Machine Learning at TACC Unsupervised Learning

July 17, 2024

PRESENTED BY:

Vlad Krendelev

Data management and Collections Group

Sikan Li

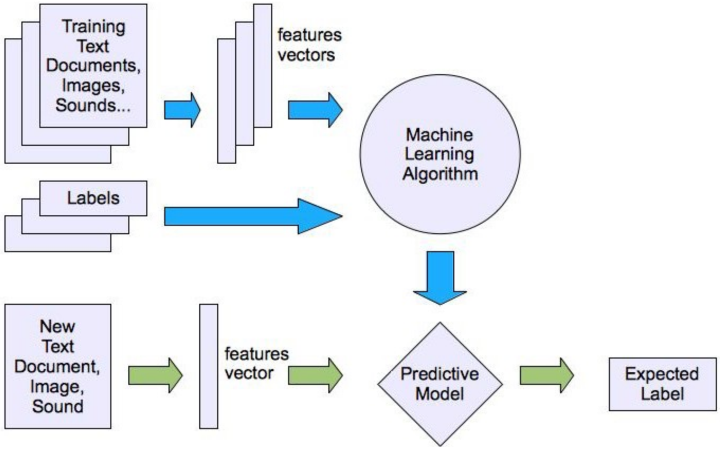
Scalable Computational Intelligence

Unsupervised Learning

Overview

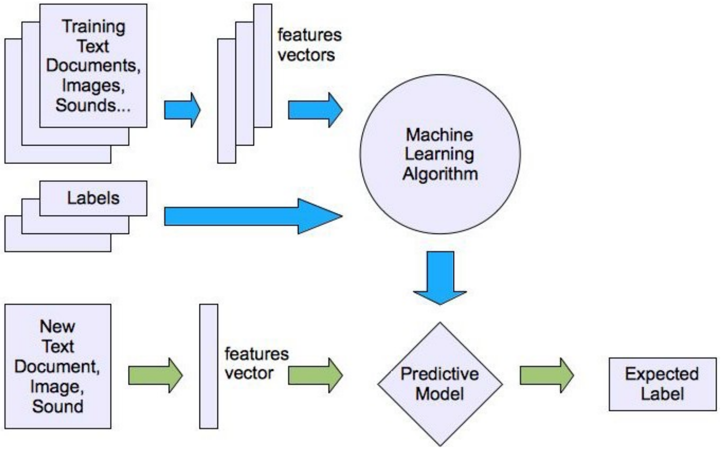
Machine Learning Taxonomy

Supervised learning
learn data model

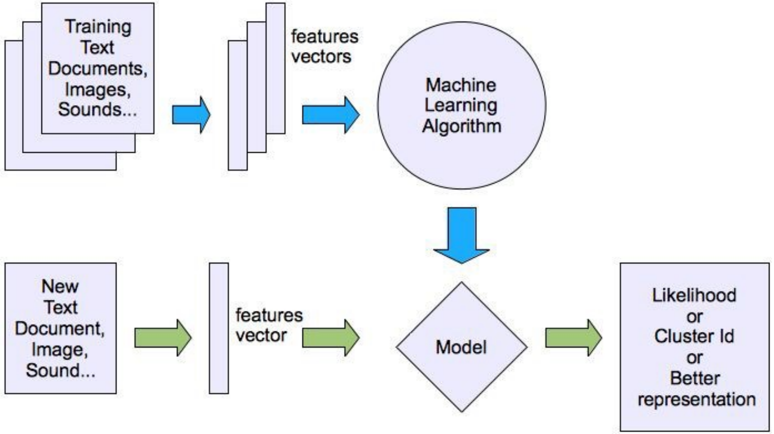


Machine Learning Taxonomy

Supervised learning
learn data model



Unsupervised learning
learn data structure



Unsupervised Learning - Overview

Unsupervised learning tools allow you to **explore the structure of your data**.

- How are the data best **represented**?
- Are there **clusters** within the data?
- Can we estimate the **density** of the data?

