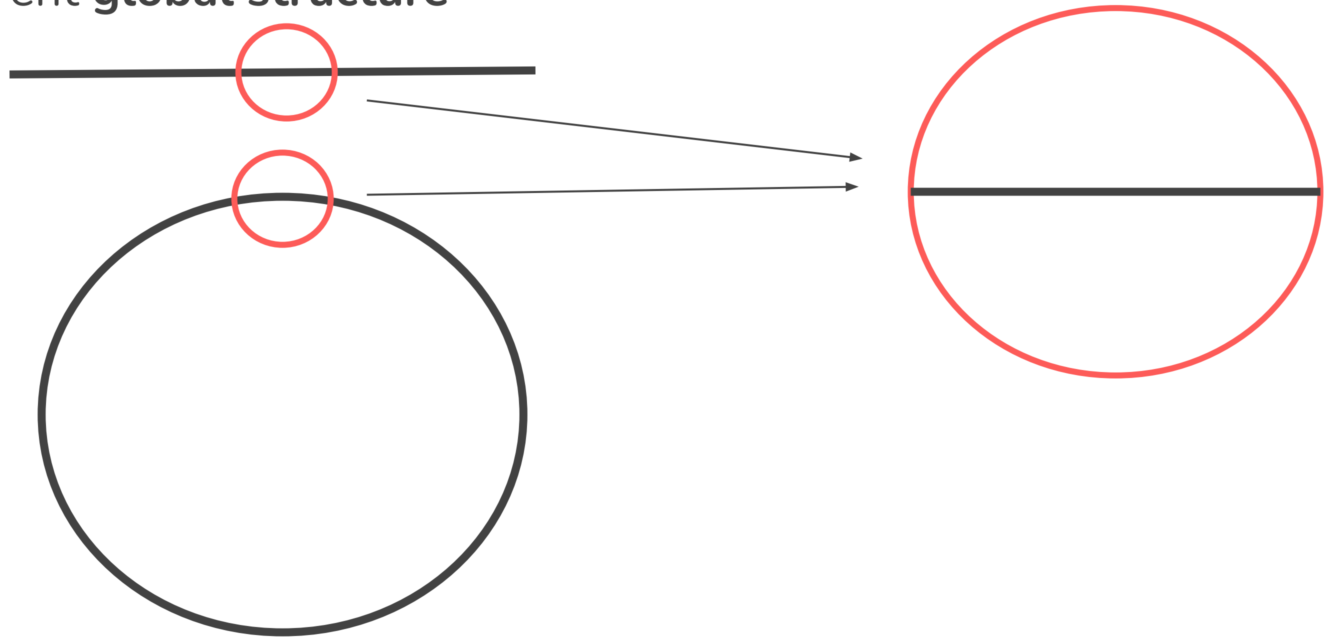
- Focuses on **small-scale geometry**.
- Captures **neighborhoods**, **clusters**, **density**, or **local curvature**.

Global structure refers to the **overall layout and geometry** of the entire dataset.

- Focuses on **long-distance relationships**.
- Captures **shape, orientation, cluster separation**, and **topology**.

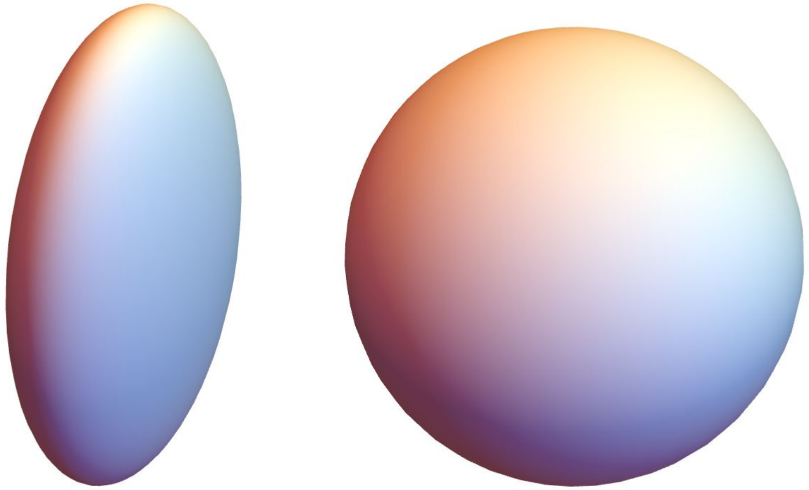
Local vs Global Structure

A circle and a line have the same **local structure**, but different **global structure**



