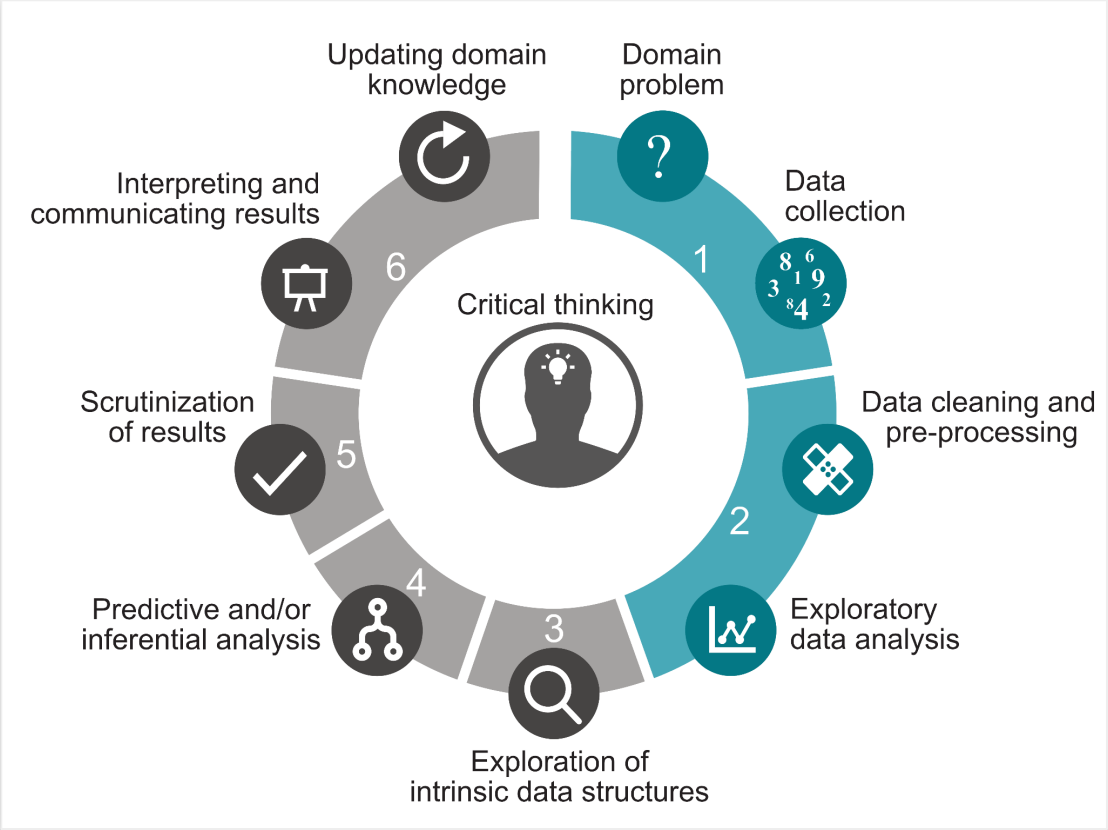
Introduction to Unsupervised Learning

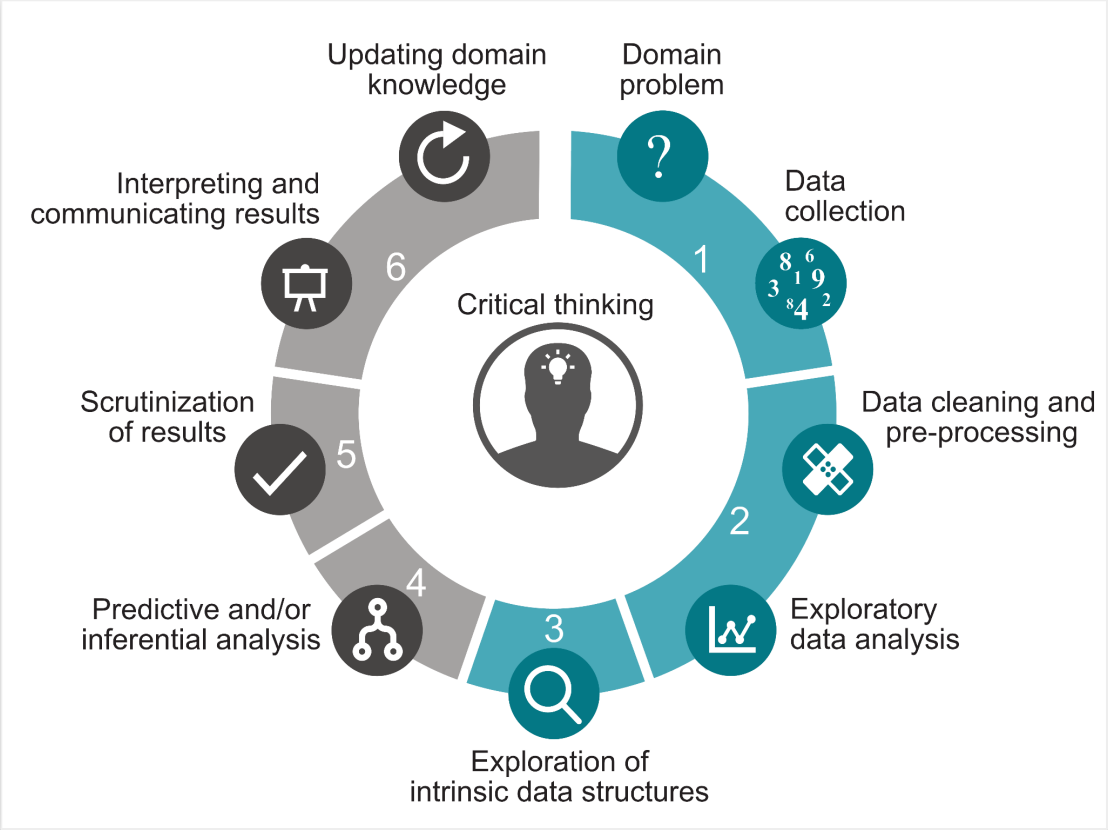
February 5, 2026

The Big Picture: Data Science Life Cycle



Credits: Veridical Data Science (VDS) Textbook

The Big Picture: Data Science Life Cycle



Credits: Veridical Data Science (VDS) Textbook

Plan for Unsupervised Learning Unit

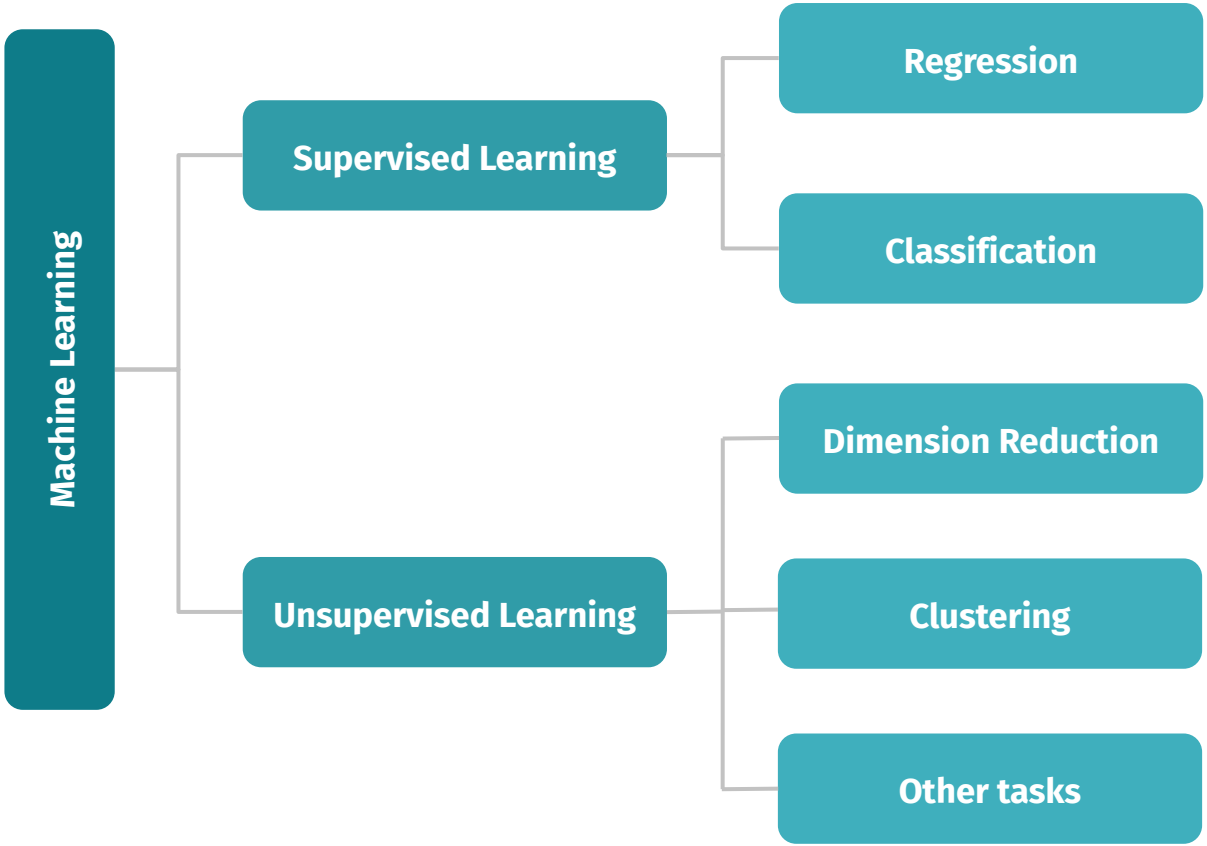
1. Supervised vs **Unsupervised** Learning
2. Overview of Popular **Dimension Reduction** Methods
3. Overview of Popular **Clustering** Methods
4. **Model Selection** and **Evaluation**
5. **In-Class Lab:** Linguistics Data

Today's Plan

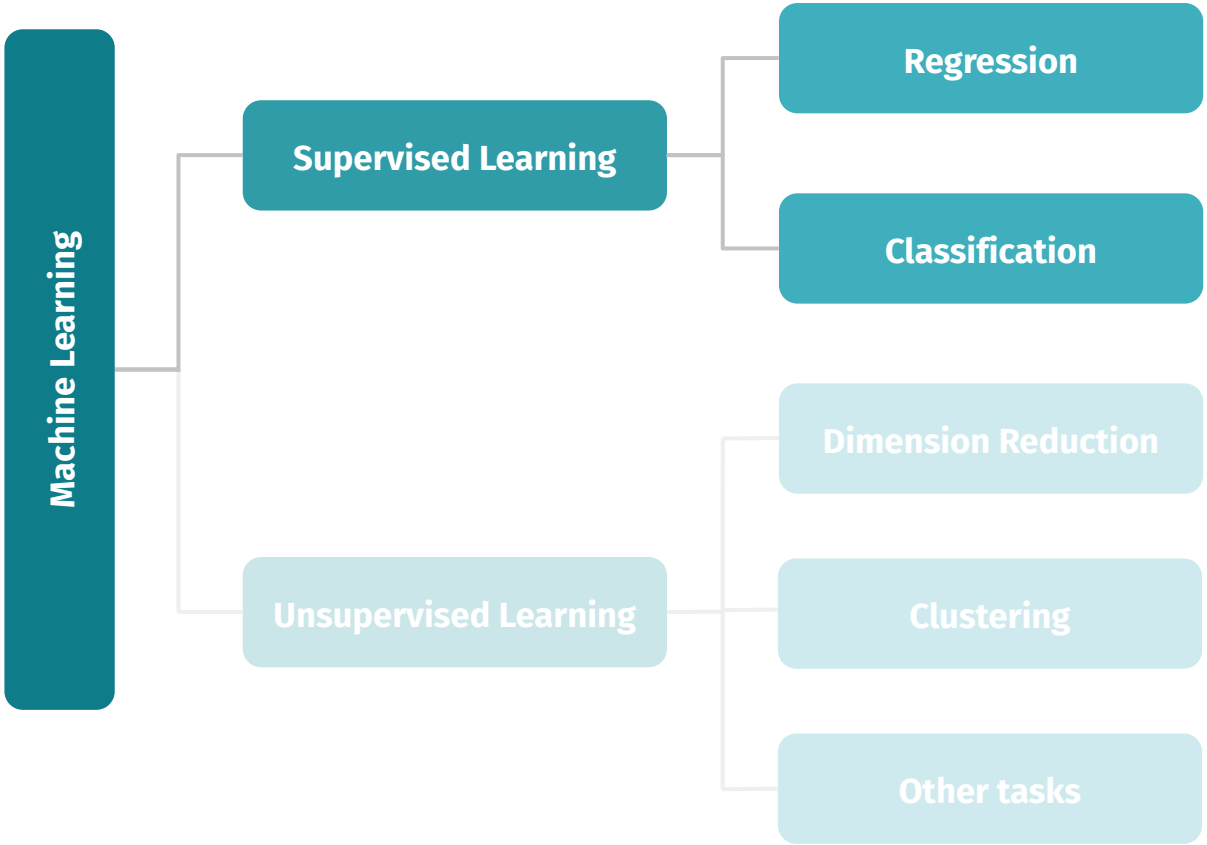
1. Supervised vs **Unsupervised** Learning
2. Overview of Popular **Dimension Reduction** Methods
3. Overview of Popular **Clustering** Methods
4. **Model Selection** and **Evaluation**
5. **In-Class Lab:** Linguistics Data