Unsupervised Learning Applications

- Recommendation systems
- Image segmentation and compression
- Life sciences
- …

Data Representation

Overview

Learning Data Representation

By changing how your data are represented, you can

- Identify patterns in your data.
- Learn which variables drive those patterns.

When should you think about data representation?

- When you have high-dimension data (>3 variables).
- As a pre-processing step before clustering or other machine learning workflows.

Data Representation

Dimensionality Reduction

Manifold Hypothesis

Many high-dimensional data sets describing processes from the real world have low-dimensional representations (manifolds).

Manifold Hypothesis

Many high-dimensional data sets describing processes from the real world have low-dimensional representations (manifolds).

Dimensionality reduction algorithms compress information contained in a dataset by transforming it onto a low-dimensional subspace while maintaining most of the relevant information.

Principal Component Analysis (PCA)

PCA helps to find a new low-dimensional coordinate system (basis) and projects the data onto it (i.e., changes its representation).

PCA's implementation is based on either eigendecomposition (eigenvectors and eigenvalues) or Singular Value Decomposition (SVD).

Principal Component Analysis (PCA)

The resulting basis is **orthogonal** and its axes can be interpreted as the directions of **maximum variance** such that the direction of the *first coordinate* corresponds to the *greatest variance*, the second coordinate – *second-greatest variance*, …, and so on for *n* dimensions.

Principal Component Analysis (PCA)

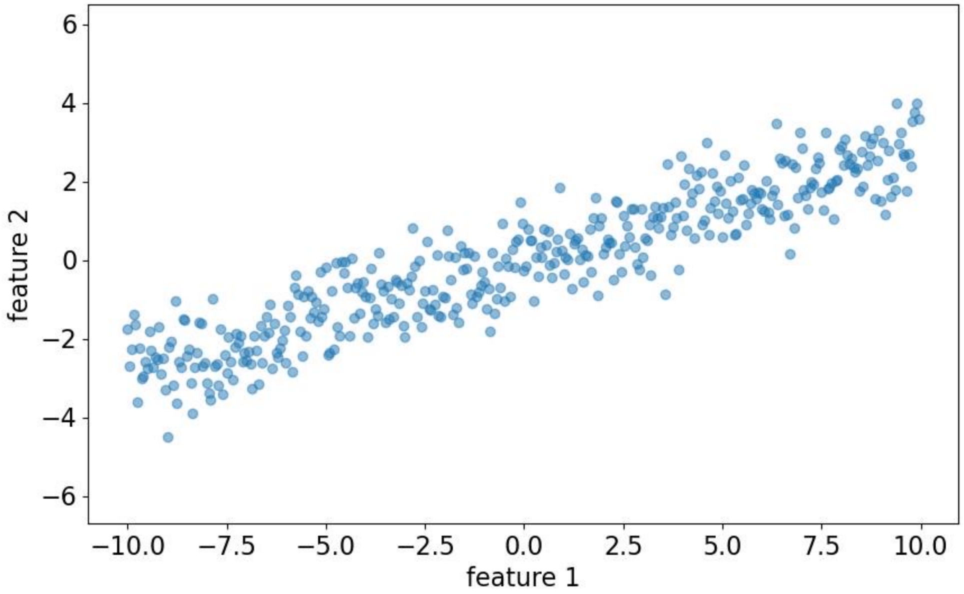
orthogonal ≈ no correlation (between projected features)

larger variance ≈ more information

!!! The axes with *smaller variance* can be neglected, hence low-dimensional (reduced) basis !!!

Principal Component Analysis (PCA)