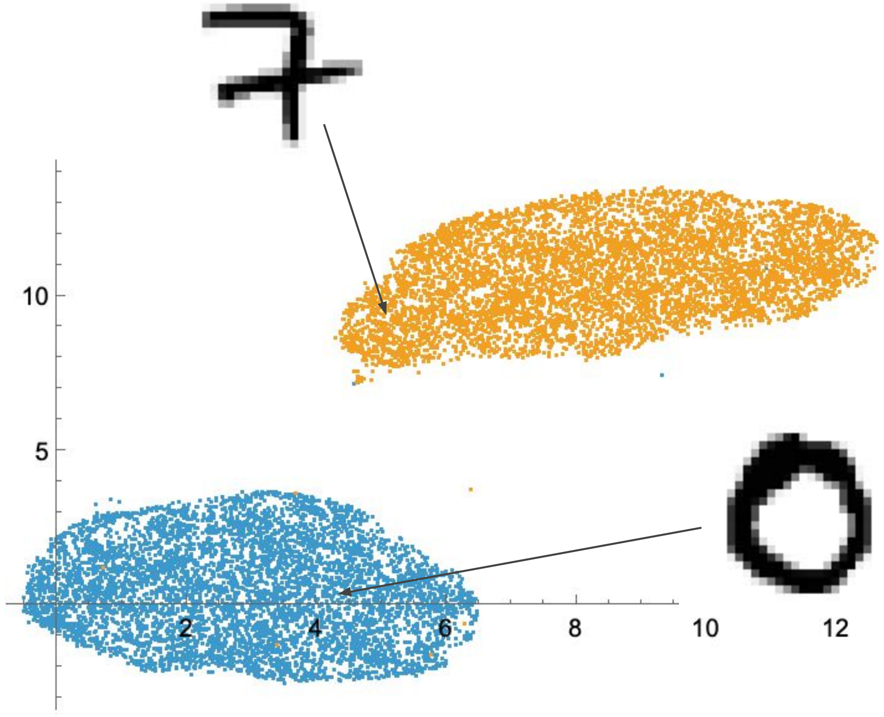
Local vs Global Structure

A sphere and an ellipsoid have the same **global structure**.