Unsupervised vs Supervised Learning

Overview of Machine Learning Terminology

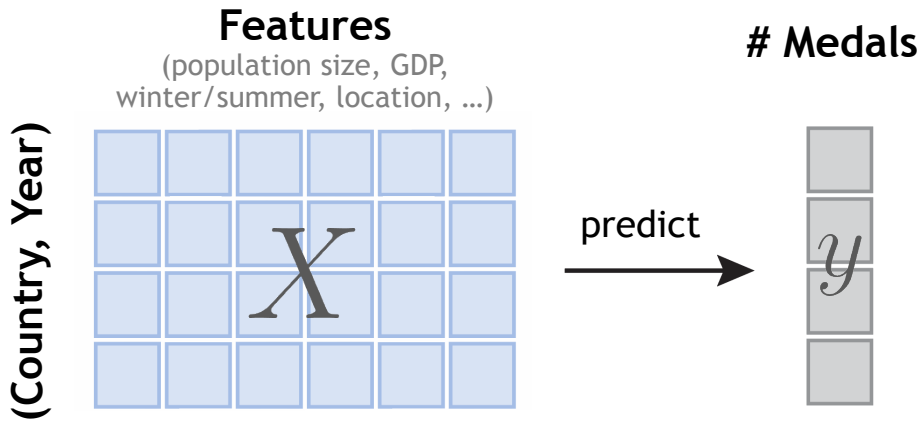


Overview of Machine Learning Terminology

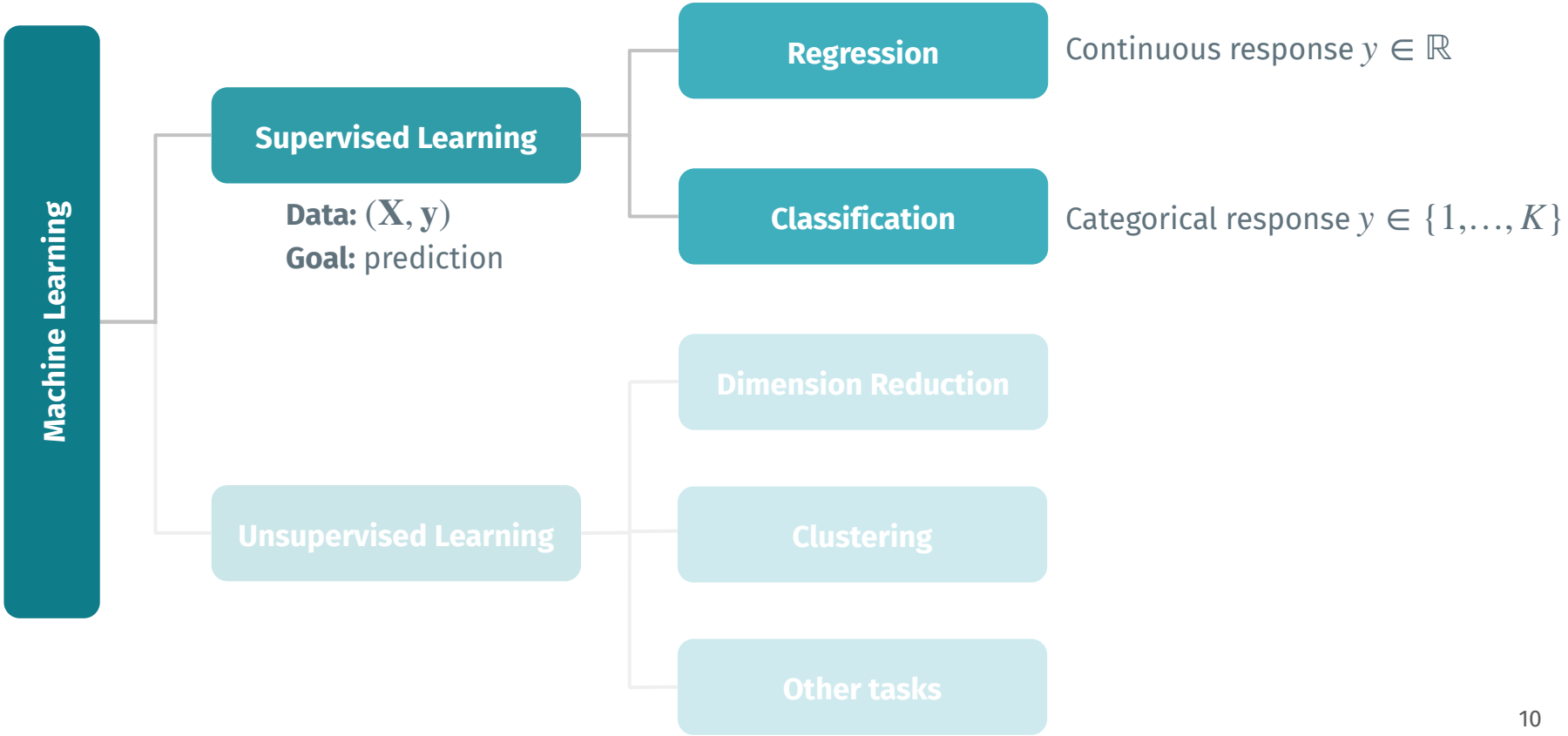


Supervised Learning

- **Data:** features/covariates $\mathbf{X}$ and response $\mathbf{y}$
- **Goal:**
  1. Prediction
  2. Interpretability (i.e., understanding the relationship between $\mathbf{X}$ and $\mathbf{y}$)
- **Example:** predicting Olympic medal count

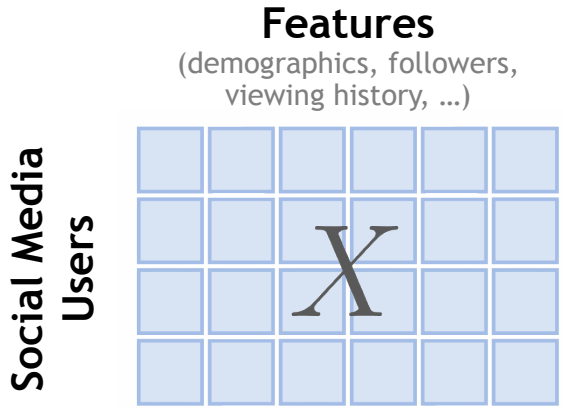


Overview of Machine Learning Terminology



Unsupervised Learning

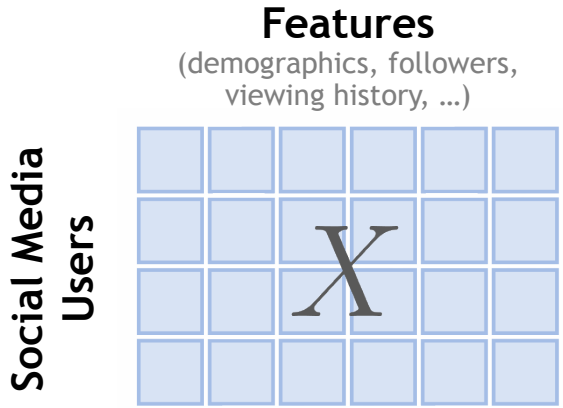
- **Data:** features/covariates $\mathbf{X}$ ~~and response $\mathbf{y}$~~
- **Goal:**
  - Discover hidden structure, patterns, or clusters in unlabeled data
- **Example:** identifying "types" of users in social media usage data



"Customer segmentation": similar applications in Amazon, Youtube, Spotify, TikTok, etc

Unsupervised Learning

- **Data:** features/covariates $\mathbf{X}$ ~~and response $\mathbf{y}$~~
- **Goal:**
  - Discover hidden structure, patterns, or clusters in unlabeled data
- **Example:** identifying "types" of users in social media usage data

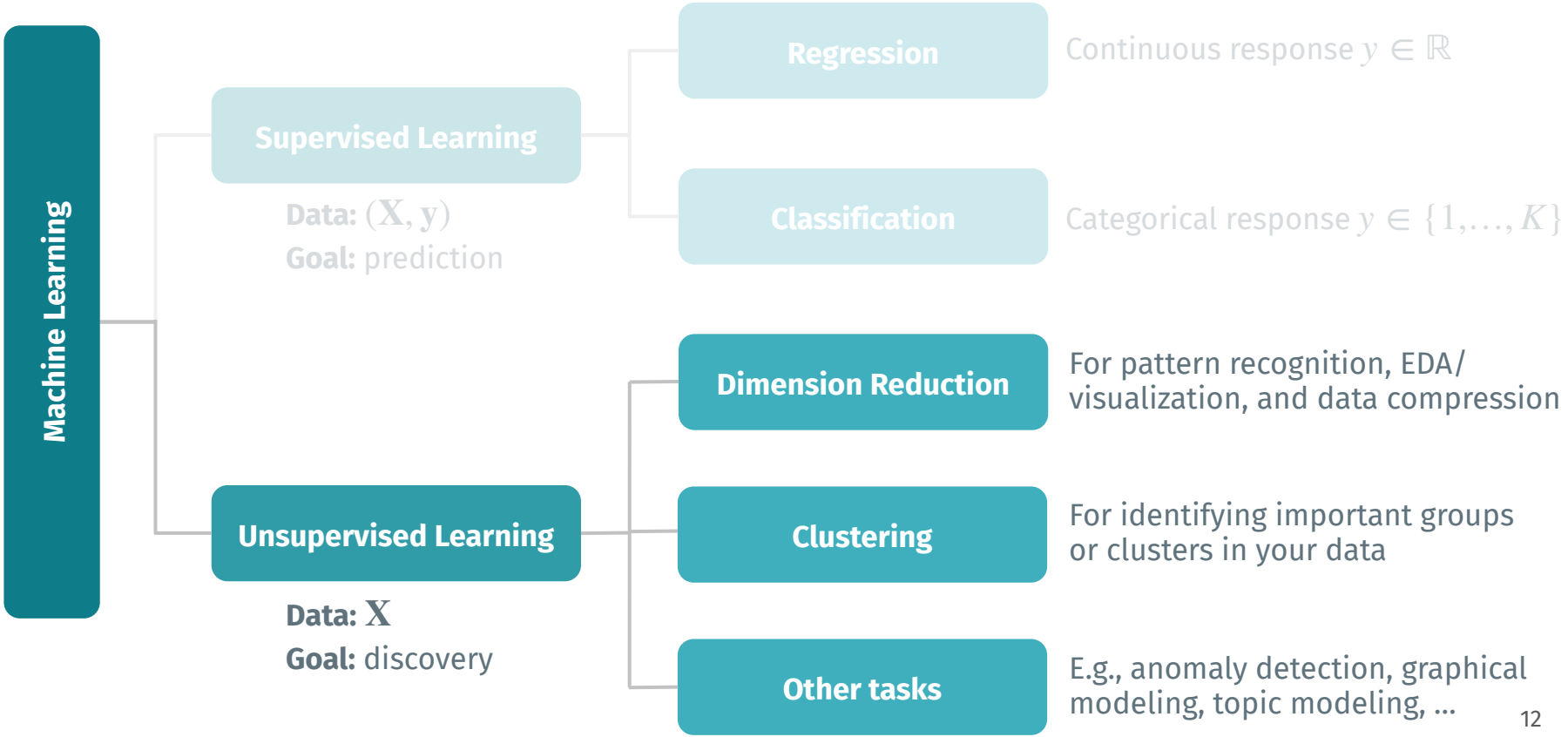


"Customer segmentation": similar applications in Amazon, Youtube, Spotify, TikTok, etc