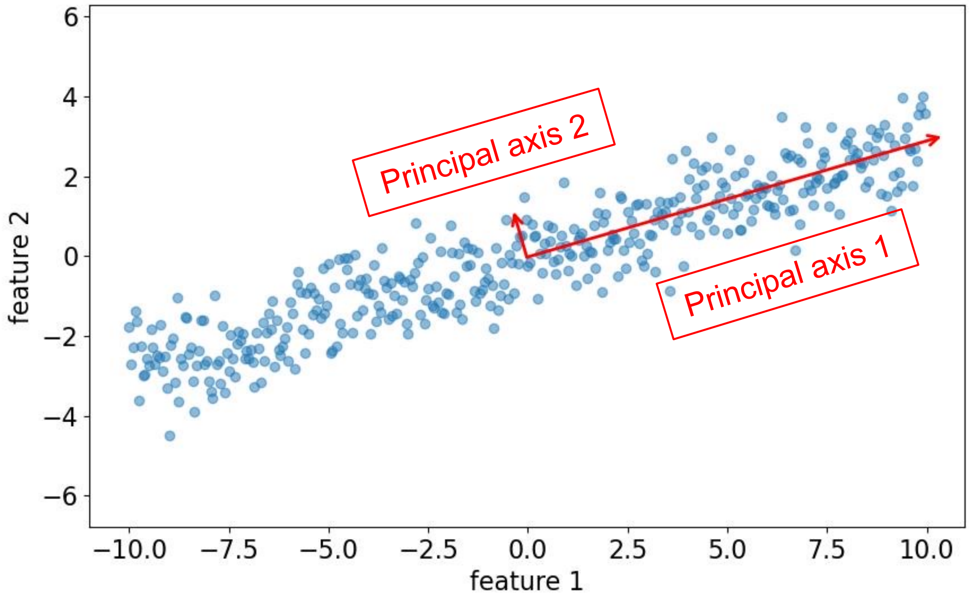
"Toy" Example #1



Question: Can we reduce the dimensions from 2 to 1?

Principal Component Analysis (PCA)

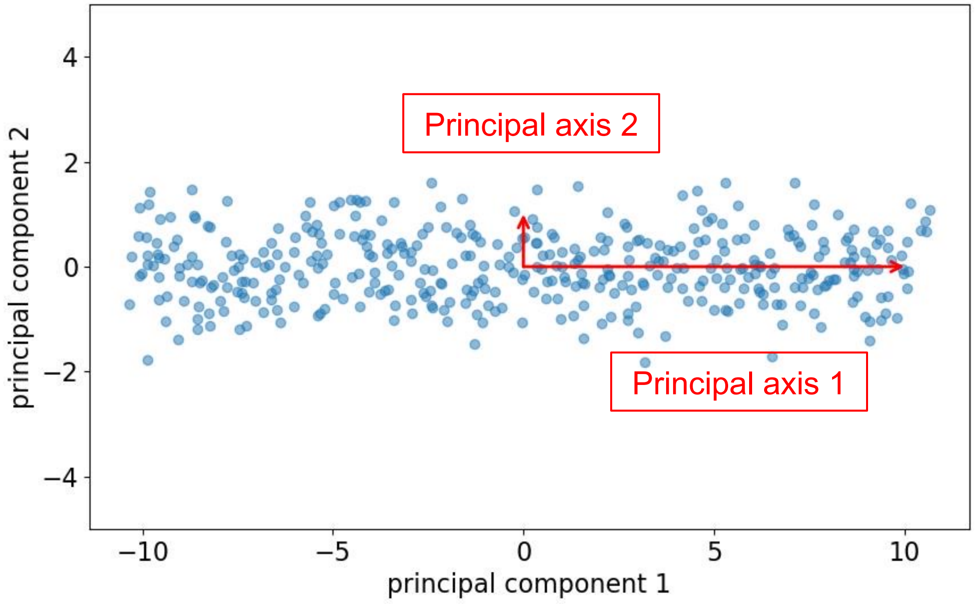
"Toy" Example #1



Question: Can we reduce the dimensions from 2 to 1?