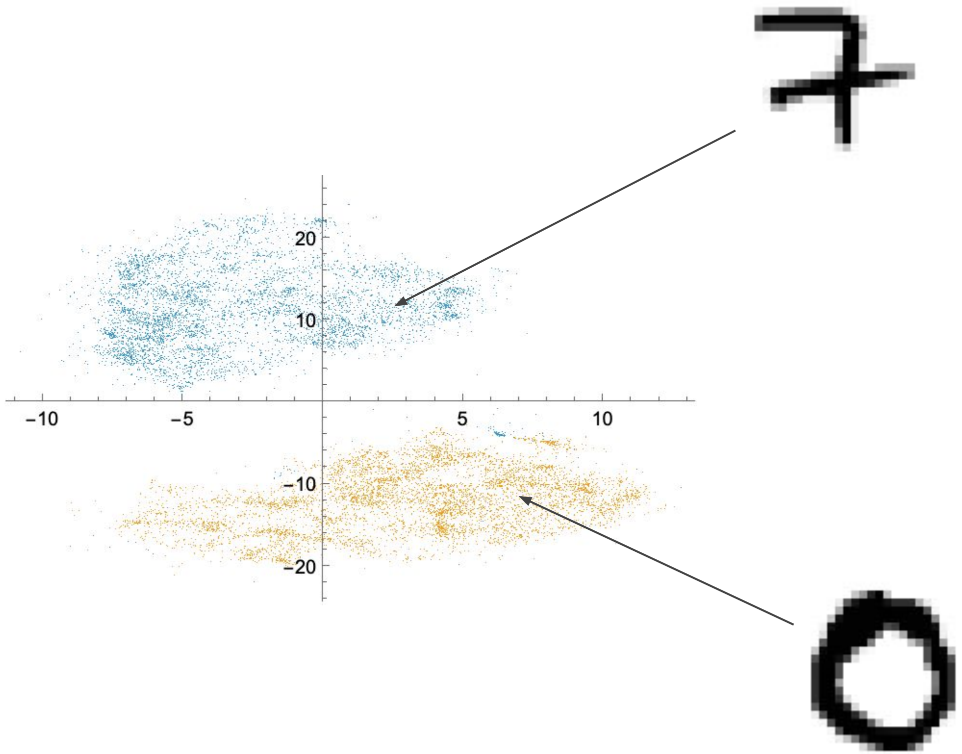
UMAP

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



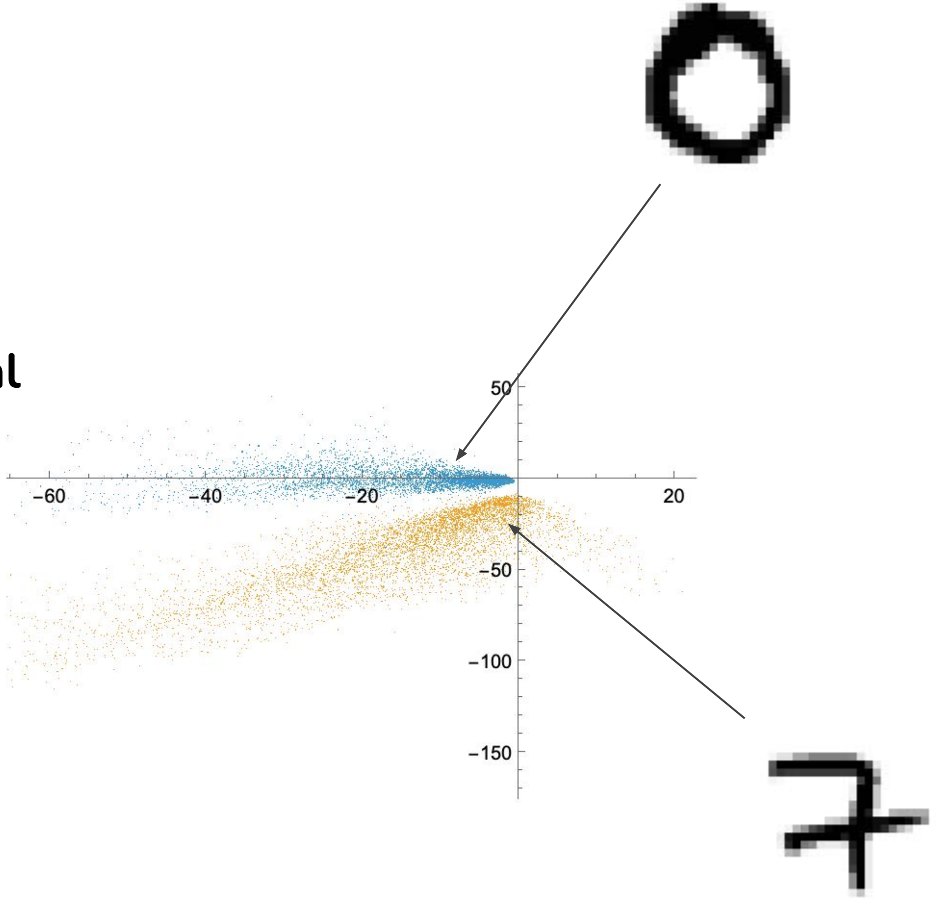
t-SNE

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

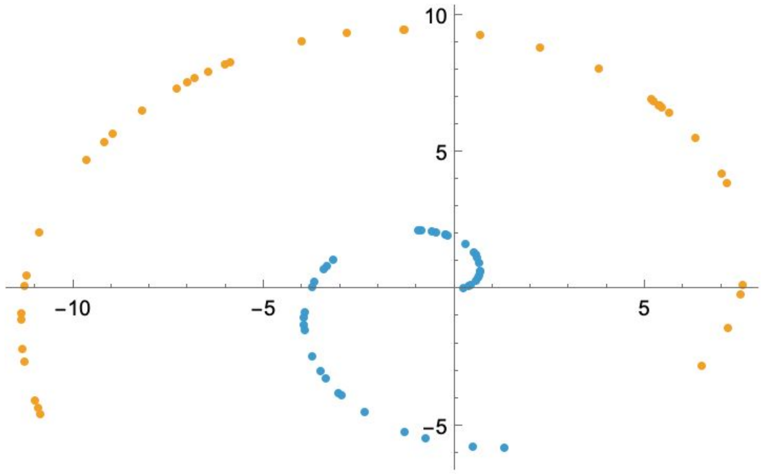
An autoencoder is a type of **neural network** that automatically identifies main features in data



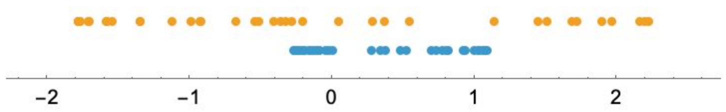
Complicated Geometries

Pathological case I:

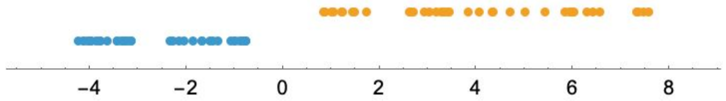
- Data is distributed in a spiral
- t-SNE detects the geometry better than PCA



PCA



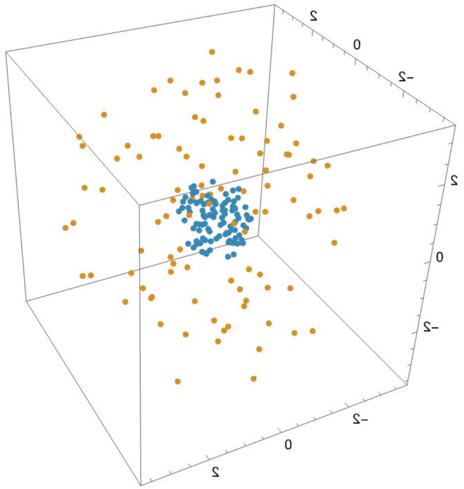
t-SNE



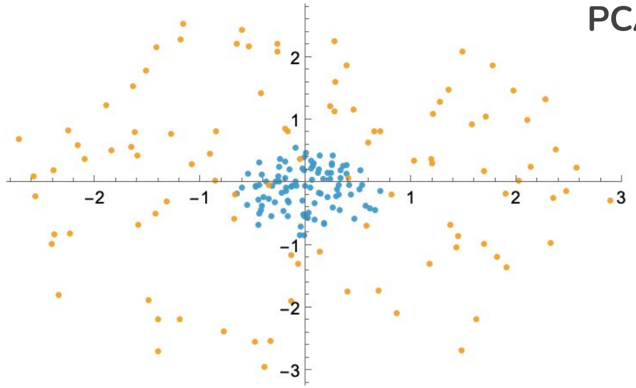
Complicated Geometries

Pathological case II:

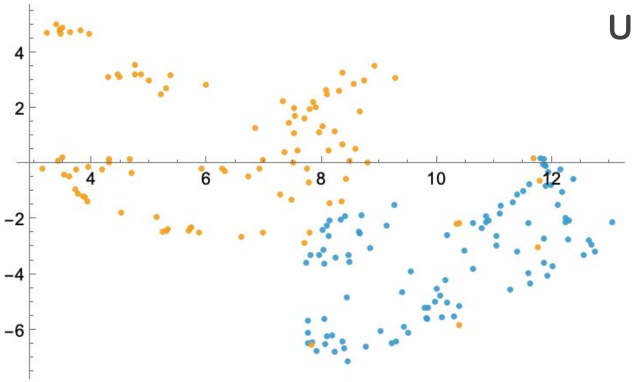
- Data is grouped in nested spheres
- UMap differentiates better than PCA



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)