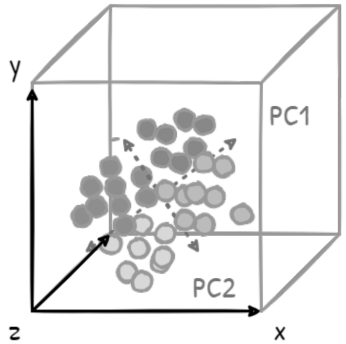
Other applications

- Political Science: voter archetypes
- Psychology: personality types
- Cell Biology: clustering cell types
- Medicine: patient subgroups

Overview of Machine Learning Terminology

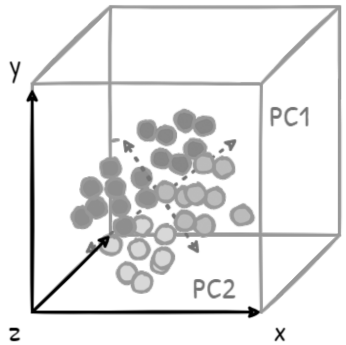


A Common Unsupervised Learning Pipeline



Original Data
(often “high-dimensional”)

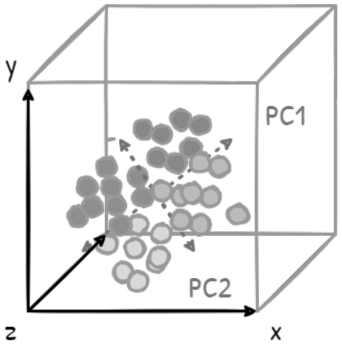
A Common Unsupervised Learning Pipeline



Original Data
(often “high-dimensional”)

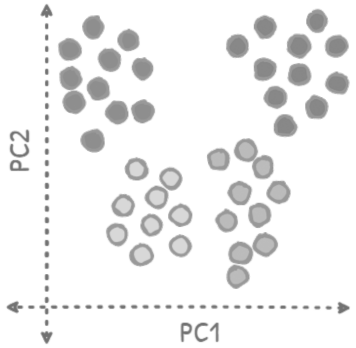
Step 1:
Dimension reduction

A Common Unsupervised Learning Pipeline



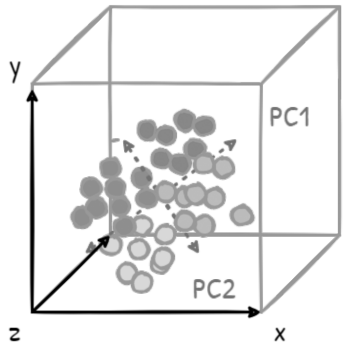
Original Data
(often “high-dimensional”)

Step 1:
Dimension
reduction



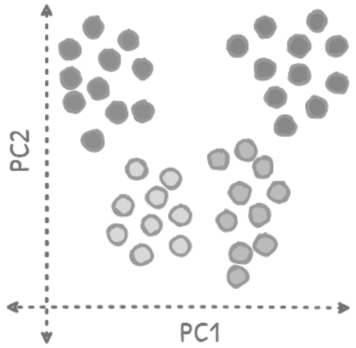
“Lower-dimensional”
representation

A Common Unsupervised Learning Pipeline



Original Data
(often “high-dimensional”)

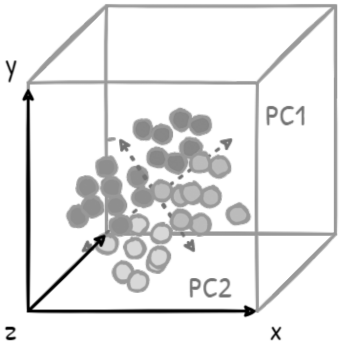
Step 1:
Dimension reduction



“Lower-dimensional” representation

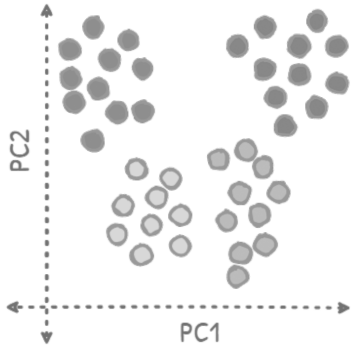
Step 2:
Clustering

A Common Unsupervised Learning Pipeline



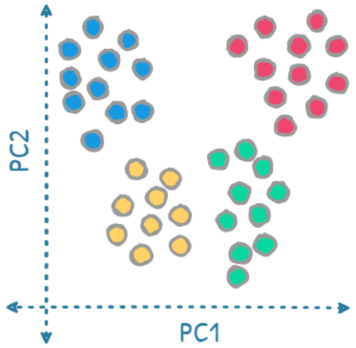
Original Data
(often “high-dimensional”)

Step 1:
Dimension reduction

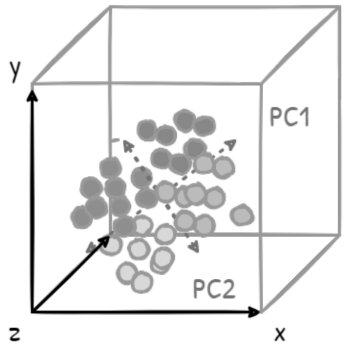


“Lower-dimensional” representation

Step 2:
Clustering

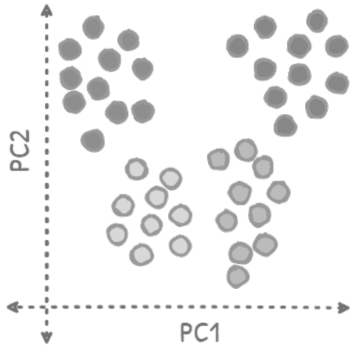


A Common Unsupervised Learning Pipeline



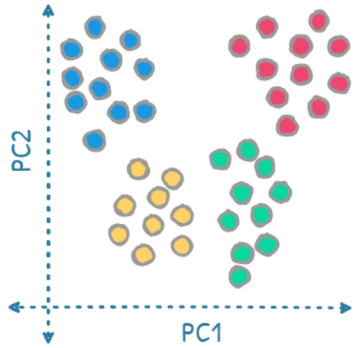
Original Data
(often “high-dimensional”)

Step 1:
Dimension
reduction



“Lower-dimensional”
representation

Step 2:
Clustering



* Focus of Lab 2

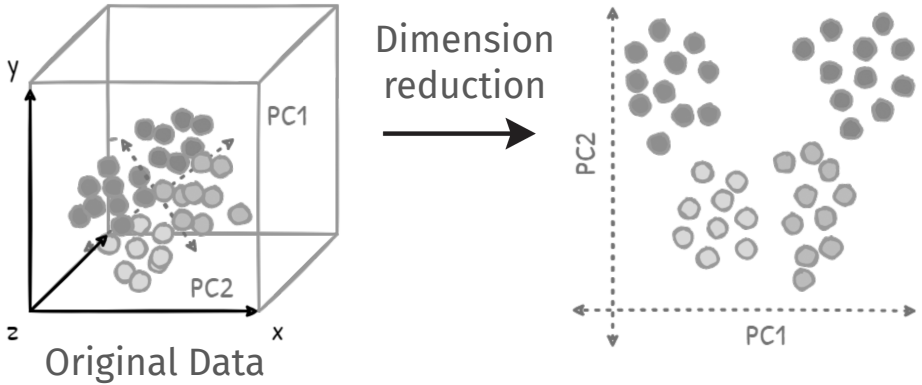
Dimension Reduction

Dimension Reduction

Dimension Reduction: aims to find a lower-dimensional representation of the data which preserves as much of the original information as possible

What can dimension reduction methods be used for?

- EDA/visualizations
- Data compression
- Feature engineering (used as input into either clustering methods or prediction models)



Common Dimension Reduction Methods

- **PCA:** Principal Component Analysis
- **t-SNE:** t-distributed Stochastic Neighbor Embedding
- **UMAP:** Uniform Manifold Approximation and Projection
- **Autoencoders**
- And more: see course website [Dimension Reduction Section]

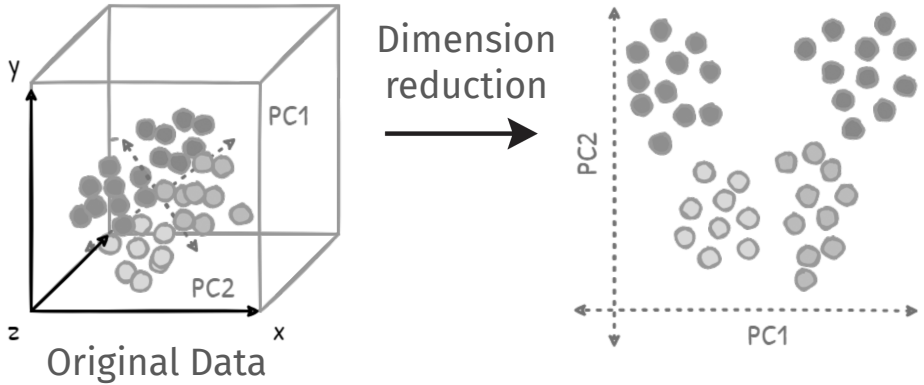
Common Dimension Reduction Methods

- **PCA:** Principal Component Analysis
- **t-SNE:** t-distributed Stochastic Neighbor Embedding
- **UMAP:** Uniform Manifold Approximation and Projection
- **Autoencoders**
- And more: see course website [Dimension Reduction Section]

Principal Components Analysis (PCA)

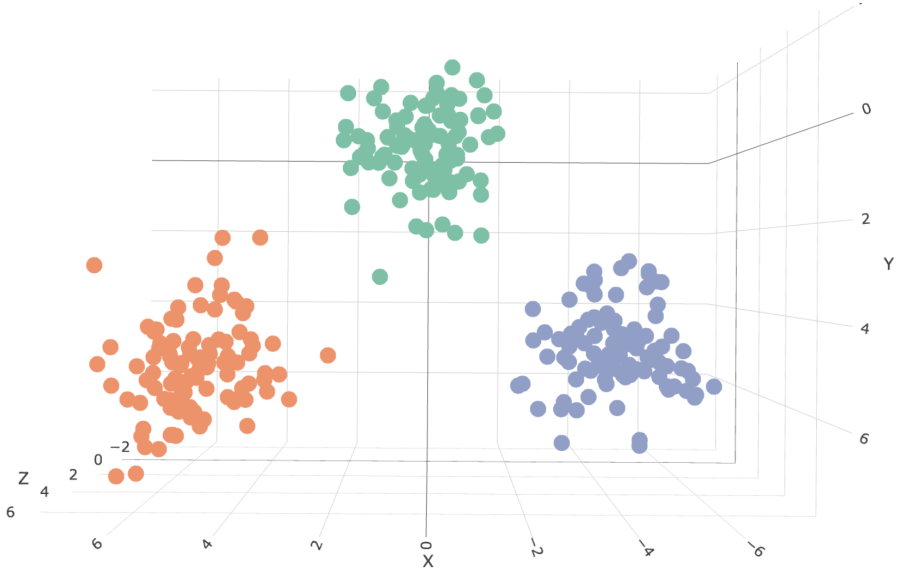
Principal Components Analysis (PCA): finds a lower-dimensional (linear) projection of the data which preserves as much of the **variance** in the data as possible