Principal Component Analysis (PCA)

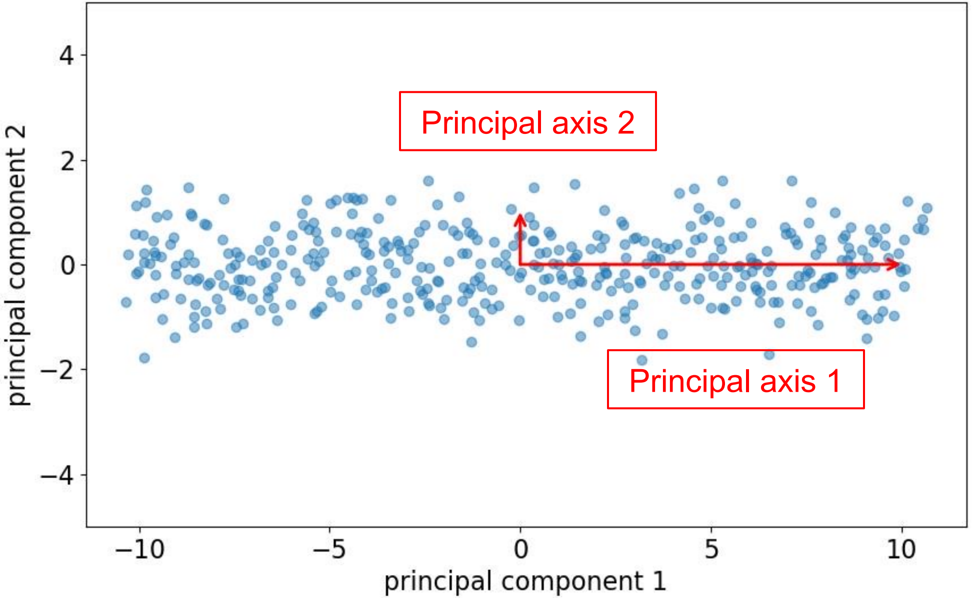
"Toy" Example #1



Question: Can we reduce the dimensions from 2 to 1?

Principal Component Analysis (PCA)

"Toy" Example #1

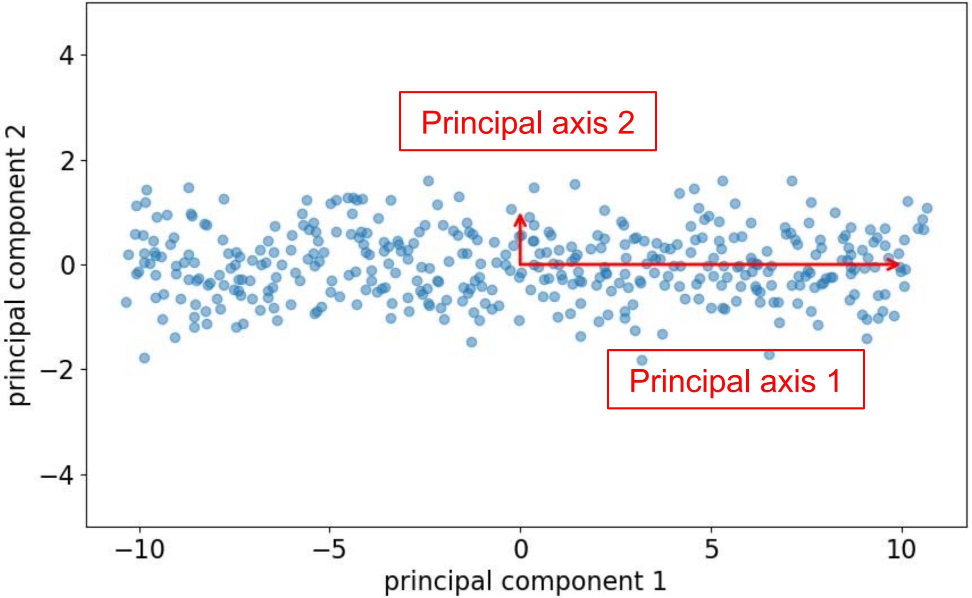



```
pca.explained_variance_ratio_
array([0.98806067, 0.01193933])
```

Question: Can we reduce the dimensions from 2 to 1?

Principal Component Analysis (PCA)

"Toy" Example #1



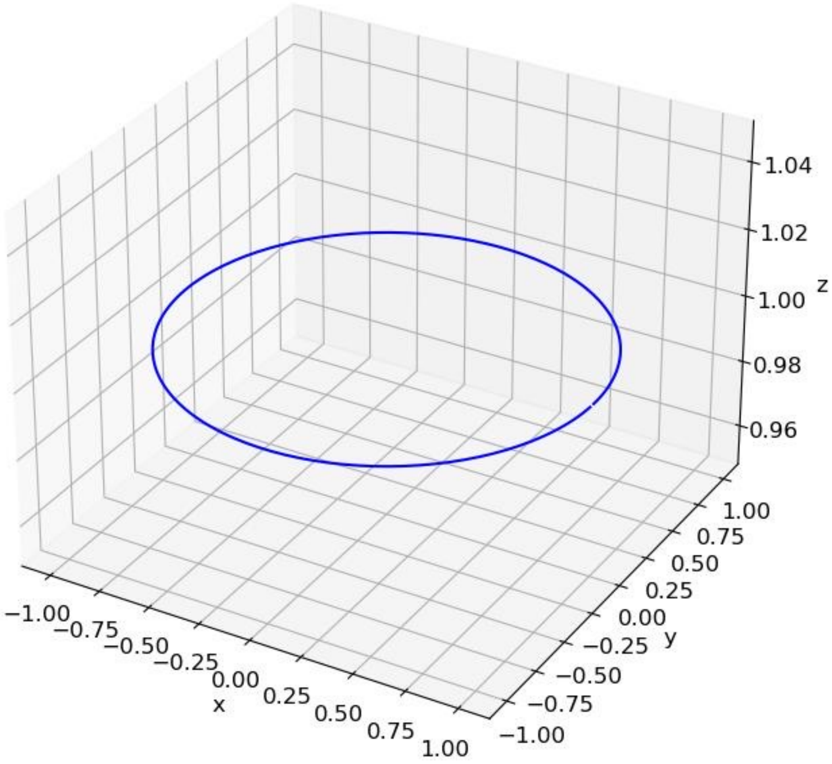

```
pca.explained_variance_ratio_
array([0.98806067, 0.01193933])
```

Answer: Yes, we can by keeping only the first PC describing 98.8% of the variance of the original dataset.

Principal Component Analysis (PCA)

"Toy" Example #2

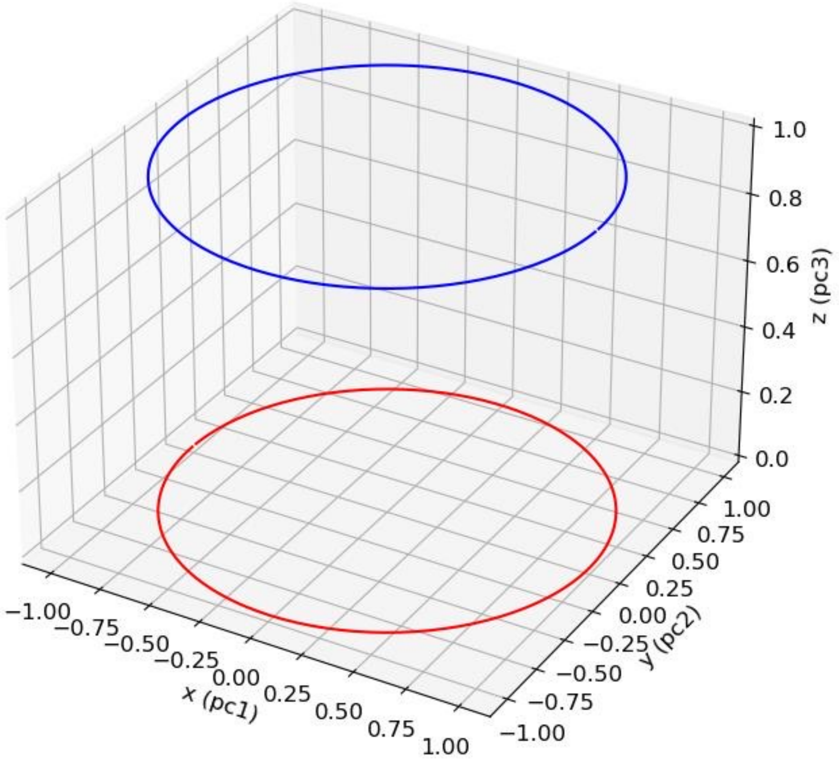
	x	y	z
0	1.000000	0.000000	1.0
1	0.999507	0.031411	1.0
2	0.998027	0.062791	1.0
3	0.995562	0.094108	1.0
4	0.992115	0.125333	1.0
...	...	...	...
195	0.987688	-0.156434	1.0
196	0.992115	-0.125333	1.0
197	0.995562	-0.094108	1.0
198	0.998027	-0.062791	1.0
199	0.999507	-0.031411	1.0