- PCA finds a lower-dimensional hyperplane (or orthogonal directions) such that when the data is projected onto the hyperplane, the variance of the data is maximized



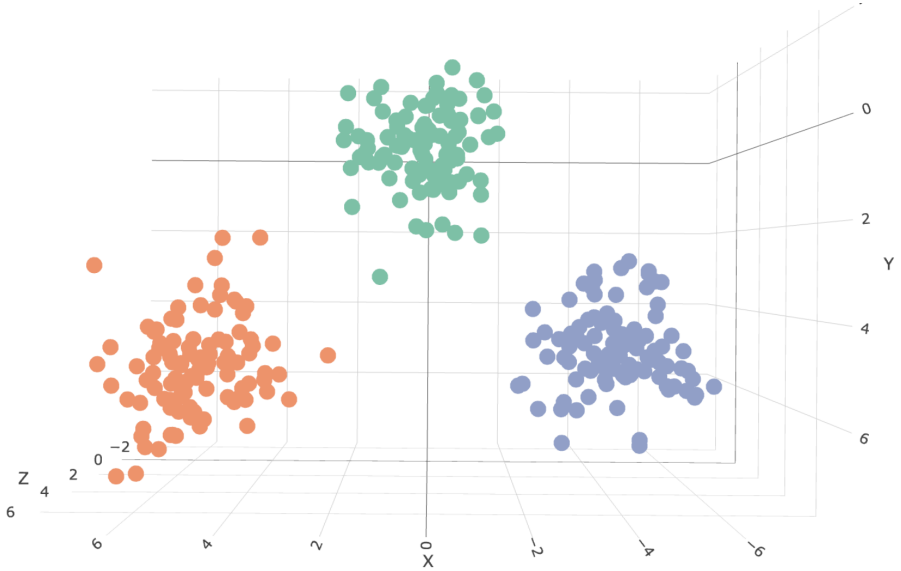
When does PCA "work" and when does PCA "not work"?

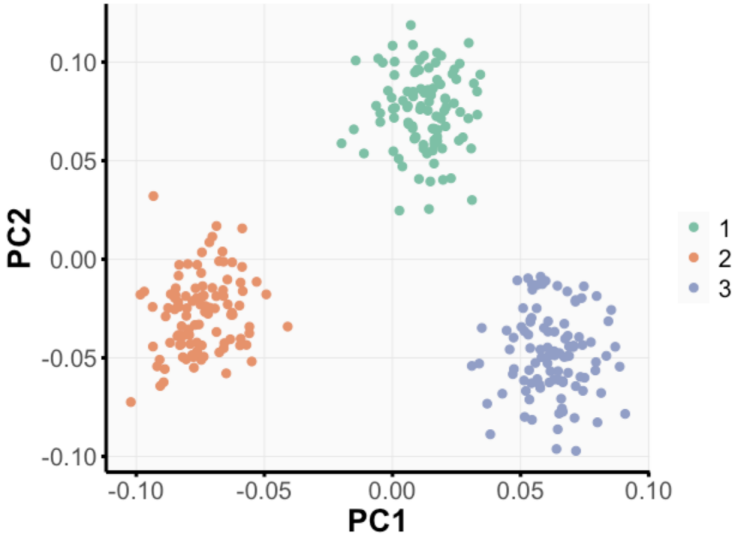
Scenario: Gaussian data



When does PCA "work" and when does PCA "not work"?

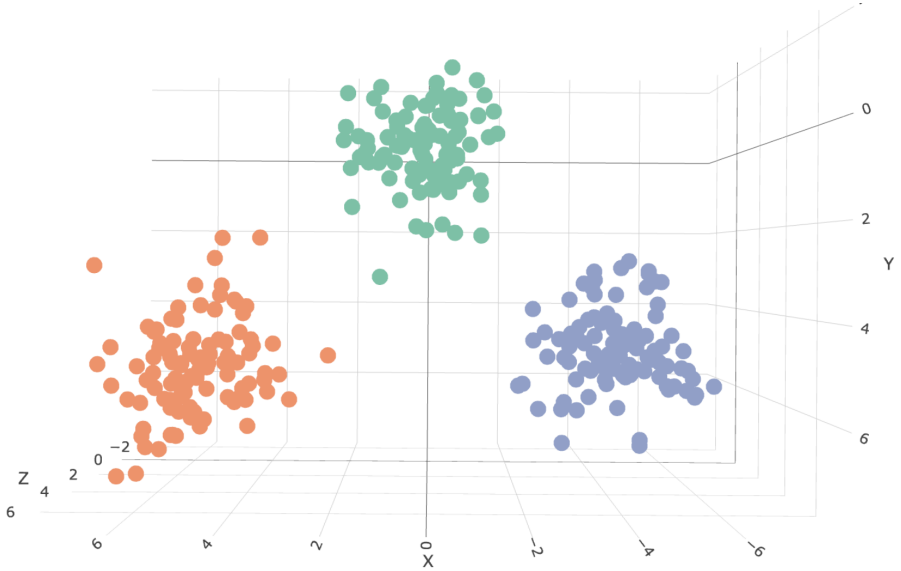
Scenario: Gaussian data

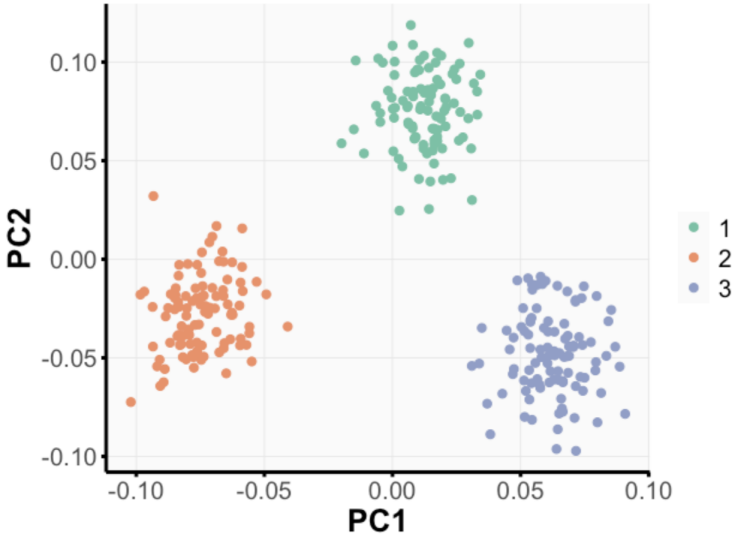




When does PCA "work" and when does PCA "not work"?

Scenario: Gaussian data



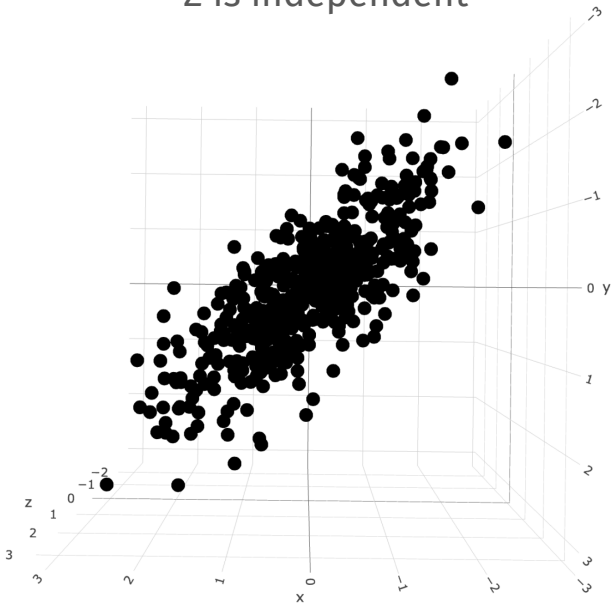


✔ This is the ideal scenario for PCA

When does PCA "work" and when does PCA "not work"?

Scenario: Correlated Variables

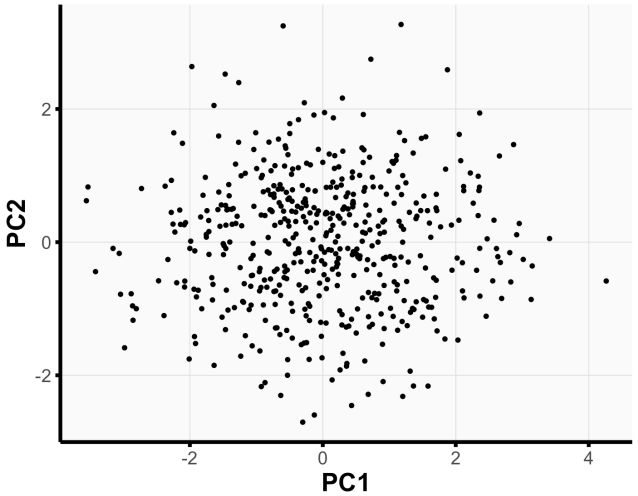
X and Y are highly correlated;
Z is independent



PC Loadings:

PC1 = 0.7X + 0.7Y + 0.1Z

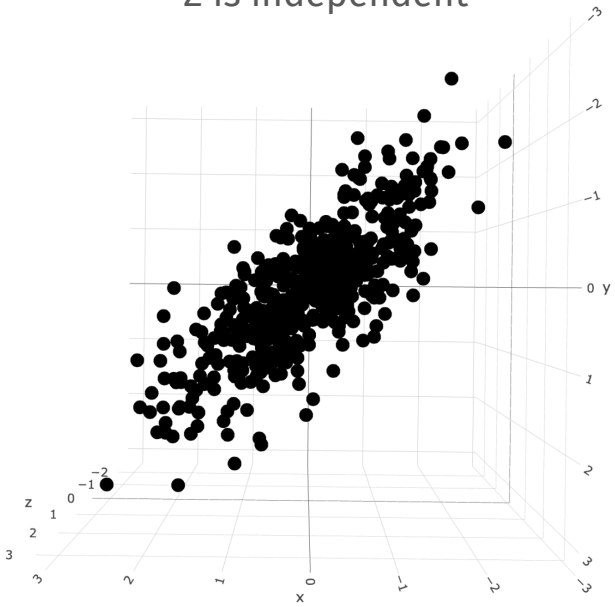
PC2 = -0.1X - 0.1Y + 1.0Z



When does PCA "work" and when does PCA "not work"?