Principal Component Analysis (PCA)

“Toy” Example #2

	pc1	pc2	pc3
0	-0.999737	0.022943	0.0
1	-0.998523	0.054334	0.0
2	-0.996323	0.085672	0.0
3	-0.993141	0.116925	0.0
4	-0.988978	0.148063	0.0
...	...	...	...
195	-0.991017	-0.133732	0.0
196	-0.994729	-0.102538	0.0
197	-0.997459	-0.071242	0.0
198	-0.999205	-0.039876	0.0
199	-0.999964	-0.008471	0.0



Limitations

- **Spurious clustering** may appear under different visualization strategies

- **Outliers** may inflate variance and influence the direction of the principal components

- **Loss of interpretability** from representation in a feature space that doesn’t map directly to measured variables

Data Representation

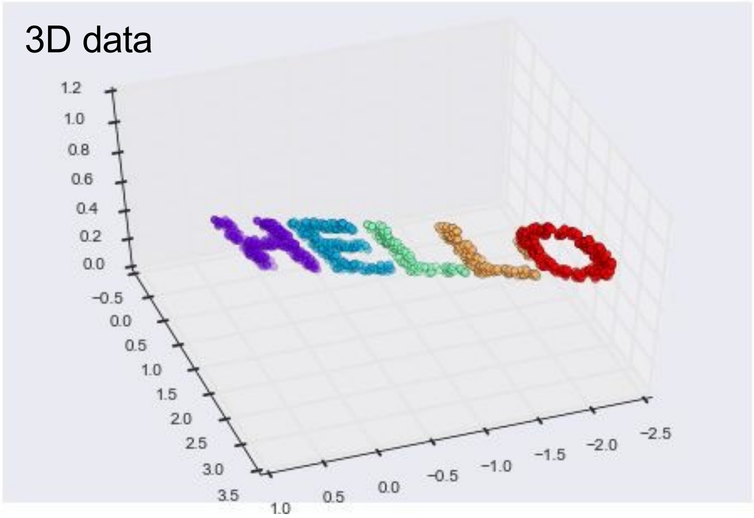
Manifold Learning

Manifold Learning

- Describe data as low-dimensional manifolds embedded in high-dimensional spaces
  - Viewing n-dimensional space through k dimensional space. ($k < n$)
  - e.g. a three dimensional space as a surface.
- Common methods:
  - Multidimensional Scaling (MDS)
  - Locally Linear Embedding (LLE)

Multidimensional Scaling

Find best low-dimensional data representation that preserves all spatial relationships.