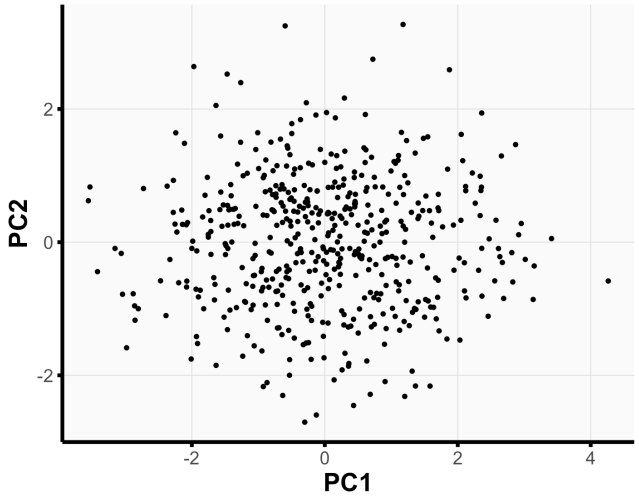
Scenario: Correlated Variables

X and Y are highly correlated;
Z is independent



PC Loadings:

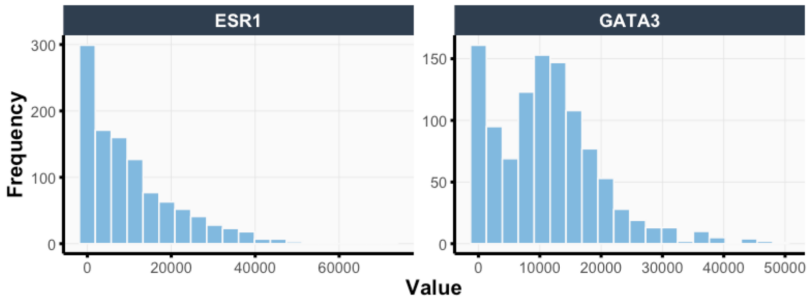
```
PC1 =  0.7X + 0.7Y + 0.1Z
PC2 = -0.1X - 0.1Y + 1.0Z
```



PCs typically group correlated variables together

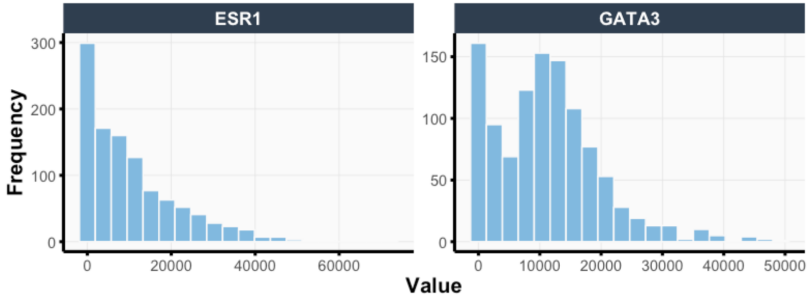
When does PCA "work" and when does PCA "not work"?

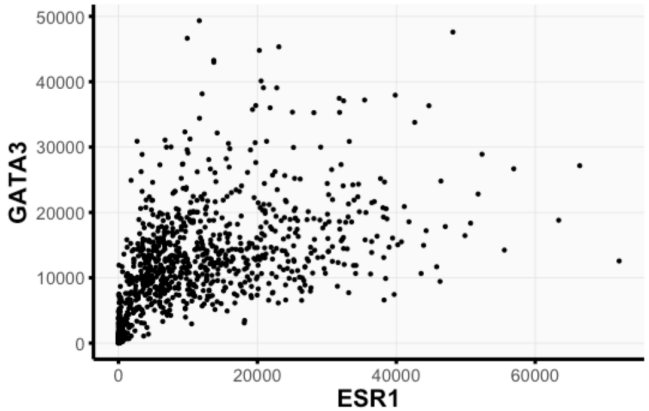
Scenario: Highly skewed data or data with outliers



When does PCA "work" and when does PCA "not work"?

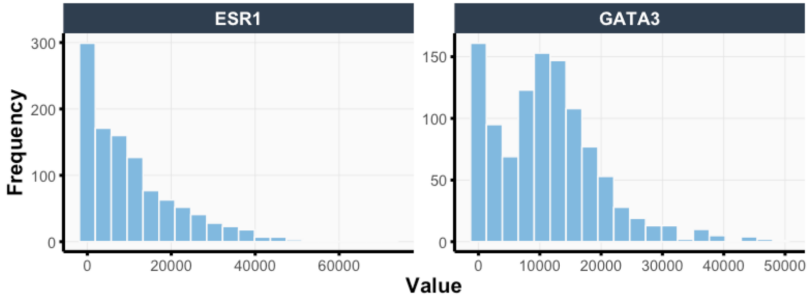
Scenario: Highly skewed data or data with outliers

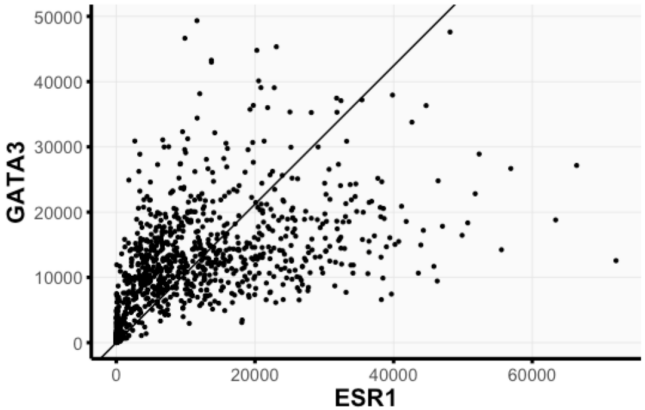




When does PCA "work" and when does PCA "not work"?

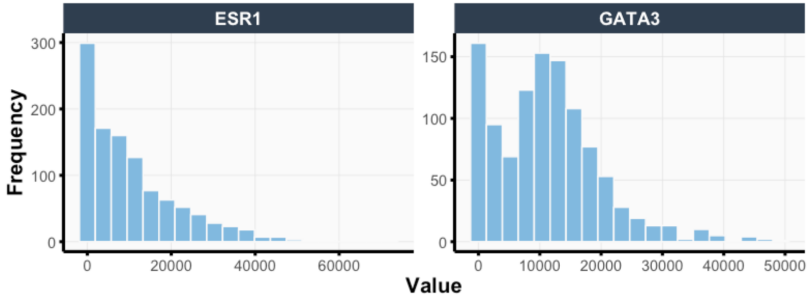
Scenario: Highly skewed data or data with outliers

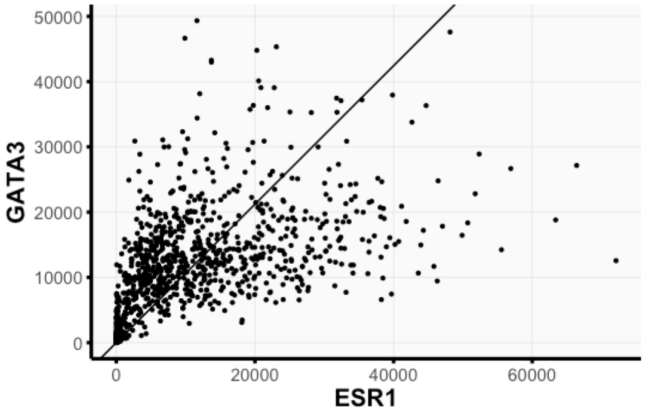


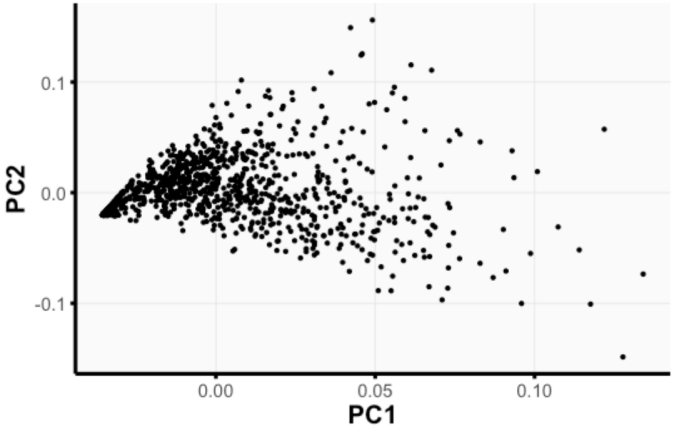


When does PCA "work" and when does PCA "not work"?

Scenario: Highly skewed data or data with outliers

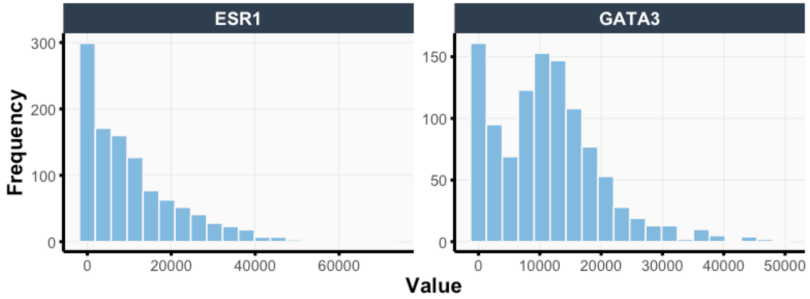


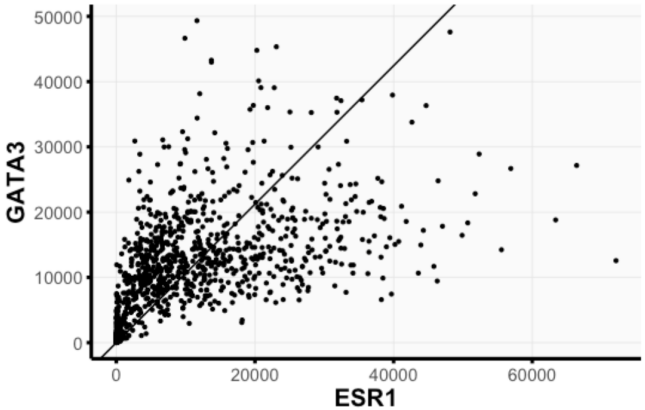


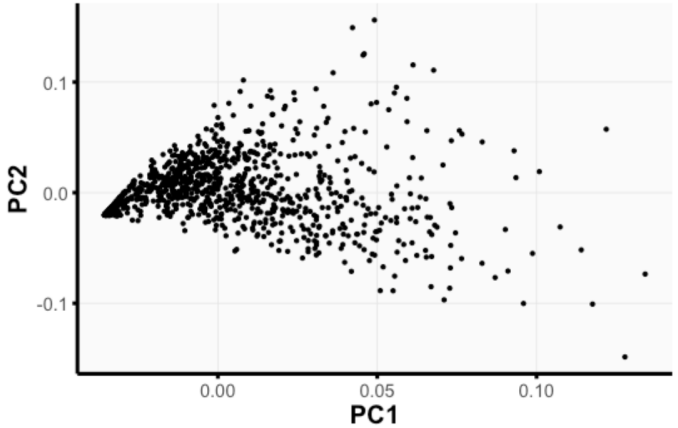


When does PCA "work" and when does PCA "not work"?

Scenario: Highly skewed data or data with outliers



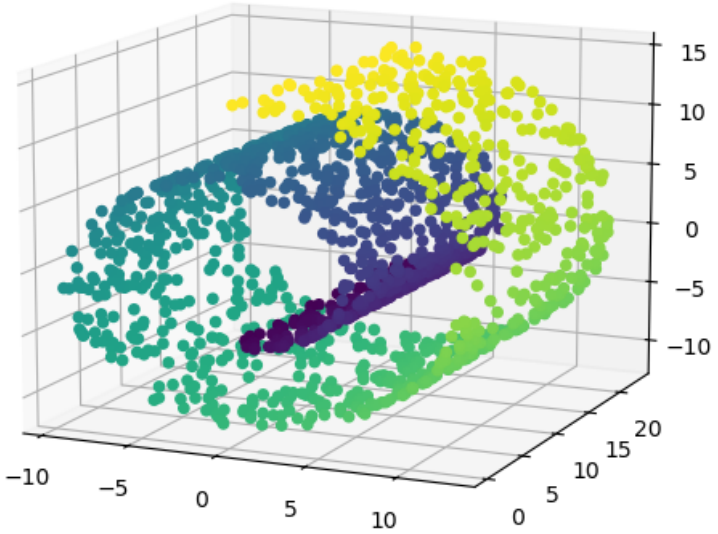




✔ **if** variance is still a meaningful measure of information

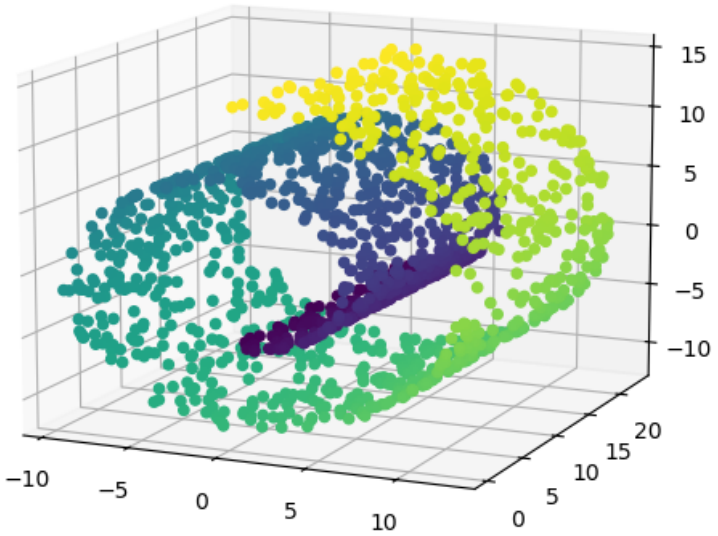
When does PCA "work" and when does PCA "not work"?

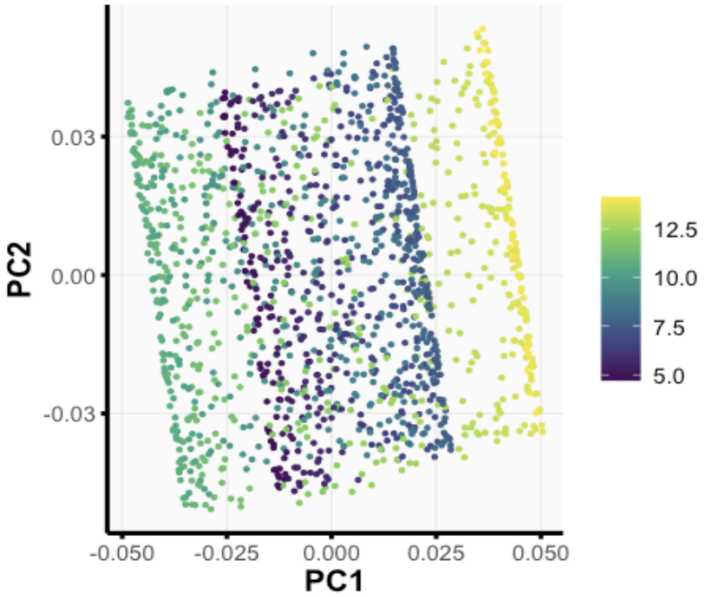
Scenario: Swiss roll



When does PCA "work" and when does PCA "not work"?

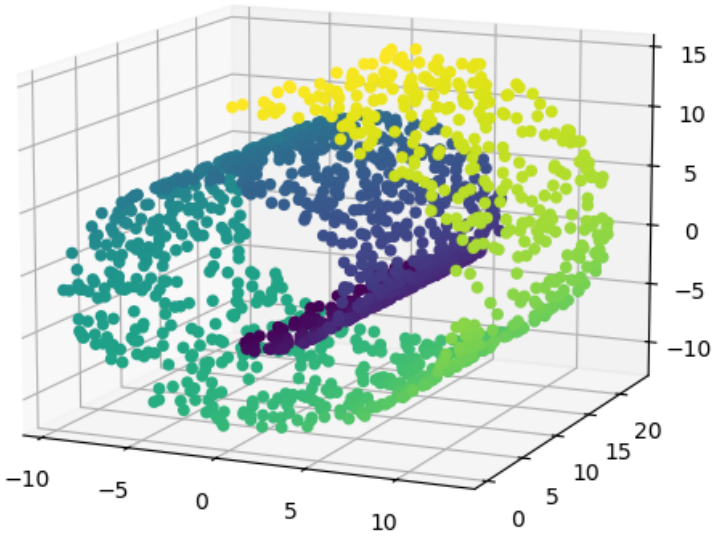
Scenario: Swiss roll

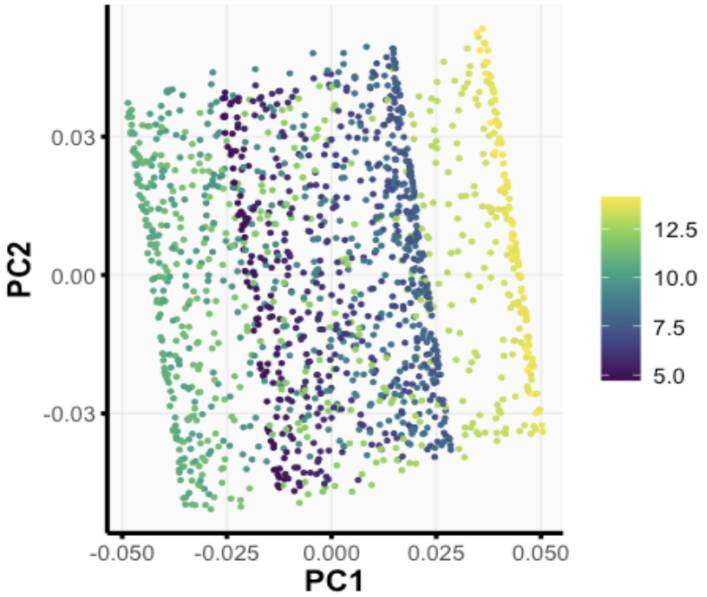




When does PCA "work" and when does PCA "not work"?

Scenario: Swiss roll





✘ not great for nonlinear manifolds

Common Dimension Reduction Methods

- **PCA:** Principal Component Analysis *[linear method]*
- **t-SNE:** t-distributed Stochastic Neighbor Embedding
- **UMAP:** Uniform Manifold Approximation and Projection
- **Autoencoders**

} *non-linear methods*

- And more: see course website *(Dimension Reduction Section)*

Common Dimension Reduction Methods

- **PCA:** Principal Component Analysis *[linear method]*
- **t-SNE:** t-distributed Stochastic Neighbor Embedding
- **UMAP:** Uniform Manifold Approximation and Projection

} *non-linear methods*

- **Autoencoders**
- And more: see course website *(Dimension Reduction Section)*

t-SNE and UMAP

- t-SNE and UMAP are examples of a broader class of methods called **neighborhood embedding methods**
- High-level idea: find a low-dimensional embedding of the data such that

Distance between points in **original high-dimensional** space $\approx$ Distance between points in **new low-dimensional** space

t-SNE: t-distributed Stochastic Neighbor Embedding

1. Compute Euclidean distance between every pair of points in X
2. Translate these pairwise distances into probability of being neighbors
   + Large pairwise distance → low probability of being neighbors
3. Find lower-dimensional representation such that

Prob(i and j are neighbors) in **original high-dimensional** space $\approx$ Prob(i and j are neighbors) in **new low-dimensional** space

t-SNE: t-distributed Stochastic Neighbor Embedding

1. Compute Euclidean distance between every pair of points in X
2. Translate these pairwise distances into probability of being neighbors
   + Large pairwise distance → low probability of being neighbors
3. Find lower-dimensional representation such that

Prob(i and j are neighbors) in **original high-dimensional** space $\approx$ Prob(i and j are neighbors) in **new low-dimensional** space

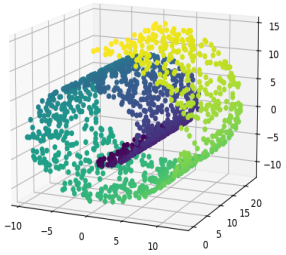
Hyperparameter: perplexity

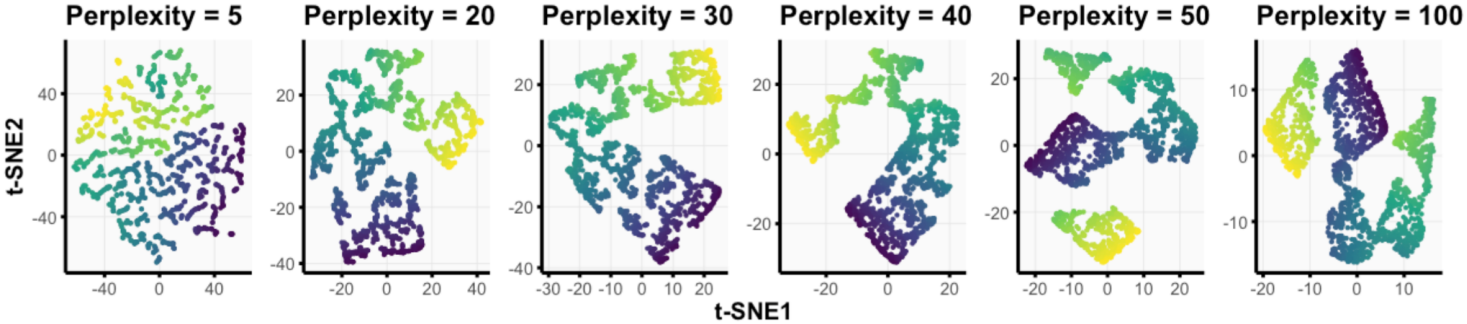
t-SNE: t-distributed Stochastic Neighbor Embedding

1. Compute Euclidean distance between every pair of points in X
2. Translate these pairwise distances into probability of being neighbors
   + Large pairwise distance → low probability of being neighbors
3. Find lower-dimensional representation such that

Prob(i and j are neighbors) in **original high-dimensional** space ≈ Prob(i and j are neighbors) in **new low-dimensional** space

Hyperparameter: perplexity





UMAP: Uniform Manifold Approximation and Projection

- Like tSNE, the idea is that pairs of points that are close in the original high-dimensional space should also be close in the new low-dimensional space
- How does UMAP differ from tSNE? Different similarity metrics, loss function, optimization algorithm

UMAP: Uniform Manifold Approximation and Projection

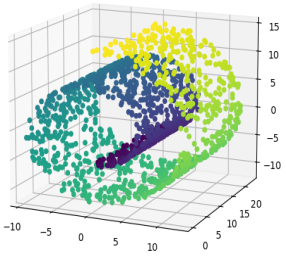
- Like tSNE, the idea is that pairs of points that are close in the original high-dimensional space should also be close in the new low-dimensional space
- How does UMAP differ from tSNE? Different similarity metrics, loss function, optimization algorithm

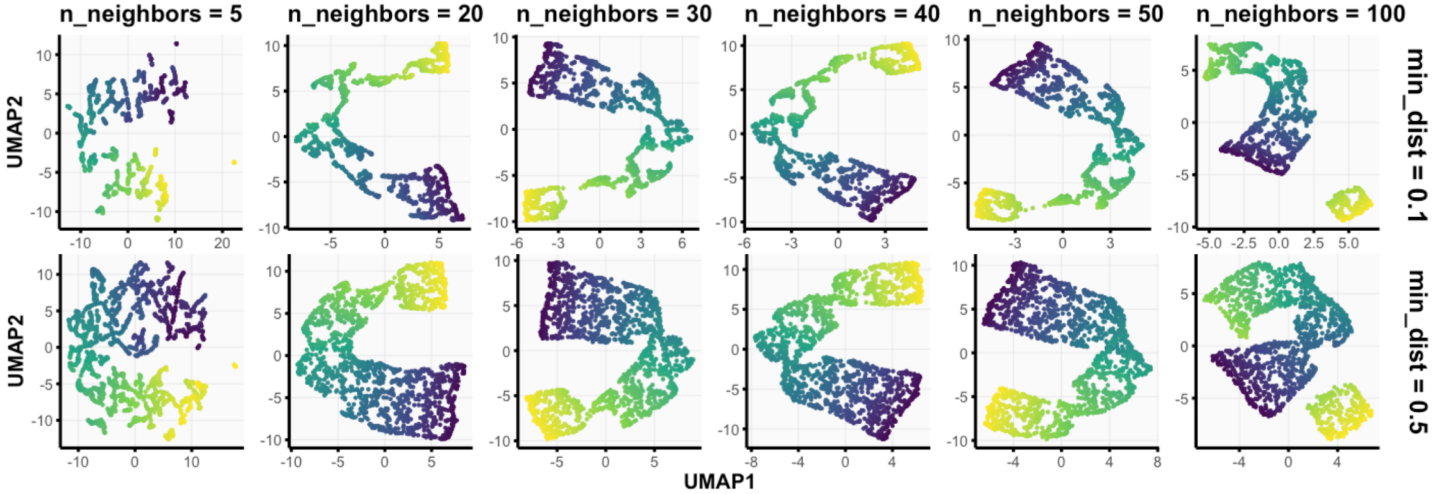
Hyperparameter: n_neighbors and min_distance