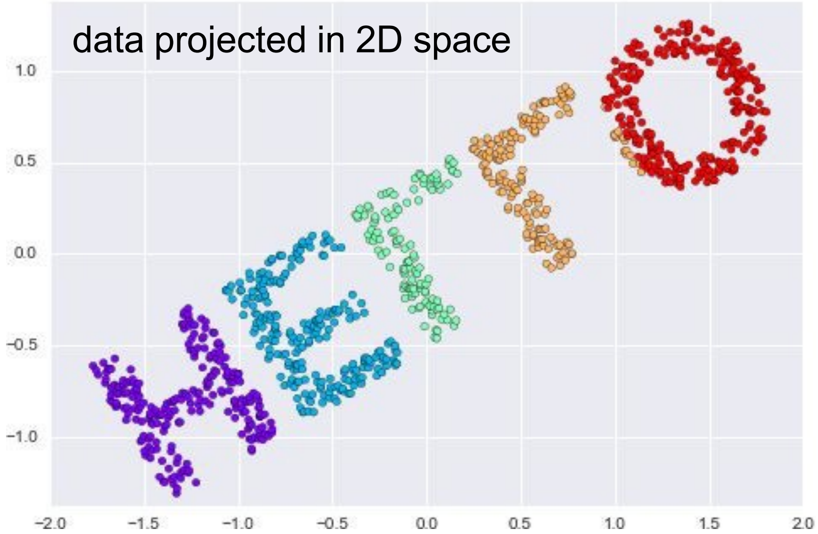
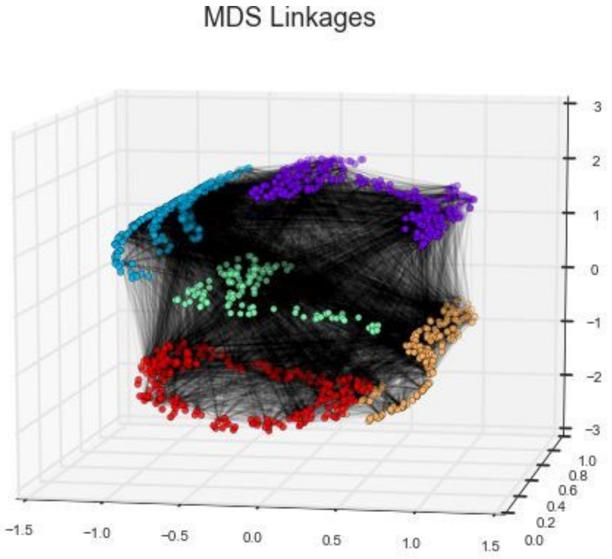


image source

Locally Linear Embedding

Handles nonlinear embeddings by preserving only local distances in neighborhoods of n points.



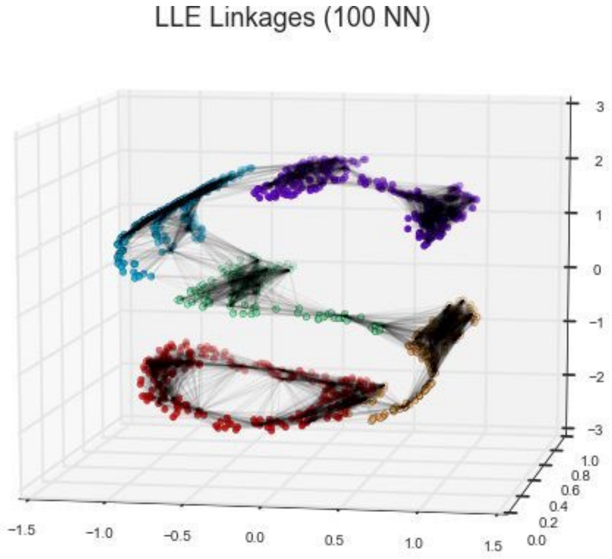


image source

Manifold Learning Summary

Sensitive to noisy and/or missing data

Pairwise distance computation incurs high computational cost

Requires more tuning; LLE sensitive to the choice of neighbors

Other methods:

- Isometric Mapping (isomap)
- t-distributed stochastic neighbor embedding (t-SNE)

Manifold Learning Useful Links

- Wikipedia: Nonlinear Dimensionality Reduction
- "Manifold learning: what, how, and why" by Marina Meila, Hanyu Zhang
- "Locally Linear Embedding and its Variants: Tutorial and Survey" by Benyamin Ghojogh, Ali Ghodsi, Fakhri Karray, Mark Crowley
- Comparison of Manifold Learning Techniques

Hands-On Exercise

Jupyter Set-Up and Data Representation

Hands-on break #1

Setup Materials

- Package Installation
- Plotting Utilities

Data Representation

- Principal Components Analysis (PCA) Example 1: Tumor Classification
    - Exercise 1
- PCA Example 2: Flower Identification
- PCA Example 3: Noise Reduction
- Multidimensional Scaling
- Locally Linear Embedding
- T-SNE