UMAP: Uniform Manifold Approximation and Projection

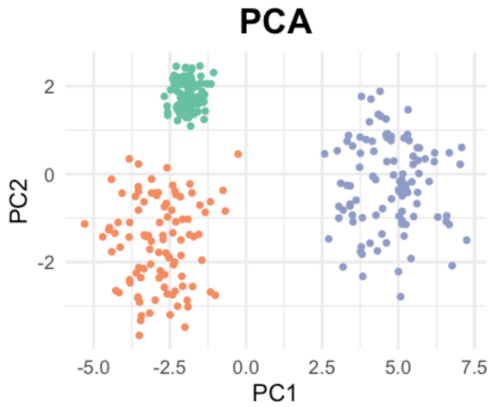
- Like tSNE, the idea is that pairs of points that are close in the original high-dimensional space should also be close in the new low-dimensional space
- How does UMAP differ from tSNE? Different similarity metrics, loss function, optimization algorithm

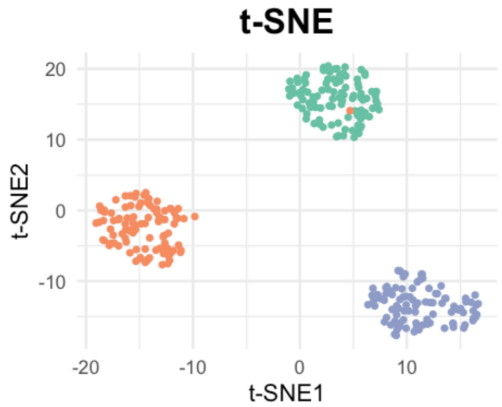
Hyperparameter: n_neighbors and min_distance

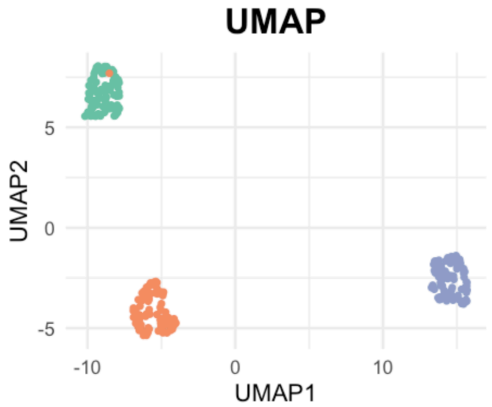




Word of caution: t-SNE and UMAP can distort global structure

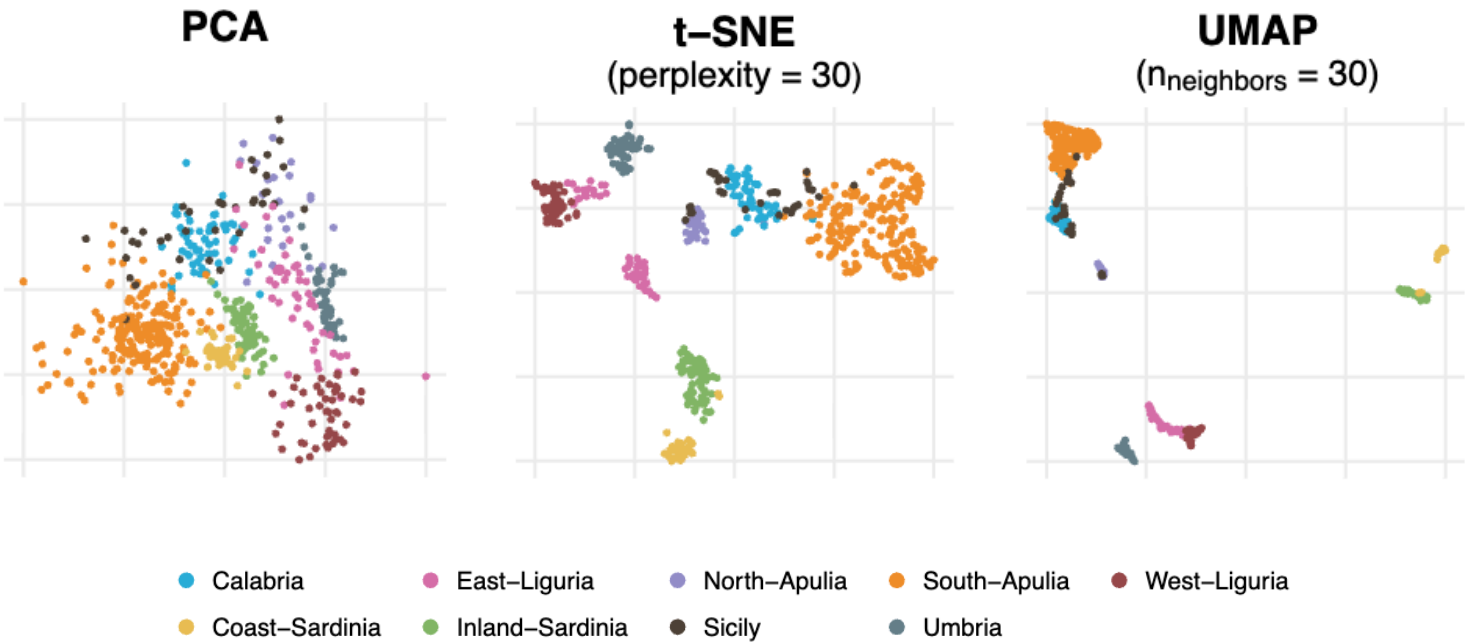






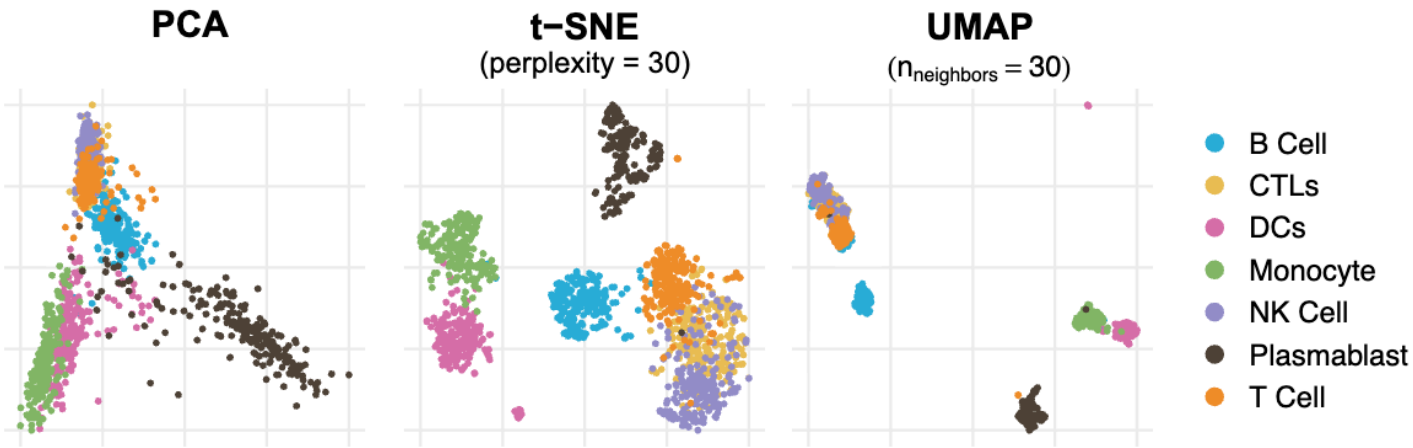
Word of caution: t-SNE and UMAP can exaggerate clusters

Example: Olive oil data



Word of caution: t-SNE and UMAP can exaggerate clusters

Example: Single-cell RNA-Sequencing data (from HIV-infected individuals)



Common Dimension Reduction Methods

- **PCA:** Principal Component Analysis *[linear method]*
- **t-SNE:** t-distributed Stochastic Neighbor Embedding
- **UMAP:** Uniform Manifold Approximation and Projection

} *non-linear methods*

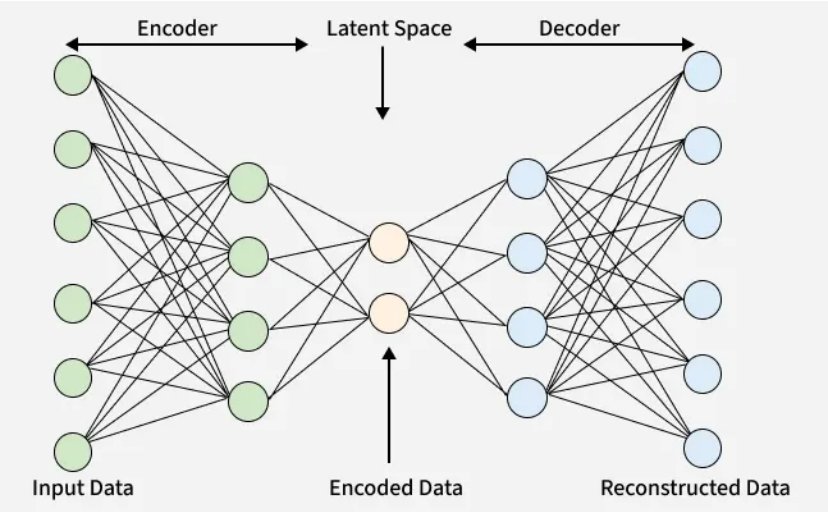
- **Autoencoders**
- And more: see course website *(Dimension Reduction Section)*

Autoencoders

- **What:** neural network architecture consisting of encoder + decoder
- **Idea:** learn low-dimensional representation that can be used to accurately reconstruct the original data
- **How:** minimize reconstruction error:

$$\|x - g(f(x))\|_2^2$$

original data — reconstructed data

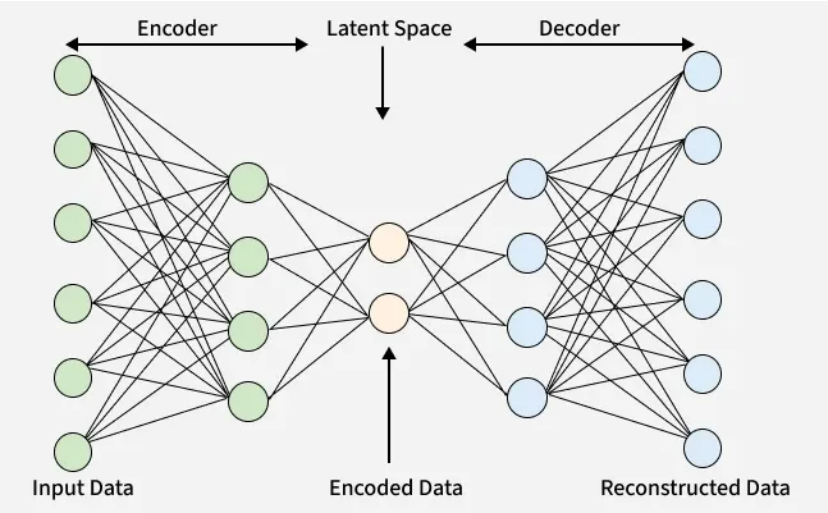


Autoencoders

- **What:** neural network architecture consisting of encoder + decoder
- **Idea:** learn low-dimensional representation that can be used to accurately reconstruct the original data
- **How:** minimize reconstruction error:

$$\|x - g(f(x))\|_2^2$$

original data — reconstructed data



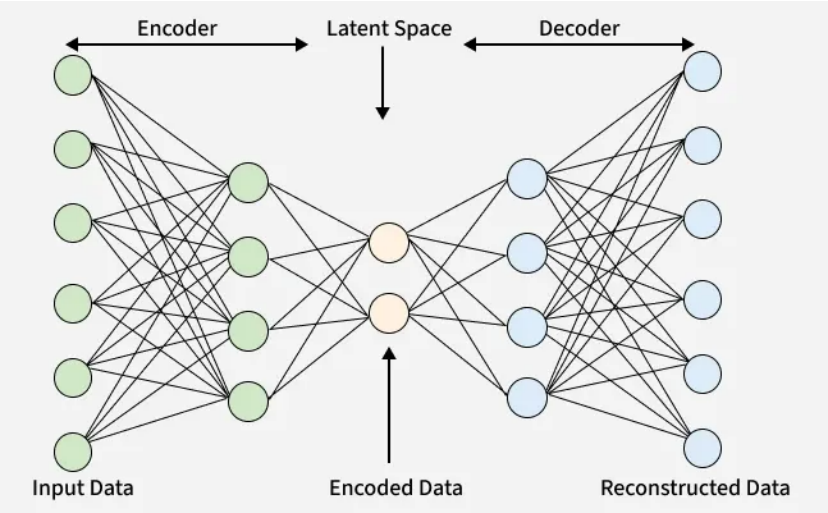
$x \in \mathbb{R}^6$

Autoencoders

- **What:** neural network architecture consisting of encoder + decoder
- **Idea:** learn low-dimensional representation that can be used to accurately reconstruct the original data
- **How:** minimize reconstruction error:

$$\|x - g(f(x))\|_2^2$$

original data — reconstructed data



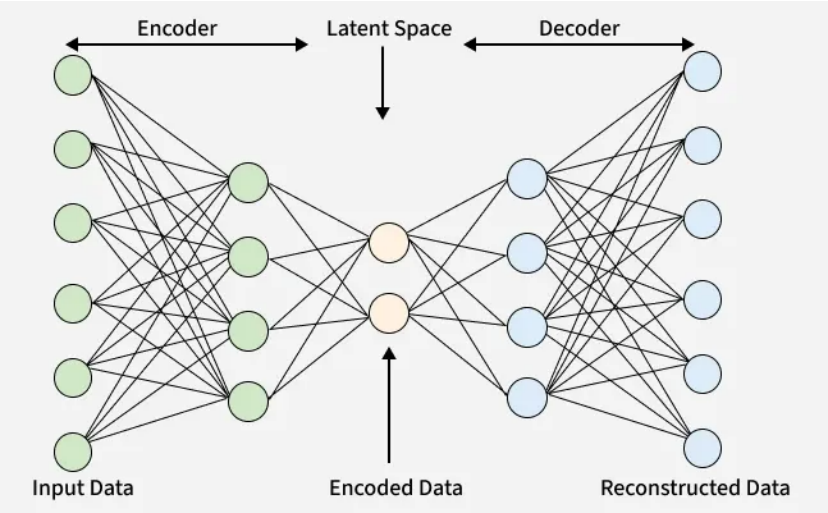
$$x \in \mathbb{R}^6 \xrightarrow{f}$$

Autoencoders

- **What:** neural network architecture consisting of encoder + decoder
- **Idea:** learn low-dimensional representation that can be used to accurately reconstruct the original data
- **How:** minimize reconstruction error:

$$\|x - g(f(x))\|_2^2$$

original data — reconstructed data



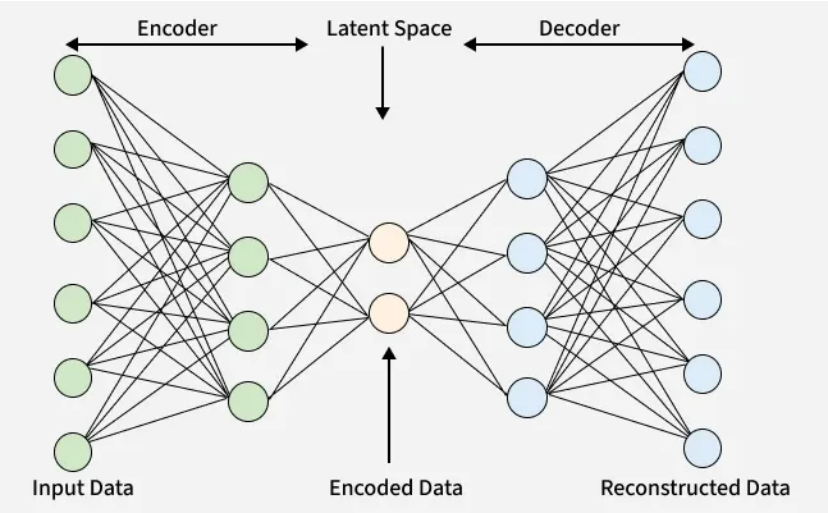
$$x \in \mathbb{R}^6 \xrightarrow{f} z = f(x)$$

Autoencoders

- **What:** neural network architecture consisting of encoder + decoder
- **Idea:** learn low-dimensional representation that can be used to accurately reconstruct the original data
- **How:** minimize reconstruction error:

$$\|x - g(f(x))\|_2^2$$

original data — reconstructed data



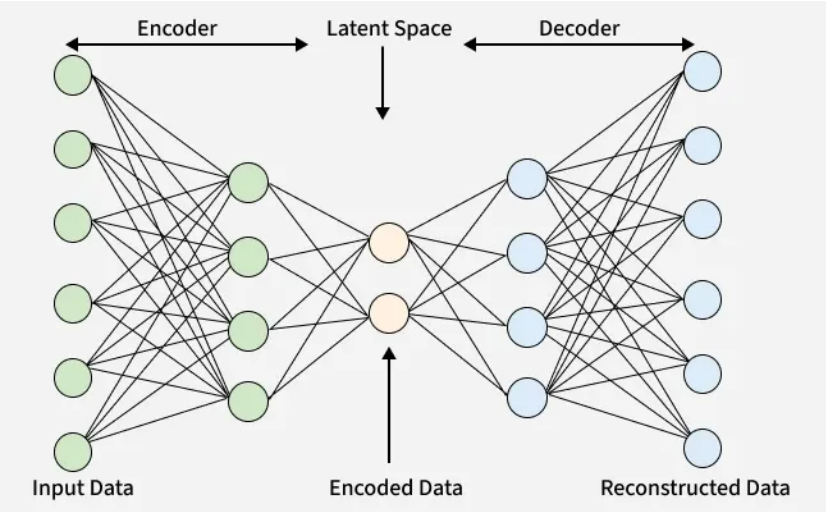
$$x \in \mathbb{R}^6 \xrightarrow{f} z = f(x) \in \mathbb{R}^2$$

Autoencoders

- **What:** neural network architecture consisting of encoder + decoder
- **Idea:** learn low-dimensional representation that can be used to accurately reconstruct the original data
- **How:** minimize reconstruction error:

$$\|x - g(f(x))\|_2^2$$

original data — reconstructed data



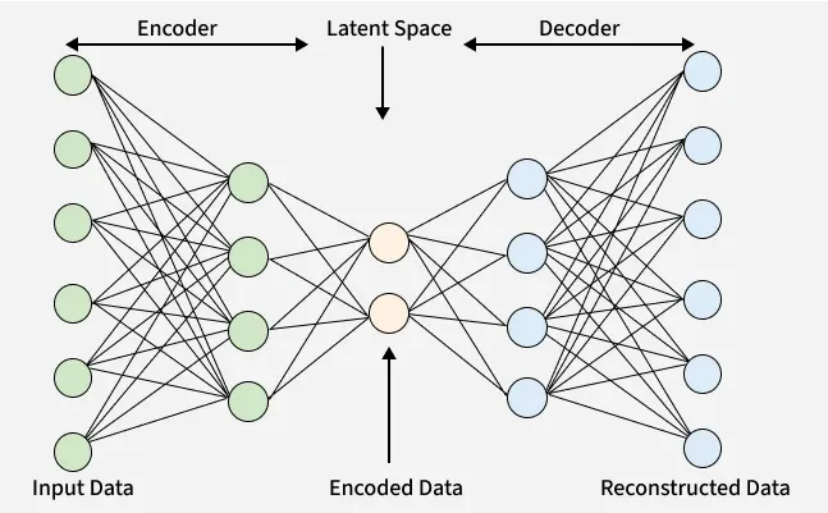
$$x \in \mathbb{R}^6 \xrightarrow{f} z = f(x) \in \mathbb{R}^2 \xrightarrow{g}$$

Autoencoders

- **What:** neural network architecture consisting of encoder + decoder
- **Idea:** learn low-dimensional representation that can be used to accurately reconstruct the original data
- **How:** minimize reconstruction error:

$$\|x - g(f(x))\|_2^2$$

original data — reconstructed data



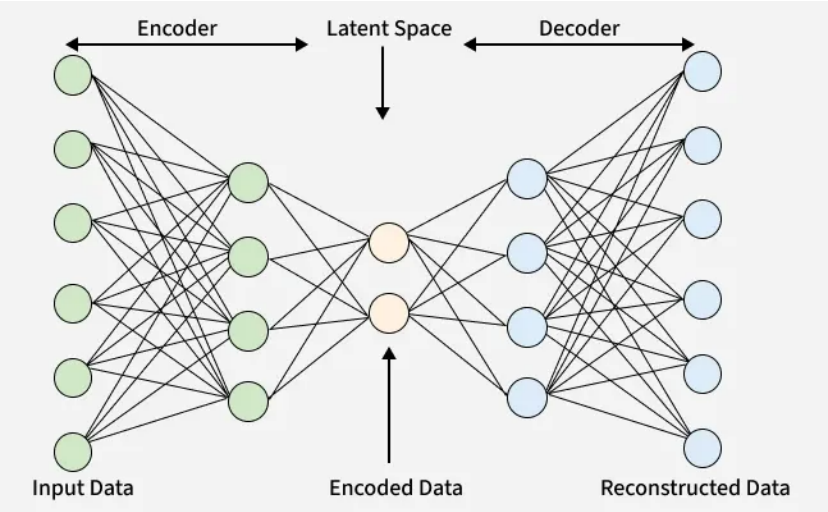
$$x \in \mathbb{R}^6 \xrightarrow{f} z = f(x) \in \mathbb{R}^2 \xrightarrow{g} g(f(x))$$

Autoencoders

- **What:** neural network architecture consisting of encoder + decoder
- **Idea:** learn low-dimensional representation that can be used to accurately reconstruct the original data
- **How:** minimize reconstruction error:

$$\|x - g(f(x))\|_2^2$$

original data — reconstructed data



$$x \in \mathbb{R}^6 \xrightarrow{f} z = f(x) \in \mathbb{R}^2 \xrightarrow{g} g(f(x)) \in \mathbb{R}^6$$

Recap: Dimension Reduction Methods

	PCA	t-SNE	UMAP	Autoencoders
Feature Interpretability	Yes	No	No	No
Linear/nonlinear	Linear	Nonlinear	Nonlinear	Nonlinear
Number of components	Orthogonal, nested; Can compute all *p* components at once	Non-nested; need to re-run for each choice of rank (typically, rank = 2 or 3)	Non-nested; need to re-run for each choice of rank (typically, rank = 2 or 3)	Non-nested; need to re-run for each choice of rank
Computation	Fast	Slower	Slower; faster than t-SNE	Expensive (best on GPU)
Unique, global solution	Yes	Local solution (results can change with different seed)	Local solution (results can change with different seed)	Local solution (results can change with different seed)
Other considerations?	No hyperparameters	Results can change drastically depending on hyperparameters; Not good at preserving global structure; Typically do PCA before inputting into tSNE	Results can change drastically depending on hyperparameters; Better at preserving global structure than tSNE; Typically do PCA before inputting into tSNE	Results can change drastically depending on architecture